Advances in Intelligent Systems and Computing 446

Thouraya Bouabana-Tebibel
Stuart H. Rubin *Editors*

Theoretical Information Reuse and Integration

Springer

Advances in Intelligent Systems and Computing

Volume 446

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

About this Series

The series "Advances in Intelligent Systems and Computing" contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing.

The publications within "Advances in Intelligent Systems and Computing" are primarily textbooks and proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

Advisory Board

More information about this series at http://www.springer.com/series/11156

Thouraya Bouabana-Tebibel
Stuart H. Rubin
Editors

Theoretical Information Reuse and Integration

Springer

Editors
Thouraya Bouabana-Tebibel
Laboratoire de Communication dans les Systèmes Informatiques
École nationale Supérieure d'Informatique
Algiers
Algeria

Stuart H. Rubin
SPAWAR Systems Center Pacific
San Diego, CA
USA

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-3-319-31309-2 ISBN 978-3-319-31311-5 (eBook)
DOI 10.1007/978-3-319-31311-5

Library of Congress Control Number: 2016934679

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

Preface

Information Reuse and Integration addresses the efficient extension and creation of knowledge through the exploitation of Kolmogorov complexity in the extraction and application of domain symmetry. Knowledge, which seems to be novel, can more often than not be recast as the image of a sequence of transformations, which yield symmetric knowledge. When the size of those transformations and/or the length of that sequence of transforms exceeds the size of the image, then that image is said to be novel or random. It may also be that the new knowledge is random in that no such sequence of transforms, which produces it exists, or is at least known.

The study of novel and symmetric knowledge has great implications for automated knowledge acquisition and automated software creation in general. That is, the extraction of such knowledge is necessarily context-sensitive—implying the use of production rules for tractable learning methodologies. In other words, reuse and integration are inexorably linked, whereby knowledge must be integrated to extract its symmetric variants; and, knowledge must be reused to find tractable pathways for its integration. Notice that theoretical reuse and integration cannot be had in the absence of self-reference. A consequence is that there are truths, which cannot be proven—implying that randomness and symmetry are not absolute, but rather heuristic concepts. That is, knowledge can only be categorized relative to specific criterion. For example, a "do-loop" is random in comparison to an assignment statement, but is symmetric in comparison to a "for" statement—just as an assignment statement is symmetric in comparison to a print statement.

The need for heuristics pervades the real world. These are not mere afterthoughts, meant solely to enable scalability, but serve the goals of randomization itself. As a result, any 5th generation programming system, any knowledge acquisition system, and any mechanics for formal representation must embody a heuristic component if it is to be reusable and integrated in a nontrivial way. For example, the need for multiple representations in problem solving implies the use of heuristics. Similarly, the Japanese 5th generation project failed because it failed to incorporate a heuristic mechanism into the back-cut mechanism of the predicate calculus. The authors of the nine papers comprising this volume incorporate

symmetry, reuse, and integration as overt operational procedures or as operations built into the formal representations of data and operators employed. Either way, the aforementioned theoretical underpinnings of information reuse and integration are supported.

Chapter "Reuse and Integration of Specification Logics: The Hybridisation Perspective" explores the combination and reuse of logics at the syntactic and the semantic levels. This methodology has application to the specification of reconfigurable software systems, where a distinct logic may be used to describe the local requirements of each system's configuration. Chapter "Test Reactive Systems with Büchi-Automaton-Based Temporal Requirements" proposes a specification-based technique that tests a reactive system for a Buchi automaton. The results validate the strength of their approach for improving the effectiveness and efficiency of testing, where the test cases are generated specifically in satisfaction of temporal requirements.

Chapter "Capturing and Verifying Dynamic Systems Behavior using UML and p-calculus" addresses the need for formal semantics for the validation of UML diagrams. In particular, it presents an approach for capturing and verifying the dynamic behavior of systems using UML diagrams and π-calculus. Chapter "A Real-Time Concurrent Constraint Calculus for Analyzing Avionic Systems Embedded in the IMA Connected Through TTEthernet" presents an approach to model and verify avionic systems embedded in the Integrated Modular Architecture (IMA) connected through the TTEthernet Network, by using TTCC, a real-time concurrent constraint process calculus with an operator to define infinite periodic behaviors specific to IMA and TTEthernet. Both operational and declarative aspects of this calculus are shown to comprise a simple and elegant way to specify the requirements of avionic systems.

Chapter "Case Indexing by Component, Context, and Encapsulation for Knowledge Reuse" proposes to provide representation criterion for concept contextualization and encapsulation for reuse in a case-based reasoning (CBR) system. It is argued that these are appropriate representations for case situations and actions, which can be effectively indexed. Chapter "Intelligent Decision Making for Customer Dynamics Management Based on Rule Mining and Contrast Set Mining" provides a business perspective involving the use of data mining techniques to support intelligent decision-making. Both random and symmetric rules are mined with a goal towards improving the decision-making ability of marketing managers.

Chapter "Is Data Sampling Required When Using Random Forest for Classification on Imbalanced Bioinformatics Data?" presents results for the determination if the inclusion of data sampling will improve the performance of the Random Forest classifier (useful for bioinformatics data). It is shown that, in general, data sampling does improve the classification performance of this classifier; although the improved performance is not statistically significant. Chapter "Concurrent Alignment of Multiple Anonymized Social Networks with Generic Stable Matching" addresses connections between the shared users' accounts in multiple social networks (which are called the anchor links), and the problem is formally defined as the M-NASA (Multiple Anonymized Social Networks

Alignment) problem. A novel two-phase network alignment framework UMA (Unsupervised Multi-network Alignment) is proposed in this chapter. Extensive experiments conducted on multiple real-world partially aligned social networks demonstrate that UMA can perform very well in solving the M-NASA problem.

Finally, Chap. "An Accurate Multi-sensor Multi-target Localization Method for Cooperating Vehicles" proposes a cooperative multi-sensor multi-vehicle localization method with high accuracy for terrestrial consumer vehicles. The problem is formulated in the context of a Bayesian framework; and, vehicle locations as well as their velocities are estimated via a Sequential Monte Carlo Probability Hypothesis Density (SMC-PHD) filter. Results provide good predictions of future vehicle locations.

January 2016

Thouraya Bouabana-Tebibel
Stuart H. Rubin

Contents

Contributors

Elham Akhond Zadeh Noughabi Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada

Amir Albadvi Department of Industrial Engineering, Tarbiat Modares University, Tehran, Iran

Luis S. Barbosa HASLab—INESC TEC & University of Minho, Braga, Portugal

Mokhtar Beldjehem University of Ottawa, Ottawa, Canada

Aissam Belghiat MISC Laboratory, Department of Computer Science, University of Mentouri, Constantine, Algeria

Thouraya Bouabana-Tebibel École nationale Supérieure d'Informatique, LCSI Laboratory, Algiers, Algeria

Allaoua Chaoui MISC Laboratory, Department of Computer Science, University of Mentouri, Constantine, Algeria

Asmaa Chebba École nationale Supérieure d'Informatique, LCSI Laboratory, Algiers, Algeria

David J. Dittman Florida Atlantic University, Boca Raton, FL, USA

Abraham O. Fapojuwo Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada

Behrouz H. Far Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada

Abdelouahed Gherbi Department of Software and IT Engineering, École de Technologie Supérieure, Montreal, QC, Canada

Sepideh Afkhami Goli Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada

Kadaouia Habib École nationale Supérieure d'Informatique, LCSI Laboratory, Algiers, Algeria

Sardaouna Hamadou Department of Computer and Software Engineering, École Polytechnique de Montréal, Montreal, QC, Canada

Taghi M. Khoshgoftaar Florida Atlantic University, Boca Raton, FL, USA

Alexandre Madeira HASLab—INESC TEC & University of Minho, Braga, Portugal

Manuel A. Martins CIDMA—Department of Mathematics, University of Aveiro, Aveiro, Portugal

John Mullins Department of Computer and Software Engineering, École Polytechnique de Montréal, Montreal, QC, Canada

Amri Napolitano Florida Atlantic University, Boca Raton, FL, USA

Renato Neves HASLab—INESC TEC & University of Minho, Braga, Portugal

Stuart H. Rubin Space and Naval Warfare Systems Center Pacific, San Diego, USA

Li Tan School of Electrical Engineering and Computer Science, Washington State University, Richland, WA, USA

Philip S. Yu University of Illinois at Chicago, Chicago, IL, USA; Institute for Data Science, Tsinghua University, Beijing, China

Bolong Zeng School of Electrical Engineering and Computer Science, Washington State University, Richland, WA, USA

Qianyi Zhan National Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

Jiawei Zhang University of Illinois at Chicago, Chicago, IL, USA

Reuse and Integration of Specification Logics: The Hybridisation Perspective

Luis S. Barbosa, Manuel A. Martins, Alexandre Madeira and Renato Neves

Abstract Hybridisation is a systematic process along which the characteristic features of hybrid logic, both at the syntactic and the semantic levels, are developed on top of an arbitrary logic framed as an institution. It also captures the construction of first-order encodings of such hybridised institutions into theories in first-order logic. The method was originally developed to build suitable logics for the specification of reconfigurable software systems on top of whatever logic is used to describe local requirements of each system's configuration. Hybridisation has, however, a broader scope, providing a fresh example of yet another development in combining and reusing logics driven by a problem from Computer Science. This paper offers an overview of this method, proposes some new extensions, namely the introduction of full quantification leading to the specification of dynamic modalities, and exemplifies its potential through a didactical application. It is discussed how hybridisation can be successfully used in a formal specification course in which students progress from equational to hybrid specifications in a uniform setting, integrating paradigms, combining data and behaviour, and dealing appropriately with systems evolution and reconfiguration.

Keywords Software specification · Hybrid logic · Hybridization

L.S. Barbosa (✉) · A. Madeira · R. Neves
HASLab - INESC TEC & University of Minho, Braga, Portugal
e-mail: luis.s.barbosa@inesctec.pt

A. Madeira
e-mail: amadeira@inesctec.pt

R. Neves
e-mail: rjneves@inescporto.pt

M.A. Martins
CIDMA - Department of Mathematics, University of Aveiro, Aveiro, Portugal
e-mail: martins@ua.pt

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_1

1 Introduction

Hybrid logic [5, 10, 14, 31] adds to a modal language the ability to name, or to explicitly refer to, specific states of the underlying Kripke structure. This is done through the introduction of propositional symbols of a new sort, called *nominals*, each of which is true at exactly one possible state. Sentences are then enriched in two directions. On the one hand, nominals are used as simple sentences, each of them holding exclusively in the state it names. On the other hand, explicit reference to states is provided by sentences such as $@_i\,\rho$, stating the validity of ρ at the state named i.

Hybrid logic was originally introduced by A. Prior in his book [46], and later revisited, in the school of Sofia, by Passy and Tinchev [47], awakening a broad interest within the modal logic community along the 90s. Our own interest in this generalisation of modal logic was triggered by a concrete problem in (rigorous) software engineering—the specification of *reconfigurable* software systems. The qualifier *reconfigurable* is used for systems whose execution modes, and not only the values stored in their internal memory, may change in response to the continuous interaction with the environment. Such systems behave differently in different modes of operation, or *configurations*, and commute between them along their lifetime.

Present such is more the norm than the exception. A typical, everyday example is offered by cloud based applications that elastically react to client demands. Another example is a modern car in which hundreds of electronic control units must operate in different modes depending on the current situation—such as driving on a highway or finding a parking spot. Switching between these modes is an intuitive example of a dynamic reconfiguration. As a matter of fact, reconfigurability, together with related issues like self-adaptation or context-awarness, became a main research topic [48], in the triple perspective of foundations, methods and technologies.

Clearly, the dynamics of reconfiguration of a software system can be described by some sort of transition system, whose states represent configurations and transitions are triggered by whatever conditions enforce a switch of configurations. However, one needs also to capture the specific, *local* requirements which characterise each configuration and distinguish one from the others. Formally, such different behaviours can be modelled by imposing additional structure upon the states of the transition system which expresses the overall dynamics.

This path was explored in our previous work [35] on a specification methodology for reconfigurable systems. The basic insight is that, starting from a classical state-machine specification, each state, regarded as a possible system's configuration, is equipped with a rich mathematical structure to describe its functionality. Technically, specifications become structured state-machines whose states denote algebras or first order structures, rather than sets. Such a specification should be able to make assertions both about the transition dynamics and, locally, about each particular configuration. This explains why hybrid logic was chosen as the *lingua franca* for the envisaged methodology. One may therefore specify (local) properties of specific configurations in the system or even assert the equality between two

particular configurations, something that is beyond what can be said in a modal language. Modalities, however, capture state transitions, providing a way to specify the *global* dynamics of reconfigurability.

For the working software architect, the relevant question goes a step forward: the envisaged methodology should be *independent* of whatever logic is found appropriate to express *local* requirements for each configuration. Actually, specific problems do require specific logics to describe their configurations (e.g., equational, first-order, fuzzy, etc.). Therefore, instead of choosing a particular version of hybrid logic, the method proposed in [35] starts by choosing a specific logic to express requirements at the configuration level. This is later taken as the *base* logic on top of which the characteristic features of hybrid logic are developed.

Such a process along which the characteristic features of hybrid logic, both syntactical and semantical, are developed on top of a given logic, in a parametric way, is called *hybridisation*, and was proposed in Madeira Ph.D. thesis [34], whose core results were published in Refs. [22, 23, 39]. Going generic entailed the need for a proper abstract foundation. Therefore, the whole approach is framed in the context of the theory of institutions of Goguen and Burstall [20, 25], each logic (base and hybridised) being treated abstractly as an institution.

As discussed in the sequel, hybridisation techniques not only offer a main conceptual tool for dealing with reconfigurable systems, but are also valuable in designing innovative teaching approaches in Software Engineering.

Aims. In such a context, this paper has a triple objective. First of all, it offers an overview of this method, emphasising conceptual exposition, rather than the purely technical style the interested reader may find in the references above. Secondly it exemplifies its potential through a *didactical* application, as a follow up to the original workshop paper [38]. The focus is on how the method can provide ways of reusing and integrating different specification logics in an undergraduate course on formal software specification. This leads to the design of a new course along which students progress from equational to hybrid specifications in a uniform setting, integrating paradigms, combining data and behaviour, and dealing appropriately with systems evolution and reconfiguration. Finally, it extends the method in two directions: (i) computational support for the translation of system's requirements in the format of boilerplates to $\mathcal{H}$CASL; (ii) introduction of full quantification in the method providing a way to specify dynamic modalities and, in general, the change 'on-the-fly' of the transition relation.

Paper structure. The hybridisation method is described and illustrated in the next section. Section 3 discusses the integration of the method in the HETS platform, therefore providing effective tool support to (some families of) hybridised specifications. Its didactical use in an introductory course to formal software specification is the subject of Sects. 4 and 5. Section 6 extends the method to deal with full quantification, which forms the main, original contribution of the paper. Finally, Sect. 7 reviews related work in the area of combination of logics and concludes pointing out current research directions.

2 The Hybridisation Method

2.1 Institutions

An *institution* is an abstract formalisation of a logical system, encompassing syntax, semantics and satisfaction. The concept was put forward by Goguen and Burstall, in the end of the Seventies, in order to "*formalise the formal notion of logical systems*", in response to the "*population explosion among the logical systems used in Computing Science*" [25].

The universal character of institutions proved effective and resilient as witnessed by the wide number of logics it was able to formalise. Examples range from the usual logics in classical mathematical logic (propositional, equational, first order, etc.), to the ones underlying specification and programming languages or used for describing particular systems from different domains. Well-known examples include *probabilistic logics* [9], *quantum logics* [18], *hidden and observational logics* [6, 8], *coalgebraic logics* [15], as well as logics for reasoning about *process algebras* [42], *functional* [50, 52] and *imperative programing languages* [52].

The theory of institutions (see [20] for an extensive account) was motivated by the need to abstract from the particular details of each individual logic and to characterise fundamental concepts, such as satisfaction and combination of logics, in very general terms. This lead to the development of a solid *institution-independent specification theory*, on which, structuring and parameterisation mechanisms, required to scale up software specification methods, are defined 'once and for all', irrespective of the concrete logic used in each application domain.

Formally, an institution

$$I = \left(\mathrm{Sign}^{\mathcal{I}}, \mathrm{Sen}^{\mathcal{I}}, \mathrm{Mod}^{\mathcal{I}}, (\models^{I}_{\Sigma})_{\Sigma \in |\mathrm{Sign}^{\mathcal{I}}|}\right)$$

consists of a category $\mathrm{Sign}^{\mathcal{I}}$ of *signatures* and *signature* morphisms; a functor $\mathrm{Sen}^{\mathcal{I}}$, $\mathrm{Sen}^{\mathcal{I}} : \mathrm{Sign}^{\mathcal{I}} \to \mathbb{S}et$, giving for each signature a set of *sentences* over that signature; another functor $\mathrm{Mod}^{\mathcal{I}} : (\mathrm{Sign}^{\mathcal{I}})^{op} \to \mathbb{C}AT$, providing for each signature Σ a category of *Σ-models* and *Σ-(model) homomorphisms*, and, finally, a satisfaction relation.

Note that each morphism of signatures $\varphi : \Sigma \to \Sigma' \in \mathrm{Sign}^{\mathcal{I}}$ induces a semantic map, i.e., a functor $\mathrm{Mod}^{\mathcal{I}}(\varphi) : \mathrm{Mod}^{\mathcal{I}}(\Sigma') \to \mathrm{Mod}^{\mathcal{I}}(\Sigma)$ called the *reduct functor*, whose effect is to cast a model of Σ' as a model of Σ. Therefore, the satisfaction relation $\models^{I}_{\Sigma} \subseteq |\mathrm{Mod}^{\mathcal{I}}(\Sigma)| \times \mathrm{Sen}^{\mathcal{I}}(\Sigma)$, for each $\Sigma \in |\mathrm{Sign}^{\mathcal{I}}|$, verifies the following condition, which, for each signature morphism φ, entailing a syntactic transformation, captures the basic principle of *truth invariance under change of notation* [25]:

$$M' \models^{I}_{\Sigma'} \mathrm{Sen}^{\mathcal{I}}(\varphi)(\rho) \;\; \text{iff} \;\; \mathrm{Mod}^{\mathcal{I}}(\varphi)(M') \models^{I}_{\Sigma} \rho$$

2.2 The Method

This section reviews the *hybridisation* method proposed in [23, 39]. The method enriches a base (arbitrary) institution I with hybrid logic features and the corresponding Kripke semantics. The result is still an institution, $\mathcal{H}I$, called the *hybridisation of* I. In the sequel we concentrate in a simplified version, i.e., quantifier-free and non-constrained, of the general method, to convey the basic intuitions.

At the syntactic level the base signatures are enriched with nominals and polyadic modalities. Therefore, the category of *I-hybrid signatures*, denoted by $\mathrm{Sign}^{\mathcal{H}I}$, is defined as the direct (cartesian) product of categories of the original category of signatures $\mathrm{Sign}^{\mathcal{I}}$ and that of signatures of *REL*, the sub-institution of (the institution of) first order logic, without non-constant operation symbols, Sign^{REL}. Signatures of the hybridised institution combine those of I with a set of constants Nom for *nominals* and a set of relational symbols Λ to represent *modalities*. $\mathcal{H}I$ signatures are, thus, triples $(\Sigma, \mathrm{Nom}, \Lambda)$, with signature morphisms $\varphi = (\varphi_{\mathrm{Sig}}, \varphi_{\mathrm{Nom}}, \varphi_{\mathrm{MS}}) : (\Sigma, \mathrm{Nom}, \Lambda) \to (\Sigma', \mathrm{Nom}', \Lambda')$, defined component-wise: the first component is inherited from I and the others simply map nominals and modalities while preserving the arities of the latter.

The second step in the method is to enrich the base sentences accordingly. The sentences of the base institution I and the nominals in Nom are taken as atoms and composed with the Boolean connectives, the modalities in Λ, and satisfaction operators indexed by nominals. For example, for a n-ary modality λ, a nominal i and $\mathcal{H}I$-sentences $\rho, \rho_1, \rho_2 \ldots, \rho_n$, the following are also sentences in $\mathcal{H}I$: $[\lambda](\rho_1, \ldots, \rho_n)$, $\langle\lambda\rangle(\rho_1, \ldots, \rho_n)$ and $@_i\rho$.

Given a $\mathcal{H}I$-signature morphism φ, the translation of sentences $\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\varphi)$ is defined structurally: e.g.,

$$\begin{aligned}
\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\varphi)(i) &= \varphi_{\mathrm{Nom}}(i) \\
\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\varphi)(@_i\rho) &= @_{\varphi_{\mathrm{Nom}}(i)}\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\rho) \text{ and} \\
\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\varphi)([\lambda](\rho_1, \ldots, \rho_n)) &= [\varphi_{\mathrm{MS}}(\lambda)](\mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\rho_1), \ldots, \mathrm{Sen}^{\mathcal{H}\mathcal{I}}(\rho_n))
\end{aligned}$$

Models of $\mathcal{H}I$ can be regarded as (Λ-)Kripke structures whose worlds are I-models. Formally, they are pairs (M, W) where W is a (Nom, Λ)-model in *REL* and M is a function which assigns to each state $w \in W$ a model $M(w) \in |\mathrm{Mod}^{\mathcal{I}}(\Sigma)|$. We denote $M(w)$ simply by M_w.

In each world (M, W), W_n provides an interpretation for nominal n, whereas relation W_λ interpretes modality λ. The reduct definition is lifted from the base institution: the reduct of a Δ'-model (M', W') along a signature morphism $\varphi : \Delta \to \Delta'$ is the Δ-model (M, W) such that W is the $(\varphi_{\mathrm{Nom}}, \varphi_{\mathrm{MS}})$-reduct of W' (i.e., $|W| = |W'|$, $W_n = W'_{\varphi_{\mathrm{Nom}}(n)}$, for each nominal n, and $W_\lambda = W'_{\varphi_{\mathrm{MS}}(\lambda)}$ for each modality in Λ).

Finally, the satisfaction relation for the hybridised institution resorts to the one in the base institution for sentences in I, i.e.,

- $(M, W) \models^w \rho$ iff $M_w \models^I \rho$ when $\rho \in \mathrm{Sen}^I(\Sigma)$,

captures the semantics of nominals

- $(M, W) \models^w i$ iff $W_i = w$, when $i \in \mathrm{Nom}$
- $(M, W) \models^w @_j \rho$ iff $(M, W) \models^{W_j} \rho$

and modalities, as in

- $(M, W) \models^w [\lambda](\xi_1, \ldots, \xi_n)$ iff, for any $(w, w_1, \ldots, w_n) \in W_\lambda$, $(M, W) \models^{w_i} \xi_i$ for some $1 \leq i \leq n$

and is defined as usual for the Boolean connectives.

The main result is that $\mathcal{H}I$ effectively constitutes an institution [39]. The next step is the systematic characterisation of encodings of the hybridised institution $\mathcal{H}I$ into the institution of many sorted first-order logic (*FOL*) building on existent encodings of the base institution I into *FOL*. This is discussed below in Sect. 3.

2.3 Examples

Propositional logic. Propositional logic gives rise to a well-known institution *PL* whose signatures are sets of propositional symbols and signature morphisms are functions between them. Models assign truth values to propositions and interpret propositional sentences, built with the Boolean connectives, in the usual way.

The hybridisation of the institution of propositional logic *PL* introduces nominals and modalities resulting in an institution whose sentences are generated by

$$\rho ::= \rho_0 \mid i \mid @_i \rho \mid \rho \odot \rho \mid \neg \rho \mid \langle \lambda \rangle(\rho, \ldots, \rho) \mid [\lambda](\rho, \ldots, \rho)$$

where ρ_0 is a sentence inherited from *PL*, $\odot = \{\vee, \wedge, \Rightarrow\}$, and i and λ stand, respectively, for a nominal and a modality symbol. Note there is a double level of connectives in the sentences: one coming from base *PL*-sentences and another introduced by the hybridisation process. However, they "semantically collapse" and, hence, no distinction between them needs to be done (see [23] for details). A $\mathcal{H}PL$ model has a transition structure to interpret each added modality. Each world comes equipped with a *PL*-model, i.e., a particular subset of propositions holding locally.

As one would expect, restricting signatures to those with just a single unary modality results in the usual institution for classical hybrid propositional logic [14].

Propositional fuzzy logic. Many-valued logics [26] generalise classic logics by replacing, as their *truth domain*, the 2-element Boolean algebra, by larger sets structured as complete residuated lattices. A residuated lattice includes an associative, monotonic binary operation $\otimes$, with the biggest element as the identity and such that there exists an element $x \Rightarrow z$ verifying $y \leq (x \Rightarrow z)$ iff $x \otimes y \leq z$. They were originally formalised as institutions in [21].

Given a complete residuated lattice L, an institution MVL_L is defined based on *PL*-signatures, but whose sentences are pairs (ρ, p) formed by an element p of L

and a PL-sentence ρ defined over the usual Boolean connectives and $\otimes$. Models are functions evaluating propositions on the lattice, rather than on the Boolean domain. Accordingly, a sentence (ρ, p) is satisfied in a model M if p is less or equal the evaluation of sentence ρ in M.

This institution captures many many-valued logics discussed in the literature. For instance, taking L as the Łukasiewicz arithmetic lattice over the closed interval [0, 1], where $x \otimes y = 1 - max\{0, x + y - 1)\}$ (and $x \Rightarrow y = min\{1, 1 - x + y\}$), yields the standard *propositional fuzzy logic*.

The institution obtained through the hybridisation of MVL_L, for a fixed L, is similar to $\mathcal{H}PL$ but for two aspects: sentences are defined as in $\mathcal{H}PL$ but taking sentences (ρ_0, p) as atomic; and a function assigning to each proposition a value in L, is associated to each world.

Note that expressivity increases even in the restricted case of a (one-world) standard semantics. Differently from what happens in the base logic, where each sentence is tagged by a L-value, in the hybridised institution expressions may involve different L-values, as in, for example, $(\rho, p) \wedge (\rho', p')$. The reason for this is the introduction of Boolean connectives by the hybridisation process.

Equational logic. Signatures in the institution EQ of equational logic are pairs (S, F) where S is a set of sort symbols and $F = \{F_{\underline{ar} \to s} \mid \underline{ar} \in S^*, s \in S\}$ is a family of sets of operation symbols indexed by arities $\underline{ar}$ (for the arguments) and sorts s (for the results). Signature morphisms map both components in a compatible way. A model for a given signature is an algebra interpreting each sort symbol as a carrier set and each operation symbol as a function; model morphisms are, of course, homomorphisms of algebras. Sentences are universal quantified equations $(\forall X)t = t'$ and the satisfaction relation is the usual Tarskian satisfaction defined recursively on the structure of the sentences.

The hybridisation of EQ gives rise to an institution $\mathcal{H}EQ$ whose signatures are triples $((S, F), \text{Nom}, \Lambda)$ and the sentences are defined as in the previous examples, but taking (S, F)-equations $(\forall X)t = t'$ as atomic base sentences instead. Models are Kripke structures with a (local) (S, F)-algebra associated to each world.

3 Hybridisation at Work

Hybridised logics provide an interesting framework to specify and reason about reconfigurable software systems. As explained above, models for reconfigurable software can be regarded as structured transition systems, whose states represent individual configurations with whatever structure they have to bear in concrete applications. Transitions, on the other hand, correspond to the admissible reconfigurations. For example, if local requirements are captured equationally, as they often are in formal specification methods, distinct configurations can be modelled by distinct algebras. Clearly, specifications are given equationally, based on EQ-signatures. Nominals identify the "relevant" configurations, and reconfigurations amount to state transitions. Therefore, one resorts to equations tagged with the satisfaction operators to

specify configurations; plain equations to specify the system global properties and modal features to specify its reconfiguration dynamics.

The key ingredient to make these ideas appealing for the working software engineer is the existence of computer-based support for reasoning about specifications in logics obtained by hybridisation. Technically, this amounts to the existence of tools to transport specifications from a logical system to another, with more effective proof support. This is done through the systematic characterisation of encodings of hybridised institutions into *FOL*, the institution of *many sorted first-order logic*. In this section we discuss such encodings and the tool support they provide on top the HETS platform [40].

3.1 First-Order Encodings

As mentioned above, for each institution "encodable" in *FOL* theories, there is a method to construct an encoding from its hybridisation to *FOL*. Therefore, a wide variety of computer assisted provers for first order logic can be "borrowed" to reason about specifications in the new, hybridised logics.

Technically such encodings extend the classical *standard translation* of modal logic into the (one-sorted) first order logic [53], more precisely, of its hybrid version [10], to the encodings of hybridised institutions into *FOL*.

The standard translation from hybrid propositional logic $\mathcal{H}PL$ into the (one-sorted) first-order logic introduces a new sort to encode the state space, interprets nominals as constants, modalities as binary relations, and propositions as unary predicates encoding the validity of each proposition in each state. Brauner [14] extends this encoding in devising the translation from hybrid first order logic $\mathcal{H}FOL$ to *FOL*. Basically, he introduces a new universe as an extra sort in the signature, and "flattens" the universes, operations and predicates of the (local) *FOL*-models to an unique (global) *FOL*-model. Local functions and predicates become parametric over states, and the state universes distinguished with a sort-family of definability predicates. Intuitively, whenever m belongs to the universe of w, $\pi(w, m)$ and $\sigma(w, m) = b$ means that $\pi(m)$ and $\sigma(m) = b$ hold in state w. The restriction of this global model M to the local universes, operations and predicates of a fixed word w, gives rise to a "slice of M", say $M|_w$, i.e., a local *FOL*-model which represents (and coincides with) M_w.

A similar method, based on a state-parametric construction, is used in our context to lift *I2FOL* to $\mathcal{H}I2FOL$. Thus, all the signatures and sentences targeted by *I2FOL* become parametric on states. A slice $M|_w$ corresponds now to the "*FOL*-interpretation" of the local I-model M_w, which can be recovered using *I2FOL*. Actually, this process can be understood as a *combination of logic encodings* between the standard translation of hybrid logic into *FOL* and other encodings into *FOL*.

Such encodings are required to be conservative "theoroidal comorphisms" [27, 41], i.e., they are supposed to map signatures to theories. Conservativity, i.e., requirement that models are translated through surjections, is a sufficient condition to use

such maps as actual encodings. In particular, this is necessary in order to borrow from *FOL* proof resources in a sound and complete way. This entails the need for an abstract characterisation of conservativity which appeared in [23]. This reference also extends the method originally proposed in [39] for generating first-order encodings in hybridised institutions to theories, constrained models and quantified sentences.

Constrained models provide a very general way to introduce sharing constraints into the picture. Those are traditionally modelled via the so-called "rigid" syntactic entities, which means that some sorts, functions, or predicates are designated as "rigid" and consequently their interpretations are invariant across possible worlds. Constrained models are indispensable for having encodings into first-order logic, more precisely to reflect the consequence relation (see [22] for a detailed account).

3.2 *Implementation in the* HETS *Platform*

Encodings, as discussed above, provide the right path to transport specifications from a logical system to another offering more effective, computer-based proof support. HETS has been described as a "motherboard" of logics where different "expansion cards" can be plugged in. These are individual logics (with their particular analysers and proof tools) as well as logic translations. To make them *compatible*, logics are formalised as institutions and translations as comorphisms. Therefore, the integration of hybrid specifications in the HETS platform is legitimate, since all formal requirements (e.g., that institutions exist, that comorphisms can be defined, etc.) are already guaranteed by the hybridisation process itself.

This implementation was done along two different directions, both documented in [43]. Firstly the general hybridisation method was incorporated in HETS, making available parsing and static analysis for the hybridisation of any base institution already supported by this platform. Secondly, the encoding along the comorphism $\mathcal{H}$CASL $\rightarrow$ CASL was implemented, offering effective tool support for proofs on a number of $\mathcal{H}$CASL-sub-institutions, namely $\mathcal{H}PL$ and $\mathcal{H}FOL$. Institution $\mathcal{H}$CASL consists of the hybridisation of the institution for CASL [36], the platform *lingua franca*, with the models restricted to those with common realisation of sorts in all the states and of the quantified variables. This provides for free the proof support environment of a particularly well established logic. The implementation of the hybridisation method in HETS proved an effective and flexible way to prove properties of hybrid specifications and thus to support the design method in [35, 37].

3.3 *An Example*

Figure 1 depicts the setting for a toy, yet illustrative example of a hybrid specification and its encoding. The system is a "swinging" calculator with only one operation which can be interpreted in two possible modes. In one of them it adds two natural numbers,

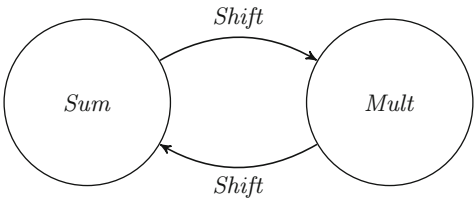


Fig. 1 The swinging calculator

in the other multiplies them. One switches between these two modes through the *Shift* command.

The underlying Kripke frame is specified as follows:

modalities *Shift*
nominals *Sum*, *Mult*
$@Sum \neg Mult$
$Sum \vee Mult$
$@Sum\ (\langle Shift\rangle\ Mult \wedge [Shift]\ Mult)$
$@Mult\ (\langle Shift\rangle\ Sum \wedge [Shift]\ Sum)$

The first axiom rules out models where *Sum* and *Mult* would collapse into each other. The second one restricts to models which admit at most two possible modes. Thus all valid Kripke frames for this example will have precisely the two desired modes of operation. Transitions between them (i.e., the reconfiguration dynamics) are characterised by the last two sentences. The "reconfigurable" operation is declared in the calculator's "global" signature:

op $\_\#\_ : Nat \times Nat \to Nat$

Global properties of the calculator, for example # commutativity and associativity, can be specified as follows,

$\forall\ n, m, p : Nat$
- $n \# m = m \# n$
- $(n \# m) \# p = n \# (m \# p)$

The behaviour of #, however, needs to be defined locally, i.e. relative to each possible mode of operation, *Sum* and *Mult*. Thus,

$\forall\ n, m : Nat$
- $@Sum\ n \# 0 = n$
- $@Sum\ n \# suc(m) = suc(n \# m)$
- $@Mult\ n \# 0 = 0$
- $\exists\ p, q : Nat$
  - $@Mult\ n \# suc(m) = p \wedge @Sum\ n \# q = p \wedge @Mult\ n \# m = q$

which concludes the specification. Note that the last sentence represents the equation $n * (m + 1) = n + (n * m)$, where $+$ and $*$ are, respectively, the usual addition and multiplication of natural numbers. The translation of these axioms to CASL proceeds as described above, with the introduction of a new sort to encode the state space upon which nominals are interpreted as constants (Wrl_Sum and Wrl_Mult, respectively). The translation of the two axioms characterising the behaviour of # in the *Sum* mode is as follows:

$$\begin{array}{l}\forall\ world : World \\ \bullet\ \forall\ n : Nat \bullet (\#(Wrl\_Sum, n\ 0(Wrl\_Sum))) = n\end{array}$$

$$\begin{array}{l}\forall\ world : World \\ \bullet\ \forall\ n, m : Nat \\ \quad \bullet\ (\#(Wrl\_Sum, n, suc(Wrl\_Sum, m))) \\ \quad\ = (suc(Wrl\_Sum, (\#(Wrl\_Sum, n, m))))\end{array}$$

The next step is to check for properties. For illustration purposes, consider the three properties below. The first one states monotonicity of addition; the second the cyclic character of the *Shift* modality; and the third represents the equation $n + n = n * 2$.

$$\begin{array}{ll}\forall\ n, m, r : Nat & \\ \bullet\ @Sum\ (n < m \Rightarrow n < m\ \#\ r) & \%1\% \\ \bullet\ \exists\ p : Nat & \\ \quad \bullet\ @Sum\ n\ \#\ m = p \wedge @Sum < Shift > < Shift > n\ \#\ m = p & \%2\% \\ \bullet\ \exists\ p : Nat \bullet @Sum\ n\ \#\ n = p \Rightarrow @Mult\ n\ \#\ suc(suc(0)) = p & \%3\%\end{array}$$

The CASL-translations computed for these properties are, respectively,

$$\begin{array}{l}\forall\ world : World \\ \bullet\ \forall\ n, m, r : Nat \\ \quad \bullet\ <(Wrl\_Sum, n, m) \\ \quad\ \Rightarrow\ <(Wrl\_Sum, n, \\ \qquad\ (\#(Wrl\_Sum, m, r : Nat))) \qquad \%1\%\end{array}$$

$$\begin{array}{l}\forall\ world : World \\ \bullet\ \forall\ n, m : Nat \\ \quad \bullet\ \exists\ p : Nat \\ \quad\ \bullet\ (\#(Wrl\_Sum, n, m)) = p \\ \quad\ \wedge\ \neg\ \forall\ world0 : World \\ \qquad\ \bullet\ Acc\_Shift(Wrl\_Sum, world0) \\ \qquad\ \Rightarrow\ \forall\ world1 : World \\ \qquad\ \bullet\ Acc\_Shift(world0, world1) \\ \qquad\ \Rightarrow\ \neg\ (\#(world1 : World, n, m)) = p \qquad \%2\%\end{array}$$

$$\begin{array}{l}\forall\ world : World \\ \bullet\ \forall\ n : Nat \\ \quad \bullet\ \exists\ p : Nat\end{array}$$

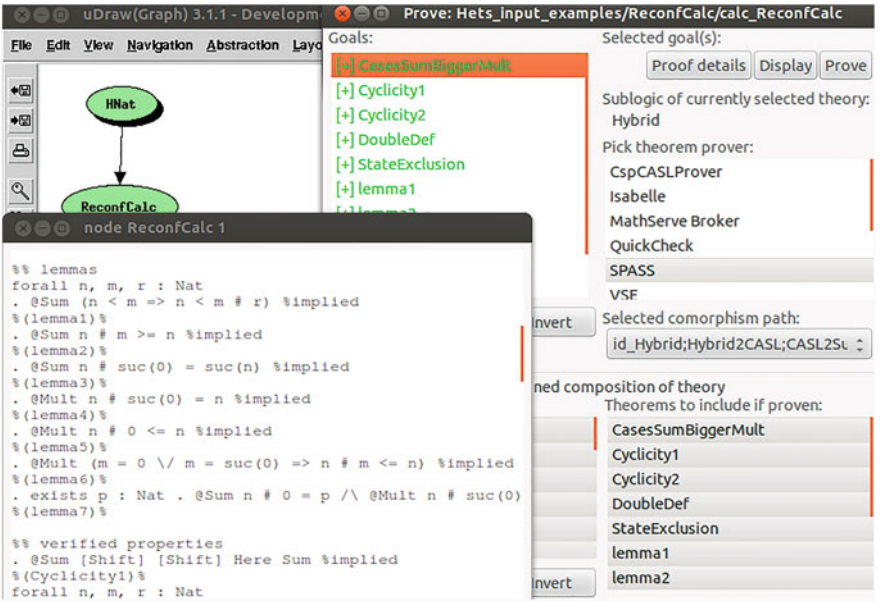


Fig. 2 A HETS session for the swinging calculator

$$\bullet\ (\#(Wrl\_Sum, n, n)) = p \Rightarrow (\#(Wrl\_Mult, n, suc(Wrl\_Mult, suc(Wrl\_Mult, 0(Wrl\_Mult))))) = p \qquad \%3\%$$

Once translated, all these properties are easily proved by one of the provers plugged into the HETS platform, for example SPASS. Figure 2 registers an HETS session relative to this example showing the proof window, part of the model theory, and the specification graph.

3.4 *From Boilerplates to $\mathcal{H}$CASL Specifications*

In order to facilitate the use of hybridised logics in real world specification projects, a language of boilerplates for modelling requirements of reconfigurable systems was proposed by the authors [37]. In the discipline of requirements engineering, a *boilerplate* [29] is defined as a simplified, normative English text, intended to capture software requirements in a controlled way. It is supposed to be highly reusable and amenable to some form of computer-based simulation.

The term derives from steel manufacturing, where it refers to steel rolled into large plates for use in steam boilers. The intuition is that a boilerplate has been time-tested and is "strong as steel" suitable for repeated reuse. Our starting point in the

above cited paper was that *the use of "controlled natural language" for requirements elicitation is a successful practice in industry and, despite of its informal character, provides an interesting starting point towards more formal approaches* [37].

This approach is extended in the present paper by providing a systematic translation scheme of this language of boilerplates to hybridised specifications in $\mathcal{H}$CASL. Once the system's requirements are captured by a collection of boilerplates which, taken jointly, specify a structured transition system, a formal specification is generated in $\mathcal{H}$CASL. The latter can then be handled through HETS . Its states, corresponding to different *configurations*, or *modes of execution*, are endowed with a specific description of the functionality available locally. The boilerplates define globally the relevant modes of execution and the transition structure, as well as, at the local level, the interface of services available and their properties.

The role of this tool is illustrated through the *swinging calculator* example discussed above. Figure 3 shows a fragment of the relevant requirements captured as boilerplates. The language comprises different classes of boilerplates to deal with different kinds of requirements. Figure 4 contains the translator output, i.e., the derived $\mathcal{H}$CASL specification. At this stage both texts offer no difficulty and the reader can appreciate the translation process. Note, however, that specifications of real systems can become rather complex, which advises the use of boilerplates. On the other hand, it should also be mentioned that not all design features can be suitably expressed through boilerplates, a few of them requiring some fine tuning directly over the specification. A complete account of the language of boilerplates is given in the paper mentioned above [37].

```
System's interface is defined by {
 sorts Nat
 op __#__ : Nat * Nat -> Nat
 op 0 : Nat
}.

System has events Shift.
System has modes Sum, Mult.

Property Mult does not hold in mode Sum.
Either mode Sum is active or mode Mult is active.

System changes from Sum to Mult through event Shift.
System may change from Sum to Mult through event Shift.
System changes from Mult to Sum through event Shift.
System may change from Mult to Sum through event Shift.

Property forall n,m, p: Nat. n # (m # p) = (n # m) # p holds in all modes.
Property forall n, m: Nat. n # m = m # n  holds in all modes.
Property forall n,m: Nat. n # 0 = n holds in mode Sum.
```

Fig. 3 Requirements for the swinging calculator encoded in boilerplates

```
logic Hybrid
spec X =
 sorts Nat
 op __#__ : Nat * Nat -> Nat
 op 0 : Nat

modalities Shift
nominals Mult,Sum

 . @ Sum  not Here Mult
 . Here Sum \/ Here Mult
 . @ Sum < Shift > Here Mult
 . @ Sum [ Shift ] Here Mult
 . @ Mult < Shift > Here Sum
 . @ Mult [ Shift ] Here Sum
 .  forall n,m, p: Nat. n # (m # p) = (n # m) # p
 .  forall n, m: Nat. n # m = m # n
 . @ Sum  forall n,m: Nat. n # 0 = n
end
```

Fig. 4 The derived $\mathcal{H}$CASL specification

The first boilerplate describes the system interface at each local state. Then the relevant configurations (`Sum` and `Mult`) are declared as well as the event labelling the transition from one to the other. The definition of the configurations proceeds with the third group of boilerplates which describes a number of properties to be respected. The transition structure is described afterwards; notice how expression "changes" is translated to a "diamond" modality (emphasising that an effective transition will take place), whereas expression "may change" leads to a "box" modality: the event under consideration, if present, can only result in such a transition. Finally, the last lines in Fig. 3 are examples of boilerplates for capturing properties of the system's functionality at different configurations.

4 An Application to the Design of a Specification Course

The ideas behind hybridisation and hybridised logics were further tested in the design of a specification course in the curriculum of the Computer Science undergraduate degree at Universidade Minho, Portugal. The underlying motivation was to explore a uniform framework for specifying system's requirements either *functional* (i.e., relative to the meaning of individual services or operations) or *behavioural* (i.e., relative to its overall evolution and reaction to external stimulus), and to emphasise a strong connection between *modelling* and *verification*.

The course rationale. The course has a standard typology: a lecture per week (1 h), an exercises class devoted to pen-and-pencil resolution of exercises previously

proposed and their discussion (2 h) and a laboratory session with the HETS system (1 h). Students work in groups of two elements.

The course develops around a triangle whose vertices are repeatedly revisited: the *models*, the *languages* in which such models and their properties are expressed and the *satisfaction relation* between them, which enables property verification and design assessment. Another methodological option concerned the adoption of a *generic framework*, in which progressively more elaborated requirements could be represented, in contrast to one with a narrower scope or clearly oriented to a particular specification style. This has the advantage of focusing students and enhancing their ability to work at higher abstraction levels.

This favoured the choice of an institutional approach and the hybridisation method described in the previous sections, computationally supported by the HETS framework.

The course structure. As expected, the course targets *reconfigurable* systems, whose components may evolve in time through a number of different stages or modes of operation, in which specific service configurations are made available through their interfaces. The envisaged teaching/learning process develops around three specification stages: *algebraic*, *modal* and *hybrid*. The idea is to cover the whole spectrum of basic specification logics in three course units, all of them sharing HETS as the common tool support. A fourth unit in the syllabus explores a number of case-studies in the project of reconfigurable systems. The course illustration in Sect. 5 is taken from this last unit. Before that, let us review the *rationale* under each of them.

The algebraic stage. At a first stage each system *configuration* is specified axiomatically as a "stand-alone" *algebraic theory*; its model being a concrete algebra satisfying such a theory. Component's functionality is therefore given in terms of input-output relations modeling operations on *data*. This stage covers the classical concepts in algebraic specification, namely those of *signature*, *sentence*, *equation* and equational reasoning, *model* and *satisfaction of an equation*. The envisaged learning outcome is the ability to master these concepts and capture informal requirements about component's functionality by defining a (syntactic) *universe of discourse* and formulating properties as axioms.

The modal stage. The second stage emphasises the *reactive* nature of the systems at hands. Component's evolution is modelled by a transition system: a configuration changes in response to a particular event in the system. Modal logics are introduced as specification languages for state transition systems. Modal formulas are evaluated inside such systems, at a particular state, and modal operators disclose access to information stored at other states accessible from the current one via a suitable transition. The main learning outcome is to make students familiar with the modal framework and the meaning of modalities as a language to specify transition structures.

The hybrid stage. The third stage starts with a crucial observation: functional and transitional behaviour are strongly interconnected in practice as the functionality offered by the system, at each moment, may depend on the stage of its evolution. This entails the need for

- enriching the basic modal language with the ability to refer to *individual* states, regarded as possible system's configurations or modes of operation;
- distinguishing *global* behaviour (in the underlying transition system) from *local* behaviour expressed, at each state, by a particular specification.

The first requirement leads to the introduction of *nominals* as explicit references to specific states of the underlying transition system. Conceptually this exposes students to another basic and pervasive notion in Computer Science, that of *naming*. Hybrid logics [10] are the appropriate tool for this last stage in the course. The need for formulating specific *local* requirements, on the other hand, imposes extra structure upon states. Actually, different states are interpreted as different *modes* of operation and each of them is equipped with an algebraic specification of the corresponding functionality. Technically, specifications become *structured* state-machines, where states are specified as *algebras*, rather than as *sets*.

As mentioned in the previous section, HETS provides for free the proof support environment needed for this course. The boilerplates translator introduced in the previous section can also be used in the course to directly generate $\mathcal{H}$CASL specifications. Its pedagogical value, in training students to write specifications, is greatly appreciated. It should be stressed, however, that, in despite of the crucial role played by institution theory in this approach, no familiarity with institutions is required from students.

5 A Glimpse of a Course Session

The course contents and methodology are better understood through the presentation of a typical problem addressed first in the exercises class and later in the laboratory, in the last stage of the course. For space limitations we only focus on a fragment of the original problem. The example, small but self-contained, is taken from a description of requirements for an *automatic cruise control* (ACC) system summarised in [30] as follows:

> The mode class CruiseControl contains four modes, Off, Inactive, Cruise, and Override. At any given time, the system must be in one of these modes. Turning the ignition on causes the system to leave Off mode and enter Inactive mode, while turning the cruise control level to const when the brake is off and the engine running causes the system to enter Cruise mode. (…) Once cruise control has been invoked, the system uses the automobile's actual speed to determine whether to set the throttle to accelerate or decelerate the automobile, or to maintain the current speed (…)To override cruise control (i.e., enter Override), the driver turns the lever to off or applies the brake.

These requirements are captured by the state machine depicted in Fig. 5 and expressed in *hybrid propositional logic* ($\mathcal{H}PL$).

A modality *next* is introduced to denote the state-machine accessibility relation. Nominals in set $\{off, inactive, override, cruise\}$ correspond to the operation modes mentioned in the requirements. The first element students can formally capture within the logic is the transition structure, as in, for example,

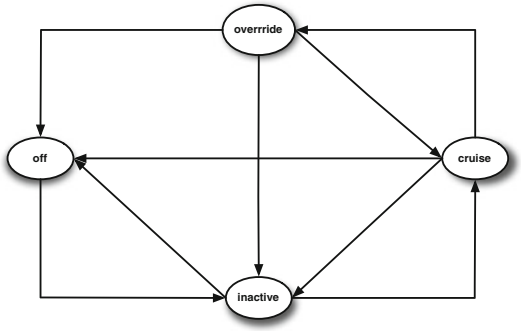


Fig. 5 The transition structure

- (T_1) $@_{off}\ \langle next\rangle\ inactive$
- (T_2) $@_{override}\ (\langle next\rangle\ off \wedge \langle next\rangle\ inactive \wedge \langle next\rangle\ cruise)$

Local properties can also be expressed through the satisfaction operator $@_i$, for each nominal i, to refer to the corresponding state. For instance, the requirement that the ignition is off when the system is in the *off* mode, while it is *on* and the engine running (*EngRunning*) in the *cruise* mode, is modelled by

- (L_1) $@_{off}(\neg IgnOn)$
- (L_2) $@_{cruise}(IgnOn \wedge EngRunning)$

Symbols *EngRunning* and *IgnOn*, with a self-explanatory designation, are propositions whose validity is discussed in each configuration (state). Others are used in the sequel. Definitional properties can also be captured, as in

- (A_1) $LeverOff \Leftrightarrow \neg\ LeverCons$
- (A_4) $HighSpeed \Rightarrow \neg\ CruiseSpeed \wedge \neg\ LowSpeed$

The second step in the case study is to equip each state of the underlying transition system with a first-order structure, to model its local functionality. Therefore, hybrid structures are enriched with a family of first-order structures indexed by the set of states, i.e., they become structures (M, W) where function M defines a family $(M_w)_{w\in|W|}$ of first-order structures over the same signature and universe (constraint necessary for the conservativity of the $\mathcal{H}FOL2FOL$ encoding). Each M_w models the system's behaviour at state $w \in W$. Note that at state w each first order formula is evaluated in the structure M_w. Properties are now expressed in a hybrid first order language $\mathcal{H}FOL$ whose detailed presentation we omit here (but see [35]). We focus instead on the sort of properties students are supposed to formulate. An algebraic specification is used to model system's functionality. This entails the need for introducing data types able to support the envisaged notions of *time*, *speed* and *acceleration*.

spec TIMESORT =INT
with sort *Int* $\mapsto$ *time*, **ops** $0 \mapsto$ *init*, *suc* $\mapsto$ *after* **end**
spec SPEEDSORT =INT **with sort** *Int* $\mapsto$ *speed* **end**
spec ACELLSORT =INT **with sort** *Int* $\mapsto$ *accel* **end**

Operation *Pedal* models the accelerations applied by the driver at each moment. On the other hand, *Automatic* captures accelerations applied on the engine by the ACC, and *CurrentSpeed* records the current speed. Finally, constant *MaxCruiseSpeed* represents the maximum speed allowed on the ACC mode:

spec ACCSIGN =
TIMESORT **and** SPEEDSORT **and** ACELLSORT
then ops $Pedal : time \to accel$;
$Automatic : time \to accel$;
$Speed : speed \times accel \to speed$;
$CurrentSpeed : time \to speed$;
$MaxCruiseSpeed : speed$

Students are asked to identify properties that globally hold, in all possible configurations, and the ones which model local requirements. In the first group we have, for example,

$\forall\ s : speed; a : accel; t : time$
• (G_1) $Speed(s, a) \geq 0$
• (G_2) $CurrentSpeed(t) = 0 \wedge Pedal(t) \geq 0 \Rightarrow$
$CurrentSpeed(after(t)) \geq 0$
• (G_3) $Pedal(t) > 0 \Leftrightarrow CurrentSpeed(t) < CurrentSpeed(after(t))$
• (G_4) $Speed(s, a) = s \Leftrightarrow a = 0$
• (G_5) $CurrentSpeed(after(t)) = Speed(CurrentSpeed(t), Pedal(t))$

Local properties refer to specific configurations. For example, in state *off*, *Speed* and *Pedal* are null and no other operation in the interface react. Thus,

$\forall\ t : time; s : speed; a : accel$
• (L^1_{off}) $@_{off}\, CurrentSpeed(t) = 0$
• (L^2_{off}) $@_{off}\, Speed(s, a) = 0$

On the other hand, in state *inactive*, speed and acceleration depend on the accelerations automatically introduced in the system, i.e.,

$\forall\ s : speed; a : accel$
• $(L^1_{inactive})$ $@_{inactive} Speed(s, a) = s + a$

$\forall\ t$: $time; s : speed; a : accel$
• $(L^{1'}_{cruise})$ $@_{cruise}[CurrentSpeed(t) > MaxCruiseSpeed \Rightarrow Automatic(after(t)) < 0]$
• $(L^{2'}_{cruise})$ $@_{cruise}[CurrentSpeed(t) \leq MaxCruiseSpeed \Leftrightarrow Automatic(after(t)) = 0]$
• (L^3_{cruise}) $@_{cruise} Speed(s, a) = s + a$
• (L^4_{cruise}) $@_{cruise} Pedal(t) \geq 0 \Rightarrow Pedal(t) = Automatic(t)$

An interesting feature in this example is that properties local to states *override* and *off* do coincide. The system's behaviour on both states only differs in what concerns the definition of the allowed transitions. Actually, students may now be invited to revisit the specification of the transition system presented above. It turns

out that some propositions may be re-stated by means of properties of local states. For instance,

$\forall\, t$: *time*;
• (L_1) $@_{cruise}[CurrentSpeed(t) = 0 \Rightarrow \langle next\rangle^u(inactive \wedge CurrentSpeed(after(t)) = 0)]$

where $\langle\lambda\rangle^u\rho$ abbreviates $\langle\lambda\rangle\rho \wedge [\lambda]\rho$.

Finally, in the laboratory session students are invited to translate hybrid to first order specifications and use HETS to animate them. On translating to $\mathcal{H}FOL2FOL$ we end up with the following signature (see Fig. 6):

ops
*Speed** : $st^* \times speed \times accel \rightarrow speed$;
*Pedal** : $st^* \times time \rightarrow accel$;...
pred
next : $st^* \times st^*$; *IgnOn** : st^*; ...

where global properties are universally quantified, and local properties take as an argument the respective nominal. For instance, global properties (G_1) and (G_2) are translated into

$\forall\, s$: *speed*; w : st^*; a : *accel*;t : *time*
• (G_{1^*}) $\geq^*(w$,*Speed**$(w, s, a), 0^*(w))$
• (G_{2^*}) *CurrentSpeed**$(w,t) = 0^*(w) \wedge \geq^*(w,$ *Pedal**$(w,t), 0^*(w))$.

and local properties (L^1_{off}) and (L^4_{cruise}), into

$\forall\, t$: *time*
• $(L^{1^*}_{off})$ *CurrentSpeed**$(off,t) = 0^*(off)$
• $(L^{4^*}_{cruise}) \geq^*$
$(cruise,$*Pedal**$(cruise,t),0^*(cruise)) \Rightarrow$ *Pedal*$(cruise,t) =$ *Automatic**$(cruise,t)$.

6 A Step Ahead: The Power of Quantification

6.1 Introducing Full Quantification

This section introduces a new, major extension to the method surveyed in the previous sections to support quantification. This requires the inclusion of another parameter in the method: a *quantification space*.[1] $\mathcal{D}^{\mathcal{HI}}$ for $\text{Mod}^{\mathcal{HI}}$.

In the institutional framework, as a subclass of $\text{Sign}^{\mathcal{HI}}$, quantification morphisms consist of triples $\chi = (\chi_{\text{Sig}}, \chi_{\text{Nom}}, \chi_{\text{MS}}) : (\Sigma, \text{Nom}, \Lambda) \rightarrow (\Sigma', \text{Nom}', \Lambda')$. Each of these components is responsible for a particular kind of quantification. We are particularly interested in inclusion morphisms, which are the ones that give rise to standard

[1] Quantification spaces are extensively discussed in Madeira's thesis [34], as well as in a joint paper with Diaconescu [23].

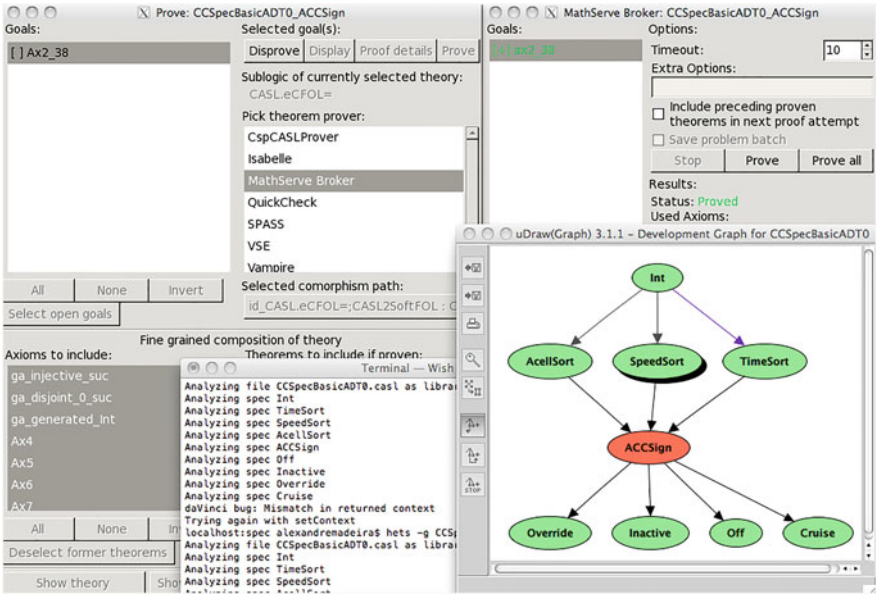


Fig. 6 A HETS session

quantifications. For example, considering $\chi_{\text{Nom}} : \text{Nom} \hookrightarrow \text{Nom} + Y$, for Y a finite set of constants, and considering χ_{MS} and χ_{Sig} the identity morphisms, we obtain the standard state quantification that can be found in the literature.

In the (fully) quantified version of the method proposed in this paper, the set $\text{Sen}^{\mathcal{HI}}(\Delta)$ is enriched with the sentence $(\forall\chi)\rho$, for any $\chi : \Delta \to \Delta' \in \mathcal{D}^{\mathcal{HI}}$ and $\rho \in \text{Sen}^{\mathcal{HI}}(\Delta')$. Similarly, the translation of sentences is extended, in each morphism $\varphi : \Delta \to \Delta_1$, by $\text{Sen}^{\mathcal{HI}}(\varphi)\big((\forall\chi)\rho\big) = (\forall\chi(\varphi))\text{Sen}^{\mathcal{HI}}(\varphi[\chi])(\rho)$. Finally, in what concerns the satisfaction relation, we consider

- $(M, W) \models^w (\forall\chi)\rho$ iff $(M', W') \models^w \rho$

for any (M', W') such that $\text{Mod}^{\mathcal{HI}}(\chi)(M', W') = (M, W)$. Existential quantification is introduced in a similar way.

In standard logical terminology, given an inclusion morphism

$$\chi = (\chi_{\text{Sig}}, \chi_{\text{Nom}}, \chi_{\text{MS}}) : (\Sigma, \text{Nom}, \Lambda) \to (\Sigma', \text{Nom}', \Lambda')$$

where $\chi_{\text{Nom}} : \text{Nom} \hookrightarrow \text{Nom} + U$ and $\chi_{\text{MS}} : \Lambda \hookrightarrow \Lambda + Y$, for finite sets $U = \{u_1, u_2, \ldots, u_n\}$ and $Y = \{y_1, y_2, \ldots, y_m\}$, the new sentence $(\forall\chi)\rho$ may be written as $\forall_{u_1,u_2,\ldots,u_n} \forall_{y_1,y_2,\ldots,y_m} \rho$. Moreover, one can say that $(M, W) \models^w (\forall\theta)\rho$ iff, for any θ-expansion $(M, W)^\theta$ of (M, W), one has $(M, W)^\theta, w \models \rho$.

Quantified sentences play a major role in specification theory. Actually,

- Quantification coming from the base institution can be used to specify local configurations.
- Quantification over nominals, makes possible to express properties about the system's global state space. This is particularly useful, for instance, to express the existence of configurations satisfying a given requirement.
- Quantification over modalities, finally, constitutes a rather powerful form of quantification useful to express enabling/disabling of reconfigurations.

The last two types of quantification are explored below as a very general way to introduce dynamic modalities. Specifically, quantification over nominals and over modalities makes possible to express paradigmatic changes on the relational model, like *swapping* and *sabotage*. This is done at minimal cost and in a very general way which captures several approaches in the literature which are specific to particular situations.

6.2 Effects and Dynamic Modalities

Suppose you take a train and start planning your trip as you go. With a proper map the task is quite straightforward. But *what if the transportation system breaks down, and a malevolent demon starts canceling connections, anywhere in the network?* This question appears in the motivation section of van Benthem seminal paper on sabotage logic [54]. The scenario is as follows: there is a transition structure (the map, a graph) over which sentences are interpreted as usual in modal logic; however this may change dynamically while being traversed.

Sabotage logic is an example of a modal logic equipped with modalities that can change the accessibility relation of the underlying Kripke model along the evaluation of a formula. In particular, edges are deleted. Adding new edges or swapping existent ones are further examples of effects leading to logics which, over time, have found interesting applications in describing and reasoning about dynamic aspects of phenomena. Some recent papers [1–3] explore specific instances of these ideas further witnessing their relevance to application areas ranging from reconfigurable software specifications to changing obligations contexts in epistemic logics. In these logics the meaning of the basic modal operators remains unchanged, but new ones, suitably called *dynamic modalities*, are introduced to encode specific changes in the accessibility relation.

Our approach aims at going a step forward. Instead of formulating new, tailor-made logics for each family of effects, we resort to the fully quantified hibridisation of the Triv institution, in which the typical dynamic modalities in the literature can be captured in a uniform way and within a unique logic. The introduction of quantification over modality symbols allows not only a suitable encoding of effects, like reversing or deleting transitions, but also the precise specification of their scope (e.g., the whole or part of the accessibility relation) and the point of application (e.g., anywhere, relative to the current evaluation point, an edge between specific named

states, etc.). This goes beyond and generalises current approaches in the literature. The only work we are aware of with a similar spirit, but through a different way, is a very recent paper by Areces et al. [4] which proposes a characterisation of what the authors call *relation-changing modal operators*. Actually, our approach differs from the one above, by the ability to express a bigger diversity of effects. The reason is that we resort to an abstract hybrid logic and, through nominals, it is possible to express changes in specific points of the relational structure.

Besides providing a uniform setting to discuss dynamic modalities, and, more generally, *effects* over Kripke models, the main advantage of the approach introduced here is the possibility to characterise typical results in the study of these logics in a generic way, for example a general notion of bisimulation parametric on the effect. Finally note that, in the approach proposed here, and contrary to what appears in the literature, models remain standard Kripke structures, no actual updating taking place in the accessibility relation. The effect of dynamic modalities is to expand the original relation into a new, updated one and, then, to hand it over the current evaluation point.

6.2.1 Effects and Events

An *effect* $E(X, Y, x, y)$ captures a specific transformation, or update, of an accessibility relation X in a Kripke model. It can be regarded as a *macro* relating two accessibility relations X and Y. For example the *swap effect*, which inverts in Y the orientation of an edge in X, is specified as

$$\textbf{(Swap)} \qquad Sw(X, Y, x, y) \stackrel{abv}{=} @_x\langle X\rangle y \wedge @_y\langle Y\rangle x$$

The *sabotage* effect, which ignores in Y the edge (x, y) of X, is given by

$$\textbf{(Sabotage)} \qquad Sg(X, Y, x, y) \stackrel{abv}{=} @_x\langle X\rangle y \wedge \neg @_x\langle Y\rangle y$$

Enriching X with a specific new edge, is expressed through the *bridge* effect:

$$\textbf{(Bridge)} \qquad Bg(X, Y, x, y) \stackrel{abv}{=} \neg @_x\langle X\rangle y \wedge @_x\langle Y\rangle y$$

Weaker forms of the two latter effects can also be considered:

$$\textbf{(Conditional Sabotage)} \qquad PSg(X, Y, x, y) \stackrel{abv}{=} @_x\langle X\rangle y \rightarrow \neg @_x\langle Y\rangle y$$

$$\textbf{(Conditional Bridge)} \qquad PBg(X, Y, x, y) \stackrel{abv}{=} \neg @_x\langle X\rangle y \rightarrow @_x\langle Y\rangle y$$

An effect can act upon a given, specific edge (x, y), or a set of edges. This is called the range (**rng**) of an effect—*exclusive* (denoted by **o**) or *partial* (**p**). Once this

specified for a particular effect, the resulting expression is called an *event*. Formally, given an effect E, an E-event $E_{\mathbf{rng}}(X, Y, x, y)$ with $\mathbf{rng} \in \{\mathbf{p}, \mathbf{o}\}$ is a sentence in $\mathcal{H}Triv$ such that

- $E_{\mathbf{p}}(X, Y, x, y) \stackrel{abv}{=} E(X, Y, x, y) \wedge Ex_E(X, Y, x, y)$
- $E_{\mathbf{o}}(X, Y, x, y) \stackrel{abv}{=} E(X, Y, x, y) \wedge U(X, Y, x, y)$

where

- $Ex_E(X, Y, x, y) \stackrel{\text{def}}{=} (\forall s, v)\big((@_s\langle X\rangle v \leftrightarrow @_s\langle Y\rangle v) \vee (E(X, Y, s, v) \wedge @_s x)\big)$
- $U(X, Y, x, y) \stackrel{\text{def}}{=} (\forall s, v)\big((@_s\langle X\rangle v \leftrightarrow @_s\langle Y\rangle v) \vee (@_s x \wedge @_v y)\big)$

Intuitively, expression $Ex_E(X, Y, x, y)$ asserts that an edge with source in x can only be updated, on going from X to Y, as result of effect E. Apart from this, relations X and Y remain equal. Expression $U(X, Y, x, y)$, on the other hand, establishes that any modification affects exclusively the pair of states x and y.

Let us illustrate this construction with the event **o**-*swap* for edge (x, y):

$$\begin{aligned} Sw_{\mathbf{o}}(X, Y, x, y) &\stackrel{\text{def}}{=} Sw(X, Y, x, y) \wedge U(X, Y, x, y) \\ &= \big(@_x\langle X\rangle y \wedge @_y\langle Y\rangle x\big) \wedge (\forall s, v)\big(@_s\langle X\rangle v \leftrightarrow @_s\langle Y\rangle v \ \vee (@_s x \wedge @_v y)\big) \end{aligned}$$

where relation Y is constructed by swapping exactly the edge (x, y) of X. The partial range version of this event, **p**-*swap*, is

$$\begin{aligned} Sw_{\mathbf{p}}(X, Y, x, y) &\stackrel{abv}{=} Sw(X, Y, x, y) \wedge Ex_{Sw}(X, Y, x, y) \\ &= \big(@_x\langle X\rangle y \wedge @_y\langle Y\rangle x\big) \wedge \\ &\quad (\forall s, v)\big((@_s\langle X\rangle v \leftrightarrow @_s\langle Y\rangle v) \vee (Sw(X, Y, s, v) \wedge @_s x)\big) \\ &= \big(@_x\langle X\rangle y \wedge @_y\langle Y\rangle x\big) \wedge \\ &\quad (\forall s, v)\big((@_s\langle X\rangle v \leftrightarrow @_s\langle Y\rangle v) \vee ((@_s\langle X\rangle v \wedge @_v\langle X\rangle s) \wedge @_s x)\big) \end{aligned}$$

As expected, the new accessibility relation Y is identical to X, but on a number of swapped edges with source in x. The result of a partial *swap* and a partial *sabotage* event is depicted in Figs. 8 and 9, respectively (over the same relation X depicted in Fig. 7).

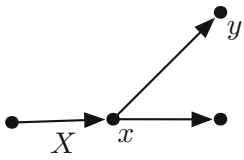


Fig. 7 The original relation X

Fig. 8 *X* swapped

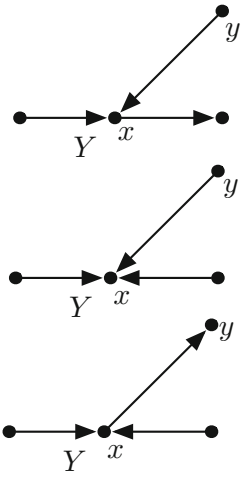
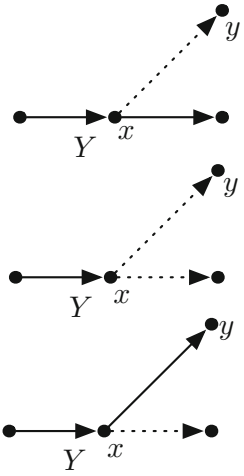


Fig. 9 *X* sabotaged



6.2.2 Dynamic Modalities

Dynamic modalities are built from the *events* introduced in the previous section. Please note that there is no actual update of the accessibility relation. A dynamic modality expands the original model with a new, modified relation with reference to which evaluation proceeds. For any event $E_{\mathbf{rng}}(X, Y, x, y)$, two dynamic modalities are defined: a *local* and a *global* modality. The first one is defined by

$$\textbf{(Local)} \quad \ll E_{\mathbf{rng}}(X) \gg_l \rho \stackrel{\text{def}}{=} (\exists Y, x, y)\big(x \wedge E_{\mathbf{rng}}(X, Y, x, y) \wedge @_y \rho_X^Y\big)$$

where Y, x, y are variables not occurring in ρ.

The intuition is that event E is performed in possible edges whose source is the current evaluation point, which then changes through a transition over an updated edge. The global modality, on the other hand, is defined by

$$\textbf{(Global)} \quad \ll E_{\mathbf{rng}}(X) \gg_g \rho \stackrel{\text{def}}{=} (\exists Y, x, y)\big(E_{\mathbf{rng}}(X, Y, x, y) \wedge \rho_X^Y\big)$$

where Y, x, y are variables not occurring in ρ. In this case the event is performed at some point in the model and the current evaluation point does not change. Observe that substitution ρ_X^Y represents the "shift" between the original relation X by the "updated" one Y.

As usual, corresponding boxed dynamic modalities are obtained through

$$[[E_{\mathbf{rng}}(X)]]_l \rho \stackrel{\text{def}}{=} (\forall Y, x, y)\big((x \wedge E_{\mathbf{rng}}(X, Y, x, y)) \rightarrow @_y \rho_X^Y\big)$$
$$[[E_{\mathbf{rng}}(X)]]_g \rho \stackrel{\text{def}}{=} (\forall Y, x, y)\big(E_{\mathbf{rng}}(X, Y, x, y) \rightarrow \rho_X^Y\big)$$

where Y, x, y are variables not occurring in ρ and ρ_X^Y is the sentence obtained by substituting all the occurrences of X by Y. As expected, for any formula $\rho \in Fm(\text{Nom}, \text{Prop}, \Lambda)$, correspondences

$$\neg \ll E_{\mathbf{rng}}(X) \gg_l \neg\rho \leftrightarrow [[E_{\mathbf{rng}}(X)]]_l \rho$$

and

$$\neg \ll E_{\mathbf{rng}}(X) \gg_g \neg\rho \leftrightarrow [[E_{\mathbf{rng}}(X)]]_g \rho$$

hold.

7 Concluding

The hybridisation method discussed in this paper can be broadly understood as a specific way of combining logics at the model theoretical level. Actually, it classifies as *a tool for simplifying problems involving heterogeneous reasoning*, a common ingredient to this family of methods according to the corresponding entry in the *Stanford Encyclopedia of Philosophy* [16]. The same entry stresses the role of Computer Science applications as a main driving force for research in obtaining new logic systems from old: *One of the main areas interested in the methods for combining logics is software specification. Certain techniques for combining logics were developed almost exclusively with the aim of applying them to this area*. [16].

More specifically, hybridisation is a form of asymmetric combination of logics in the sense that specific features of hybrid logic are developed "on top" of another logic. This follows the pattern of, and to a certain extent extends, previous work by Diaconescu and Stefaneas [24] on "modalisation" of institutions, which endows systematically institutions with Kripke semantics for standard modalities. The insti-

tutional setting [7] in which we worked offers a suitable framework to discuss the generation of new logics from old, and to identify the sort of properties preserved or reflected along such a process. As in many other areas of theoretical Computer Science, going categorial means going generic.

In the following paragraphs we briefly discuss some directions for future work. The first is concerned with the extension of the educational application of the hybridisation method described above. The other two are specific research challenges on pushing forward the method reviewed in this paper.

A curricular challenge. Sects. 4 and 5 introduced the *rationale* for a somehow not very standard introductory course to software specification with hybrid(ised) logics. Building on an institution-based framework kept implicit along the lectures, the course aims at conducting students through two orthogonal paradigms (equational and hybrid) which are then combined in a common specification framework.

The approach underlying the course is based on a particular instance of the hybridisation method. However, other possible "hybridisations" (e.g., of institutions of multialgebras or partial algebras) are suitable to explore a wide range of exercises in a similar spirit. Moreover, the course skills may be easily expanded into new directions: for instance, functional and imperative programming languages may be presented as institutions (see [52]) whose hybridisation may be used to develop reconfigurable algorithms. On a different note, a two-level hybridisation of a base logic, as discussed in [34], provides modalities and nominals at two different levels: local and global. This seems a suitable setting to talk about reconfigurable software applications whose local configurations are also described by transition systems. More generally, models become *hierarchical* transition systems. In [44], the authors have also presented the logic underlying ALLOY [32] in an institutional setting. This paves the way to hybridising ALLOY and combining in the course the use of the traditional ALLOY model finder with theorem proving (in HETS) in an integrated way.

Beyond reconfigurability, hybridised logics may provide flexible frameworks to address related problems in software design, namely those concerning adaptation and software evolution.

Hybridisation for quantitative reasoning. Specification frameworks for *quantitative reasoning*, dealing for example with weighted or probabilistic transition systems, emerged recently as a main challenge for software engineers. This witnesses a shift from classical models of computation, such as labeled transition systems, to similar structures where quantities can be handled. Examples include weighted [19], hybrid [28, 33] or probabilistic [49] automata, as well as their coalgebraic rendering (e.g., [51]). An interesting topic to pursue is taking up this "quantitative" challenge within the context of the hybridisation process itself. The simplest move in such a direction proceeds by instantiation. In this case quantitative reasoning is just reflected and expressed at the *local* level of concrete, specific configurations. A complementary path may focus on generalising the underlying semantic structures, replacing the *REL*-component in models by coalgebras over suitable categories of probability distributions, metric, or topological spaces.

Calculus. Comparing the calculus for hybrid propositional logic in Ref. [14] with the one for hybrid first-order logic in [13], a common structure pops out: both "share" rules involving sentences with nominals and satisfaction operators (i.e., formulas of a "hybrid nature") and have specific rules to reason about "atomic sentences" that come from the base institution. Hence, it makes sense to consider the development of a general proof calculus for hybrid institutions on top of the calculus of the corresponding base institution, in the style of [12, 17]. Somehow anticipating the general construction, a calculus for equational hybrid logic was proposed in [11].

Recent work [45] reports preliminary general results in this direction. In particular, it is shown that, whenever the base logic has the usual Boolean connectives, hybridisation preserves decidability, and furthermore, the generated calculus is sound and complete whenever the one for the base logic is. These results have not only a theoretical interest on their own, but also pave the way for new approaches to tool supported verification.

Acknowledgments This work is financed by the ERDF—European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation—COMPETE 2020 Programme, and by National Funds through the FCT (Portuguese Foundation for Science and Technology) within project POCI-01-0145-FEDER-006961. M. Martins was further supported by project UID/MAT/04106/2013. A. Madeira and R. Neves research was carried out in the context of a post-doc and a Ph.D. grant with references SFRH/BPD/103004/2014 and SFRH/BD/52234/2013, respectively. L.S. Barbosa is also supported by SFRH/BSAB/113890/2015.

References

1. Areces, C., Fervari, R., Hoffmann, G.: Moving arrows and four model checking results. In: Ong, L., de Queiroz, R. (eds.) Proceedings of the 19th International Workshop on Logic. Language, Information and Computation (WoLLIC 2012). Lecture Notes in Computer Science, vol. 7456, pp. 142–153. Springer, Buenos Aires, Argentina (2012)
2. Areces, C., Fervari, R., Hoffmann, G.: Tableaux for relation-changing modal logics. In: Proceedings of Frontiers of Combining Systems 2013, Nancy, France, Sept 2013
3. Areces, C., Fervari, R., Hoffmann, G.: Swap logic. Logic J. IGPL **22**(2), 309–332 (2014)
4. Areces, C., Fervari, R., Hoffmann, G.: Relation-changing modal operators. Logic J. IGPL **23**(4), 601–627 (2015)
5. Areces, C., ten Cate, B.: Hybrid logics. In: Blackburn, P., Wolter, F., van Benthem, J. (eds.) Handbook of Modal Logic. Studies in Logic and Practical Reasoning, vol. 3, pp. 822–868. Elsevier (2007)
6. Burstall, R., Diaconescu, R.: Hiding and behaviour: an institutional approach. In: Roscoe, W. (ed.) A Classical Mind: Essays in Honour of C.A.R. Hoare, pp. 75–92. Prentice-Hall (1994)
7. Burstall, R.M., Goguen, J.A.: The semantics of CLEAR, a specification language. In: Bjørner, D. (ed.) Abstract Software Specifications (1979 Copenhagen Winter School, 22 Jan–2 Feb 1979), Lecture Notes in Computer Science, vol. 86, pp. 292–332. Springer (1980)
8. Bidoit, M., Hennicker, R.: Constructor-based observational logic. J. Log. Algebr. Program. **67**(1–2), 3–51 (2006)
9. Beierle, C., Kern-Isberner, G.: Looking at probabilistic conditionals from an institutional point of view. In: Kern-Isberner, G., Rödder, W., Kulmann, F. (eds.) Conditionals, Information, and

Inference (Revised Selected Papers of WCII 2002, Hagen, Germany, 13–15 May 2002), Lecture Notes in Computer Science, vol. 3301, pp. 162–179. Springer (2005)
10. Blackburn, P.: Representation, reasoning, and relational structures: a hybrid logic manifesto. Logic J. IGPL **8**(3), 339–365 (2000)
11. Barbosa, L.S., Martins, M.A., Carreteiro, M.: A Hilbert-style axiomatisation for equational hybrid logic. J. Logic Lang. Inf. **23**(1), 31–52 (2014)
12. Borzyszkowski, T.: Logical systems for structured specifications. Theor. Comput. Sci. **286**(2), 197–245 (2002)
13. Braüner, T.: Natural deduction for first-order hybrid logic. J. Logic Lang. Inf. **14**(2), 173–198 (2005)
14. Braüner, T.: Hybrid Logic and Its Proof-Theory. Applied Logic Series. Springer (2010)
15. Cîrstea, C.: An institution of modal logics for coalgebras. J. Log. Algebr. Program. **67**(1–2), 87–113 (2006)
16. Carnielli, W., Coniglio, M.E.: Combining logics. In: Zalta, E.N. (ed.) *The Stanford Encyclopedia of Philosophy*. Winter 2011 edn. (2011)
17. Codescu, M., Găină, D.: Birkhoff completeness in institutions. Logica Universalis **2**(2), 277–309 (2008)
18. Caleiro, C., Mateus, P., Sernadas, A., Sernadas, C.: Quantum institutions. In: Futatsugi, K., Jouannaud, J.-P., Meseguer, J. (eds.) Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday. Lecture Notes in Computer Science, vol. 4060, pp. 50–64. Springer (2006)
19. Droste, M., Gastin, P.: Weighted automata and weighted logics. Theor. Comput. Sci. **380**(1–2), 69–86 (2007)
20. Diaconescu, Răzvan: Institution-independent Model Theory. Studies in Universal Logic. Birkhäuser Basel (2008)
21. Diaconescu, R.: On quasi-varieties of multiple valued logic models. Math. Log. Q. **57**(2), 194–203 (2011)
22. Diaconescu, R.: Quasi-varieties and initial semantics in hybridized institutions. J. Logic Comput. (2015)
23. Diaconescu, R., Madeira, A.: Encoding hybridized institutions into first-order logic. Math. Struct. Comput. Sci. 1–44 (2015) (in print)
24. Diaconescu, R., Stefaneas, P.S.: Ultraproducts and possible worlds semantics in institutions. Theor. Comput. Sci. **379**(1–2), 210–230 (2007)
25. Goguen, J.A., Burstall, R.M.: Institutions: abstract model theory for specification and programming. J. ACM **39**(1), 95–146 (1992)
26. Gottwald, S.: A Treatise on Many-Valued Logics. Studies in Logic and Computation vol. 9. Research Studies Press (2001)
27. Goguen, J.A., Roşu, G.: Institution morphisms. Formal Asp. Comput. **13**(3–5), 274–307 (2002)
28. Henzinger, T.A.: The theory of hybrid automata. In: 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96, New Brunswick, New Jersey, USA, 27–30 July 1996), pp. 278–292 (1996)
29. Hull, M.E.C., Jackson, K., Dick, J.: Requirements Engineering, 2nd edn. Springer Verlag (2005)
30. Heitmeyer, C.L., Kirby, J., Labaw, B.G.: The SCR method for formally specifying, verifying, and validating requirements: tool support. In: Richards Adrion, W., Fuggetta, A., Taylor, R.N., Wasserman, A.I. (eds.) Pulling Together, Proceedings of the 19th International Conference on Software Engineering, Boston, Massachusetts, USA, 17–23 May 1997, pp. 610–611. ACM (1997)
31. Indrzejczak, A.: Modal hybrid logic. Logic Logical Philos. **16**, 147–257 (2007)
32. Jackson, D.: Software Abstractions (Logic, Language, and Analysis), 2nd edn. MIT Press (2011)
33. Lynch, N.A., Segala, R., Vaandrager, F.W., Weinberg, H.B.: Hybrid i/o automata. In: Alur, R., Henzinger, T.A., Sontag, E.D. (eds.) Hybrid Systems III: Verification and Control (DIMACS/SYCON Workshop, 22–25 Oct 1995, Ruttgers University, New Brunswick, NJ, USA). Lecture Notes in Computer Science, vol. 1066, pp. 496–510. Springer (1995)

34. Madeira, A.: Foundations and techniques for software reconfigurability. Ph.D. thesis, Universidades do Minho, Aveiro and Porto (Joint MAP-i Doctoral Programme), July 2013
35. Madeira, A., Faria, J.M., Martins, M.A., Barbosa, L.S.: Hybrid specification of reactive systems: an institutional approach. In: Barthe, G., Pardo, A., Schneider, G. (eds.) Software Engineering and Formal Methods (SEFM 2011, Montevideo, Uruguay, 14–18 Nov 2011). Lecture Notes in Computer Science, vol. 7041, pp. 269–285. Springer (2011)
36. Mossakowski, T., Haxthausen, A., Sannella, D., Tarlecki, A.: CASL: the common algebraic specification language: semantics and proof theory. Comput. Inf. **22**, 285–321 (2003)
37. Madeira, A., Martins, M.A., Barbosa, L.S.: Boilerplates for reconfigurable systems: a language and its semantics. In: Du Bois, A.R., Trinder, P. (eds.) Programming Languages—17th Brazilian Symposium, SBLP 2013, Brasília, Brazil, 3–4 Oct 2013. Proceedings. Lecture Notes in Computer Science, vol. 8129, pp. 75–89. Springer (2013)
38. Martins, M.A., Madeira, A., Barbosa, L.S., Neves, R.: Paradigm integration in a specification course. In: Joshi, J., Bertino, E., Thuraisingham, B.M., Liu, L. (eds.) Proceedings of 15th IEEE International Conference on Information Reuse and Intergration, IRI 2014, Redwood City, CA, USA, 13–15 Aug 2014, pp. 492–499. IEEE Press (2014)
39. Martins, M.A., Madeira, A., Diaconescu, R., Barbosa, L.S.: Hybridization of institutions. In: Corradini, A., Klin, B., Cîrstea, C. (eds.) Algebra and Coalgebra in Computer Science (CALCO 2011, Winchester, UK, 30 Aug–2 Sept 2011). Lecture Notes in Computer Science, vol. 6859, pp. 283–297. Springer (2011)
40. Mossakowski, T., Maeder, C., Lüttich, K.: The heterogeneous tool set, Hets. In: Grumberg, O., Huth, M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2007—Braga, Portugal, 24 Mar—1 Apr 2007). Lecture Notes in Computer Science, vol. 4424, pp. 519–522. Springer (2007)
41. Mossakowski, T.: Different types of arrow between logical frameworks. In: auf der Heide, F.M., Monien, B. (eds.) Automata, Languages and Programming (ICALP96, Paderborn, Germany, 8–12 July 1996). Lecture Notes in Computer Science, vol. 1099, pp. 158–169. Springer (1996)
42. Mossakowski, T., Roggenbach, M.: Structured CSP—a process algebra as an institution. In: Fiadeiro, J.L., Schobbens, P.-Y. (eds.) Recent Trends in Algebraic Development Techniques (Revised Selected Papers of WADT 2006, La Roche en Ardenne, Belgium, 1–3 June 2006). Lecture Notes in Computer Science, vol. 4409, pp. 92–110. Springer (2006)
43. Neves, R., Madeira, A., Martins, M.A., Barbosa, L.S.: Hybridisation at work. In: Heckel, R., Milius, S. (eds.) Algebra and Coalgebra in Computer Science—5th International Conference, CALCO 2013, Warsaw, Poland, 3–6 Sept 2013. Proceedings. Lecture Notes in Computer Science, vol. 8089, pp. 340–345 (2013)
44. Neves, R., Madeira, A., Martins, M.A., Barbosa, L.S.: An institution for alloy and its translation to second-order logic. In: Bouabana-Tebibel, T., Rubin, S.H. (eds.) Integration of Reusable Systems [extended versions of the best papers presented at IEEE International Conference on Information Reuse and Integration and IEEE International Workshop on Formal Methods Integration, San Francisco, CA, USA, Aug 2013]. Advances in Intelligent Systems and Computing, vol. 263, pp. 45–75. Springer (2013)
45. Neves, R., Madeira, A., Martins, M.A., Barbosa, L.S.: Completeness and decidability results for hybrid(ised) logics. In: Braga, C., Martí-Oliet, N. (eds.) Formal Methods: Foundations and Applications - 17th Brazilian Symposium, SBMF 2014, Maceió, AL, Brazil, 29 Sept–1 Oct 2014. Lecture Notes in Computer Science, vol. 8941, pp. 146–161. Springer (2015)
46. Prior, A.N.: Past, Present and Future. Oxford University Press (1967)
47. Passy, S., Tinchev, T.: An essay in combinatory dynamic logic. Inf. Comput. **93**(2), 263–332 (1991)
48. Ciocarlie, H., Szepesia, R.: An overview on software reconfiguration. Theory Appl. Math. Comput. Sci. **1**, 74–79 (2011)
49. Segala, R.: A compositional trace-based semantics for probabilistic automata. In: Lee, I., Smolka, S.A. (eds.) Concurrency Theory (CONCUR'95—Philadelphia, PA, USA, 21–24 Aug 1995). Lecture Notes in Computer Science, vol. 962, pp. 234–248. Springer (1995)

50. Schröder, L., Mossakowski, T.: HasCasl: integrated higher-order specification and program development. Theor. Comput. Sci. **410**(12–13), 1217–1260 (2009)
51. Sokolova, A.: Probabilistic systems coalgebraically: a survey. Theor. Comput. Sci. **412**(38), 5095–5110 (2011)
52. Sannella, D., Tarlecki, A.: Foundations of Algebraic Specification and Formal Software Development. Monographs on Theoretical Computer Science, an EATCS Series. Springer (2012)
53. van Bentham, J.: Modal Logic and Classic Logic. Humanities Press (1983)
54. van Benthem, J.: An essay on sabotage and obstruction. In: Hutter, D., Stephan, W. (eds.) Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday. Lecture Notes in Computer Science, vol. 2605, pp. 268–276. Springer (2005)

Test Reactive Systems with Büchi-Automaton-Based Temporal Requirements

Bolong Zeng and Li Tan

Abstract A reactive system is expected to interact with its environment constantly, and its executions may be modeled as infinite words. To capture temporal requirements for a reactive system, Büchi automaton has been used as a formalism to model and specify temporal patterns of infinite executions of the system. A key feature of a Büchi automaton is its ability of accepting infinite words through its acceptance condition. In this paper, we propose a specification-based technique that tests a reactive system with respect to its requirements in Büchi automaton. Our technique selects test suites based on their relevancy to the acceptance condition of a Büchi automaton. By focusing the testing efforts on this key element of a Büchi automaton that is responsible for accepting infinite words, we are able to build a testing process driven by the Büchi automaton specified temporal properties of a reactive system. At the core of our approach are new coverage metrics for measuring how well a test suite covers the acceptance condition of a Büchi automaton. We propose both weak and strong variants of coverage metrics for applications that need tests of different strengths. Each variant incorporates a model-checking-assisted algorithm that automates test case generation. Furthermore our testing technique is capable of revealing not only bugs in a system, but also problems in its requirements. By collecting and analyzing the information produced by a model-checking-assisted test case generation algorithm, our approach may identify inadequate requirements. We also propose an algorithm that refines a requirement in Büchi automaton. Finally, we conduct a thorough computational study to evaluate the performance of our proposed criteria using cross-coverage comparison and fault sensitivity analysis. The results validate the strength of our approach on improving the effectiveness and efficiency of testing, with test cases generated specifically for temporal requirements.

B. Zeng · L. Tan (✉)
School of Electrical Engineering and Computer Science, Washington State University, Richland, WA 99354, USA
e-mail: litan@wsu.edu

B. Zeng
e-mail: bzeng@wsu.edu

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_2

1 Introduction

Reactive systems refer to systems that constantly interact with their environments. Many of high dependable systems are reactive systems: they are expected to interact with their environments constantly (e.g. users, physical objects, etc.) even under adversary conditions. Typical safety sensitive systems such as air traffic control systems and nuclear reactor control systems all fall under the definition of reactive systems [26]. As the correct functioning of reactive systems are highly critical, engineers often deploy a mixture of Verification and Validation (V&V) techniques to ensure their correctness. Two of the most frequently used V&V techniques for reactive systems are testing and formal verification.

Testing examines a system by monitoring its behaviors under a fixed set of stimuli, and determines if the system behaves as expected. Based on a "trial and error" ideology, testing has been an essential part of many software V&V processes. In comparison, formal verification refers to a variety of techniques that establish the correctness of a system with respect to a formal requirement, by establishing a mathematically sound proof. Formal verification techniques, particularly model checking, have received much attention from research community, and they are being adopted by the industry as a powerful tool to verify designs of safety-critical systems [15] with an increasing presence.

Testing and formal verification are two complementary V&V techniques, each of which has its own set of pros and cons. Compared with formal verification, testing in general is more feasible in practice, and it may be applied to both specification and implementation. Nevertheless, a major drawback of testing, as notably noted by Dijkstra is that testing can only show the existence of a bug, but not its absence [2]. In contrast, formal verification techniques such as model checking build a mathematically sound proof for the correctness of a design. Nevertheless, formal verification falls short when it is unable to find a conclusive proof. In addition, it does not scale nearly as well as testing, limiting its application to designs or models extracted from implemented systems, such as in the case of software model checking.

A research theme in V&V field is how to harness the synergy of testing and formal verification. Techniques such as model-checking-based test case generation [11] have been proposed to utilize such a synergy. In this paper we are interested in specification-based testing with temporal requirements. In recent years, formal verification techniques, particularly model checking, have become significantly more popular in industrial applications (c.f. [19]). One of the consequences of the proliferation of formal verification techniques is that the formal requirements are also becoming more available. We want to take this opportunity and extend the application of formal requirements into testing. Our objectives are two-fold: (1) improve the effectiveness of testing by centering it around formal requirements; and (2) improve the efficiency of testing by developing a model-checking-assisted test case generator for our proposed test criteria.

In this paper, we extend our original work presented in [30] from both the theoretical and practical aspects. We choose reactive systems as the subject of our study, and focus on specification-based testing for reactive systems whose formal requirements are defined in Büchi automaton. The practical importance of reactive systems in developing safety-critical systems justifies the potential of our work. The essence of a reactive system is its ability of performing infinite executions. A challenge in testing a reactive system is how to specify and test these infinite behaviors. Many of previous works on testing reactive systems (c.f. [18, 21, 24]) have been focusing on testing finite prefixes of infinite executions. We want to develop a testing technique focusing on features of infinite executions themselves, that is, temporal patterns exhibited by an infinite execution of a reactive system. Particularly, we consider requirements encoded in Büchi automaton. Büchi automaton, a type of ω-automata that accept infinite words, has been widely used to specify linear temporal behaviors of a reactive system. It is also instrumental in developing linear temporal model checkers: other formalisms such as Linear Temporal Logic (LTL) are often first translated into Büchi automaton, before being used in model checking.

A first-order question in specification-based testing, or any software testing for that matter, is to measure the adequacy of a test suite. We define two coverage metrics measuring how a test suite covers a requirement in Büchi automaton. The metrics define how *thoroughly* a test suite covers the acceptance condition of a Büchi automaton. A Büchi automaton differs from a finite automaton in its acceptance condition, which enables the Büchi automaton to accept infinite words. By focusing on the acceptance condition of a Büchi automaton encoding the requirement for a reactive system, our approach centralizes testing efforts on infinite executions of the system. Test criteria derived from these metrics can then be used in producing and executing test suites.

To improve the efficiency of test generation, we also propose model-checking-assisted test case generation algorithms for proposed test criteria. By utilizing the counterexample producing capability of an off-the-shelf model checker, these algorithms automate the test case generation for reactive systems with Büchi automata. We also establish the correctness of said algorithms through mathematical proofs.

By deriving test cases from a reactive system model and its formal requirement, our specification-based testing approach becomes a powerful tool to detect the discrepancy between the model and its requirements. In addition to debugging a reactive system, our approach may also help in detecting the deficiency of a specification. The latter also enables the refinement of a requirement, reducing the gap between the semantics of the requirement and a system implementation. We propose a technique that automates the property-refinement process, by reusing the information from model-checking-assisted test generation algorithms.

We conduct two sets of computational study to compare the effectiveness of the proposed acceptance condition coverage criteria against other existing criteria, including traditional test criteria such as branch coverage, as well as the other LTL or Büchi automaton based test criteria introduced in [21, 23, 31]. The first set of

computational study is to measure the cross coverage of different test criteria, that is, how well a test suite generated for one test criterion covers another criterion. The results provide a measurement of the effectiveness of a test criterion in comparison with another criterion. The second set of computational study uses fault-injection technique. Fault-injection technique is a classic method for examining the coverage of a test suite [1]. In this study, we take a test suite generated from a specific test criterion, and use fault-injection technique to measure its ability to spot artificially planted errors. The measurement is an indicator for the effectiveness of the underlying test criterion. Both sets of computational study demonstrate the effectiveness and efficiency of our proposed approach. For these experiments, we select sample applications from a diversified range of fields, including software engineering (GIOP protocol for middleware construction), security (Needham-Schroeder public key protocol), and automobile (a fuel system example).

The rest of the paper is organized as follows: Sect. 2 prepares the notations used in the rest of the paper; Sect. 3 introduces two variants of accepting state combination coverage metrics and criteria for Büchi automata; Sect. 4 describes the model-checking-assisted test case generation algorithms for the proposed criteria; Sect. 5 discusses the requirement refinement using the feedback from the model-checking-assisted test case generation; Sect. 6 discusses the result of our computational study on the performance comparison between the new criteria and other existing test criteria; and finally Sect. 7 concludes the paper.

[Related Works] An important component of our approach is a model-checking-assisted algorithm that utilizes the counterexample mechanism of an off-the-shelf model checker to generate test cases. Model checkers are able to generate counterexamples of a model that violates a temporal formula that describes a desired property. Taking advantage of such ability of model checkers to assist test generation has received a significant amount of attention in recent years. One of the core problems in model-checking-assisted test generation is how to translate test objectives into temporal properties that can be fed to model checkers. Both [5, 9] have discussed the usage of formal specification in software testing. Gaudel further provided an overview for the conjunction area of testing and model checking, encouraging a more clear and uniformed field for the "industrial actors" [6].

Various works have shown that traditional structural test criteria can be used as the core standard for test generation via model checkers. For instance, Fraser et al. show that Modified Condition/Decision Coverage (MC/DC) can be encoded in Computational Tree Logic (CTL) [3], and be used by a model checker such as NuSMV for generating tests. The authors also evaluated different test criteria such as logic expression coverage criteria and dataflow criteria in the context of model-checking-assisted test generation. In [11], Hong et al. expressed the dataflow criteria in CTL. All these works presented their methods of translating one or more existing structure-based test objectives into temporal properties in CTL or LTL.

A key feature of our work is that it is based on Büchi automaton. This enables us to translate and encode linear temporal properties expressed in other formalisms such

as LTL. Our work further extends previous research based on LTL [22], in which the authors proposed coverage metrics measuring how well a requirement in LTL was covered by a test suite. In [21, 23, 31] we studied test metrics for covering states and transitions of a Büchi automata. While these works explored specification-based testing with Büchi automata, they were also limited to testing finite prefixes of infinite words. In this paper, we focus on the acceptance condition, which allows us to test temporal properties of infinite words.

Meanwhile, using formal specification to model requirements inspires a variety of automatic test generation and test execution tools. For example, AGEDIS project [8] was created as an effort to automate test generation and execution for distributed systems. Simulink Design Verifier [16] was developed as a verification and test generation tool for Simulink. Reactis [20] is another commercial tool developed by Reactive Systems Inc. that also accepts models in the Simulink and StateFlow modeling language. It uses a guided simulation strategy that is not as exhausting as model checking, hence avoiding the state explosion problem. In this work we developed a model-checking-assisted test generation algorithm that works with requirements encoded in Büchi automaton.

2 Preliminaries

2.1 Kripke Structures, Traces, and Tests

We model systems as *Kripke structures*. A Kripke structure is a finite transition system in which each state is labeled with a set of atomic propositions. Semantically atomic propositions represent primitive properties held at a state. Definition 1 formally defines Kripke structures.

Definition 1 (*Kripke Structures*) Given a set of atomic proposition $\mathcal{A}$, a Kripke structure is a tuple $\langle V, v_0, \rightarrow, \mathcal{V}\rangle$, where V is the set of states, $v_0 \in V$ is the start state, $\rightarrow \subseteq V \times V$ is the transition relation, and $\mathcal{V} : V \rightarrow 2^{\mathcal{A}}$ labels each state with a set of atomic propositions.

We write $v \rightarrow v'$ in lieu of $\langle v, v'\rangle \in \rightarrow$. We let $a, b, \ldots$ range over $\mathcal{A}$. We denote $\mathcal{A}_\neg$ for the set of negated atomic propositions. Together, $P = \mathcal{A} \cup \mathcal{A}_\neg$ defines the set of *literals*. We let $l_1, l_2, \ldots$ and $L_1, L_2, \ldots$ range over P and 2^P, respectively.

We use the following notations for sequences: let $\beta = v_0 v_1 \ldots$ be a sequence, we denote $\beta[i] = v_i$ for ith element of β, $\beta[i, j]$ for the subsequence $v_i \ldots v_j$, and $\beta^{(i)} = v_i \ldots$ for the ith suffix of β. A *trace* τ of the Kripke structure $\langle V, v_0, \rightarrow, \mathcal{V}\rangle$ is defined as a maximal sequence of states starting with v_0 and respecting the transition relation $\rightarrow$, i.e., $\tau[0] = v_0$ and $\tau[i-1] \rightarrow \tau[i]$ for every $i < |\tau|$. We also extend the labeling function $\mathcal{V}$ to traces: $\mathcal{V}(\tau) = \mathcal{V}(\tau[0])\mathcal{V}(\tau[1]) \ldots$.

Definition 2 (*Lasso-Shaped Sequences*) A sequence τ is *lasso-shaped* if it has the form $\alpha(\beta)^\omega$, where α and β are finite sequences. $|\beta|$ is the *repetition factor* of τ. The length of τ is a tuple $\langle|\alpha|, |\beta|\rangle$.

Definition 3 (*Test and Test Suite*) A test is a word on $2^{\mathcal{A}}$, where $\mathcal{A}$ is a set of atomic propositions. A test suite ts is a finite set of test cases. A Kripke structure $K = \langle V, v_0, \rightarrow, \mathcal{V}\rangle$ passes a test case t if K has a trace τ such that $\mathcal{V}(\tau) = t$. K passes a test suite ts if and only if it passes every test in ts.

2.2 Generalized Büchi Automata

Definition 4 A generalized Büchi automaton is a tuple $\langle S, S_0, \Delta, \mathcal{F}\rangle$, in which S is a set of states, $S_0 \subseteq S$ is the set of start states, $\Delta \subseteq S \times S$ is a set of transitions, and the acceptance condition $\mathcal{F} \subseteq 2^S$ is a set of sets of states.

We write $s \rightarrow s'$ in lieu of $\langle s, s'\rangle \in \Delta$. A generalized Büchi automaton is an ω-automaton, which can accept the infinite version of regular languages. A run of a generalized Büchi automaton $B = \langle S, S_0, \Delta, \mathcal{F}\rangle$ is an infinite sequence $\rho = s_0 s_1 \ldots$ such that $s_0 \in S_0$ and $s_i \rightarrow s_{i+1}$ for every $i \geq 0$. We denote $\inf(\rho)$ for a set of states that appear for infinite times on ρ. A successful run of B is a run of B such that for every $F \in \mathcal{F}$, $\inf(\rho) \cap F \neq \emptyset$.

In this work, we extend Definition 4 using state labeling approach in [7] with one modification: we label the state with a set of literals, instead of with a set of sets of atomic propositions in [7]. A set of literals is a succinct representation of a set of sets of atomic propositions: let L be a set of literals labeling state s, then semantically s is labeled with a set of sets of atomic propositions $\Lambda(L)$, where $\Lambda(L) = \{A \subseteq \mathcal{A} \mid (A \supseteq (L \cap \mathcal{A})) \wedge (A \cap (L \cap \mathcal{A}_\neg) = \emptyset)\}$, that is, every set of atomic propositions in $\Lambda(L)$ must contain all the atomic propositions in L but none of its negated atomic propositions. In the rest of the paper, we use Definition 5 for (labeled) generalized Büchi automata (GBA).

Definition 5 A labeled generalized Büchi automaton is a tuple $\langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, in which $\langle S, S_0, \Delta, \mathcal{F}\rangle$ is a generalized Büchi automaton, P is a set of literals, and the label function $\mathcal{L} : S \rightarrow 2^P$ maps each state to a set of literals.

A GBA $B = \langle \mathcal{A} \cup \mathcal{A}_\neg, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$ accepts infinite words over the alphabet $2^{\mathcal{A}}$. Let α be a word on $2^{\mathcal{A}}$, B has a run ρ induced by α, written as $\alpha \vdash \rho$, if and only if for every $i < |\alpha|$, $\alpha[i] \in \Lambda(\mathcal{L}(\rho[i]))$. B accepts α, written as $\alpha \models B$ if and only if B has a successful run ρ such that $\alpha \vdash \rho$.

GBAs are of special interests to the model checking community. Because a GBA is an ω-automaton, it can be used to describe temporal properties of a finite-state reactive system, whose executions are infinite words of an ω-language. Formally, a GBA accepts a Kripke structure $K = \langle V, v_0, \rightarrow, \mathcal{V}\rangle$, denoted as $K \models B$, if for every trace τ of K, $\mathcal{V}(\tau) \models B$. Efficient Büchi-automaton-based algorithms have

been developed for linear temporal model checking. The process of linear temporal model checking generally consists of translating the negation of a linear temporal logic property ϕ to a GBA $B_{\neg\phi}$, and then checking the emptiness of the product of $B_{\neg\phi}$ and K. If the product automaton is not empty, then a model checker usually produces an accepting trace of the product automaton, which serves as a counterexample to $K \models \phi$.

3 Accepting State Combination Coverage Criteria

A Büchi automaton differs from a finite automaton in its acceptance condition, which enables a Büchi automaton to accept infinite words. Since we are interested in testing a reactive system, and particularly the temporal patterns of its infinite executions, we focus on covering the acceptance condition of a Büchi automaton. In what follows, we denote $\bigcup \mathcal{F} = F_0 \cup \cdots \cup F_{n-1}$, where $\mathcal{F} = \{F_0, \ldots, F_{n-1}\}$.

Definition 6 (*Accepting State Combination*) Given a Büchi automaton $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, an accepting state combination (ASC) C is a minimal set of states such that (i) $C \subseteq \bigcup \mathcal{F}$; (ii) $\forall F \in \mathcal{F}, F \cap C \neq \emptyset$.

Definition 7 (*Covered Accepting State Combinations*) Given a GBA $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, let C be one of B's ASCs,

1. A run ρ of B covers C if ρ visits every state of C infinitely often;
2. A test t *strongly* covers B's ASC C if t satisfies B and every successful run induced by t on B covers C;
3. A test t *weakly* covers B's ASC C if at least one run induced by t on B covers C.

Intuitively, an ASC is a basic unit for the sets of acceptance states covered by a successful run. That is, any successful run must visit every state of some ASC infinitely often, as stated in Lemma 1.

Lemma 1 *Given a Büchi automaton $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, ρ is a successful run of B if and only if ρ covers some ASC of B.*

Proof

($\Rightarrow$) Let $inf(\rho)$ be the set of states visited by ρ infinitely often, and $C' = inf(\rho) \cap (\bigcup \mathcal{F})$ be the set of acceptance states visited by ρ infinitely often. Then, C' satisfies the following properties: (i) $C' \subseteq \bigcup \mathcal{F}$; and, (ii) $\forall F \in \mathcal{F}.(F \cap C') \neq \emptyset$. (ii) is due to the assumption that ρ is a successful run of B. Note that by Definition 6 the ASCs are all the *minimal sets* satisfying (i) and (ii). Therefore, C' has to be a superset of some ASC say C. It follows that ρ visits every state in C infinitely often, that is, ρ covers C.

($\Leftarrow$) Let C be an ASC of B covered by ρ, that is, $\inf(\rho) \supseteq C$. By Definition 6 $\forall F \in \mathcal{F}.F \cap C \subseteq F \cap \inf(\rho) \neq \emptyset$. Therefore, ρ satisifies the acceptance condition of B and hence it is one of B's successful runs. □

Algorithm 1 ASC_Gen($B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$)

Require: B is a GBA
Ensure: Return a set of all Accepting State Combinations $\mathcal{C}_\perp$ of B.
1: $\mathcal{C}_\perp = \emptyset$; bool new = tt;
2: **for** each state $s_{0_j} \in F_0$, $j = 1 \cdots |F_0|$ **do**
3: $\cdots$
4: **for** each state $s_{n-1_k} \in F_{n-1}$, $k = 1 \cdots |F_{n-1}|$ **do**
5: $C = \bigcup\{s_{0_j}\}, \cdots, \{s_{n-1_k}\}$.
6: **for** every $C' \in \mathcal{C}_\perp$ **do**
7: **if** $C' \subseteq C$ **then**
8: new = ff; break;
9: **end if**
10: **if** $C' \supset C$ **then**
11: $\mathcal{C}_\perp = \mathcal{C}_\perp - C'$
12: **end if**
13: **end for**
14: **if** new **then**
15: $\mathcal{C}_\perp = \mathcal{C}_\perp \cup C$; new = tt;
16: **end if**
17: **end for**
18: **end for**
19: **return** $\mathcal{C}_\perp$;

Algorithm 1 gives a code example that computes all the ASCs for a given GBA B. We denote $\mathcal{C}(B)$ as the set of the ASCs of B. The coverage metrics are thus about covering these ASCs. Definition 7 presents two different ways to cover an ASC, due to the non-deterministic nature of a GBA. In the strong variant, every successful run induced by a successful test is required to visit the ASC infinitely often; and in the weak variant, only one successful run induced by the test is required to visit the ASC infinitely often. By Lemma 2 the strong coverage criterion subsumes the weak one. Users may pick and choose the type of coverage, depending on the desired strength of testing set forth for an application.

Lemma 2 *An ASC C of a Büchi automaton B is weakly covered by a test t if C is strongly covered by t.*

Proof It immediately follows from Definition 7. □

Definition 8 (*Strong/Weak ASC Coverage Metric and Criterion*)
Given a generalized Büchi automaton $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, let $\mathcal{C}(B)$ be the set of B's ASCs, the strong (or weak) ASC coverage metric for a test suite T on B is $\frac{|\delta'|}{|\delta|}$, where $\delta' = \{C \mid t$ strongly (or weakly) covers $C\}$, and $\delta = \mathcal{C}(B)$. T strongly (or weakly) covers δ if and only if $\delta' = \delta$.

It shall be noted that the number of all the ASCs is significantly smaller than the number of all the possible combinations of acceptance states. The first is bounded by $O(m^n)$ and the latter is bounded by 2^m, where $n = |\mathcal{F}|$ is the cardinality of

$\mathcal{F}$ and $m = |\bigcup \mathcal{F}|$ is the number of acceptance states. By focusing on covering ASCs instead of every combination of acceptance states, we significantly reduce the complexity of computing our coverage metrics.

4 Model-Checking-Assisted Test Generation for Accepting States Combination Coverage

To improve the efficiency of test case generation, we develop a model-checking-assisted algorithm for generating test cases under proposed criteria. The algorithm uses the counterexample capability of an off-the-shelf linear temporal model checker to generate test cases. One of the fundamental questions in model-checking-assisted test generation is how to specify test objectives in a formalism acceptable by a model checker. The properties specifying test objectives are often referred to as "trap properties" in the context of model-checking-assisted test generation. In our case, "trap properties" are defined in the form of Büchi automaton. We synthesize a set of "trap (Büchi) automata" from the original Büchi automaton, using graphic transformation techniques.

Definition 9 (*ASC Excluding Automaton*) Given a Büchi automaton $B = \langle P, S, S_0, \Delta, \mathcal{L}, F \equiv \{F_0, \ldots, F_{n-1}\}\rangle$ and an ASC C, an ASC excluding (ASC-E) GBA is $B_{\overline{C}} = \langle P, S^e, S_0^e \equiv S_0 \times \{\bot\}, \Delta^e, \mathcal{L}^e, \mathcal{F}^e \equiv \{F_0^e, \ldots, F_{n-1}^e\}\rangle$, where,

1. $S^e = (S \times (C \cup \{\bot\})) - \bigcup_{s \in C}\{\langle s, s\rangle\}$
2. $F_i^e = \{\langle s, u\rangle \mid s \in F_i \wedge u \neq \bot\}$.
3. $\Delta^e = \{(\langle s, u\rangle \rightarrow \langle s', u'\rangle) \mid (s \rightarrow s') \in \Delta \wedge (u = u' \vee (u = \bot))\}$
4. $\mathcal{L}^e(\langle s, u\rangle) = \mathcal{L}(s)$

Intuitively speaking, for a Büchi automaton B and an ASC C, its ASC-E-GBA $B_{\overline{C}}$ accepts precisely B's successful runs, except for those visiting C infinitely often. $B_{\overline{C}}$ does so by extending B with additional copies. To distinguish these copies, the states of the original copy (denoted as $B_\bot$) is indexed by the symbol $\bot$, whereas the states of each additional copy (denoted as $B_{\neg s}$) are indexed by a state $s \in C$. $B_{\neg s}$ inherits all the states from B except for s, the very state indexing B (i.e. $\langle s, s\rangle$ in Definition 9(1)). Intuitively, $B_{\neg s}$ accepts all the successful runs of B, except for those visiting the indexing state s. Each copy $B_{\neg s}$ retains the transitions from B (except for, of course, the ones associating with the indexing state s, which is not in $B_{\neg s}$). In addition, for each transition of the original copy $B_\bot$, say $\langle s, \bot\rangle \rightarrow \langle s', \bot\rangle$, we create $|C|$ copies of that transition, each of which replaces the destination node $\langle s', \bot\rangle$ with its counterpart in a copy $B_{\neg t}$ indexed by a state $t \in C$ (Definition 9(3)). Formally, for each transition $\langle s, \bot\rangle \rightarrow \langle s', \bot\rangle$, we add more transitions $\delta = \bigcup_{t \in C}\{\langle s, \bot\rangle \rightarrow \langle s', t\rangle\}$. We refer to these new transitions as "bridging" transitions. Note that these bridging transitions go one-way only, that is, they jump from the original copy $B_\bot$ to a copy $B_{\neg t}$, where $t \in C$.

There are no transitions linking $B_{\neg t}$ back to $B_\perp$. As a final touch, only the copies indexed by the states of C, not the original one, retain the acceptance condition. Since every additional copy $B_{\neg s}$ misses its indexing state s, it implies that the indexing state is not part of the acceptance condition of $B_{\neg s}$.

It follows from the construction of $B_{\overline{C}}$ that a successful run ρ of $B_{\overline{C}}$ must satisfy the following conditions: (1) it starts at $B_\perp$ (i.e. the start states S_0^e in Definition 9) and can spend only a finite number of steps in $B_\perp$, since $B_\perp$ does not have an acceptance state (Definition 9(2)); (2) at some point, ρ will make a non-deterministic choice to take a bridging transition to one of the copies indexed by a state $s \in C$, say $B_{\neg s}$, and satisfy $B_{\neg s}$'s acceptance condition. Clearly ρ is also a successful run of the original GBA B, since each state on ρ may be mapped back to a state of B, and the acceptance condition of $B_{\neg s}$ accepting ρ is a subset of the acceptance condition of B. In addition, because $B_{\neg s}$ does not have s (Definition 9(1)), ρ cannot visit s infinitely often. Furthermore, no matter which copy the ρ jumps to, there is no way that ρ can visit every state of C infinitely often, since there is one state of C missing in that copy, i.e., its indexing state. It follows that a successful run ρ' of B becomes a successful run of its ASC-E-GBA only if ρ' does not visit every state of C infinitely often.

Theorem 1 *Given a GBA* $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$ *and an ASC* C, $B_{\overline{C}} = \langle P, S^e, S_0^e, \Delta^e, \mathcal{L}^e, \mathcal{F}^e\rangle$ *be the ASC-E-GBA for* B *and* C, *then a test* t *satisfies* $B_{\overline{C}}$ *if and only if* B *has a successful run* ρ *such that* $t \vdash \rho$ *and* $\inf(\rho) \not\supseteq C$.

Proof

($\Rightarrow$) By Definition 9, since t satisfies $B_{\overline{C}}$, $B_{\overline{C}}$ has a successful run ρ' such that $t \vdash \rho'$. We construct a successful run ρ for B from ρ' by projecting states in $B_{\overline{C}}$ to B. Assume that $\rho' = \langle s_0, u_0\rangle\langle s_1, u_1\rangle \ldots$, then $\rho = s_0 s_1 \ldots$. By Definition 9, ρ has to be a successful run of B, since all the transitions in ρ': $\langle s_i, u_i\rangle \rightarrow \langle s_{i+1}, u_{i+1}\rangle$ follow the same guards as $s_i \rightarrow s_{i+1}$, and the acceptance conditions in $B_{\overline{C}}$ are the same states in B marking with states in C. We also have $\mathcal{L}^e(\langle s, u\rangle) = \mathcal{L}(s)$ and $t \vdash \rho'$, therefore $t = \mathcal{L}^e(\rho') = \mathcal{L}(\rho)$. Hence, $t \vdash \rho$.
Now we will prove $\inf(\rho) \not\supseteq C$ by showing at least one state in ASC C is not visited by ρ infinitely often. Note that by the construction of $B_{\overline{C}}$ every acceptance state is resided in a copy of B indexed by a state (i.e. not $\perp$). Therefore, since ρ' is a successful run of $B_{\overline{C}}$, ρ' must visit at least one copy of B indexed by a state. Without loss of generality, let s be the indexing state of a B's copy that ρ' visits. We denote the indexed copy as $B_{\neg s}$. By Definition 9 there is no outgoing transition from $B_{\neg s}$ to other copies of B. Therefore once ρ' is in $B_{\neg s}$, it is "trapped" within $B_{\neg s}$ and only states ρ' may visit infinitely often are those inside $B_{\neg s}$. By Definition 9 $B_{\neg s}$ does not include a copy of state s itself, therefore $\inf(\rho')$ does not visit s or its indexed copies infinitely often. That is, $s \notin \inf(\rho)$. Note that an indexing state must be a state in ASC C by Definition 9(1). Therefore $\inf(\rho) \not\supseteq C$.

($\Leftarrow$) Let's denote $D = C - \inf(\rho)$. Since $\inf(\rho) \not\supseteq C$, $D \neq \emptyset$. Let $\rho = s_0 s_1 \ldots$. Let s_k for the last occurrence of any state in D on ρ. If ρ does not visit any state in D, then $k = 0$. We randomly pick a state $s \in D$. Now we construct $\rho' = \langle s_0, u_0 \rangle \langle s_1, u_1 \rangle \ldots$ such that $\forall i \leq k.u_i = \perp$ and $\forall i > k.u_i = s$. Intuitively speaking, ρ' visits states in the copy indexed by $\perp$ (denoted as $B_\perp$), and then jump to the copy $B_{\neg s}$. Note that $B_{\neg s}$ has an (indexed) copy of every state of B, except for s. Since $s_{k+1} s_{k+2} \ldots$ does contain any state of C, it does not visit (an indexed copy of) s either. Clearly ρ' is a run of $B_{\overline{C}}$, since it respects the transition relation in Definition 9.

Next we will show that ρ' also satisfies the acceptance condition of B. As a successful run of B, ρ satisfies B's acceptance condition, that is, $\forall \mathbf{F}_i \in \mathcal{F}. \inf(\rho) \cap \mathbf{F}_i \neq \emptyset$. Without loss of generality, we pick up $\mathbf{F}_i$ and show that ρ' will visit some state in $\mathbf{F}_i^e$, $B_{\overline{C}}$'s counterpart of $\mathbf{F}_i$, infinitely often. Let $s_f \in \bigcup \mathbf{F}_i$ be a state visited by ρ infinitely often, Because there is an one-to-one mapping between states on ρ and ρ', ρ' also visits $\langle s_f, s \rangle$ infinitely often. By Definition 9, $\langle s_f, s \rangle \in \mathbf{F}_i^e$. Therefore, we have $\forall \mathbf{F}_i^e \in \mathcal{F}^e. \inf(\rho') \cap \mathbf{F}_i^e \neq \emptyset$.

Finally we show $t \vdash \rho'$ by noting Definition 9(4), that is, $B_{\overline{C}}$ is labelled in the same way as B. Therefore, if t induces ρ on B, it may also induce ρ' on B'. Therefore t induces a successful run ρ' of $B_{\overline{C}}$ and hence it satisfies $B_{\overline{C}}$. □

As an example, consider a GBA in Fig. 1. The GBA represents LTL property $\phi = \mathbf{G}(\neg t \implies ((\neg p \mathbf{U} t) \vee \mathbf{G} \neg p))$, a temporal requirement used with the GIOP model [14] in our experimental study. ϕ's semantics is explained in Sect. 6. Since its acceptance condition $\{\{s_0, s_2\}\}$ contains only one set of states, its ACSs are the singleton set of each acceptance state, that is, $\{s_0\}$ and $\{s_2\}$. Figure 2 gives an ASC excluding automaton $B_{\overline{\{s_0\}}}$ with respect to the ASC $\{s_0\}$. $B_{\overline{\{s_0\}}}$ has two copies of B: the original copy $B_\perp$ and the copy indexed by s_0, the only state in $C = \{s_0\}$. Note that the indexing state itself s_0 (i.e. $\langle s_0, s_0 \rangle$) and its transitions are removed from the copy $B_{\overline{\{s_0\}}}$. These are represented by the dashed circle and lines in Fig. 2. The highlighted solid links represent bridging transitions linking from the original copy to the copy indexed by s_0. Since the only acceptance state, $\langle s_2, s_0 \rangle$ exists in the copy $B_{\neg s_0}$, a successful run of $B_{\overline{\{s_0\}}}$ must visit s_2 (in the form of $\langle s_2, s_0 \rangle$), not s_0 (in the form of $\langle s_0, s_0 \rangle$), infinitely often.

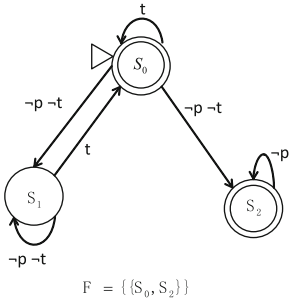


Fig. 1 A general Büchi automaton representing the LTL property $\mathbf{G}(\neg t \implies ((\neg p \mathbf{U} t) \vee \mathbf{G} \neg p))$

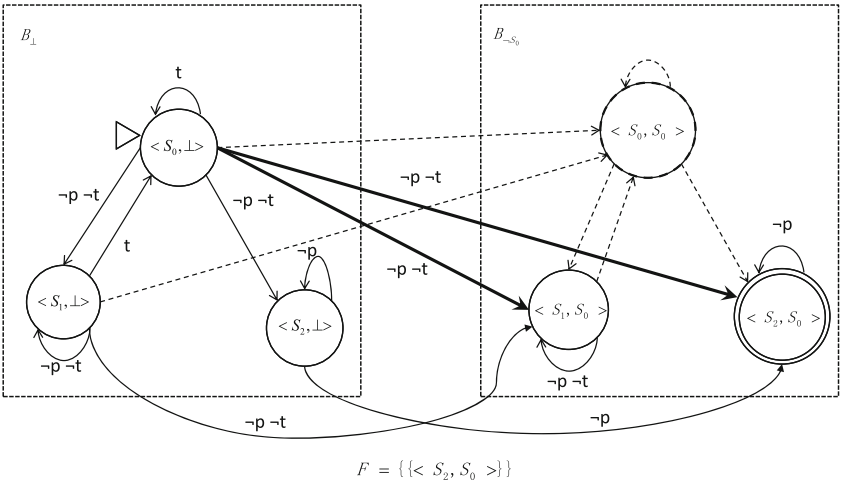


Fig. 2 An ASC excluding general Büchi automaton for the ASC $\{s_0\}$ of the GBA in Fig. 1

Algorithm 2 TestGen_SC($B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, $K_m = \langle \mathcal{S}, s_0, \rightarrow, \mathcal{V}\rangle$)

Require: B is a GBA, K_m is a system model, and K_m satisfies B
Ensure: Return a test suite ts such that ts strongly covers every ASC of B and K_m passes ts. Return $\emptyset$ if such a test suite is not found;
1: $\mathcal{C}(B) = ASC\_Gen(B)$;
2: **for** every ASC $C \in \mathcal{C}(B)$ **do**
3: $B_{\overline{C}} = \langle P, S \times C_s \cup \emptyset, S_0 \times \emptyset, \Delta_e, \mathcal{L}_e, \mathcal{F}_e\rangle$;
4: $\tau = MC\_isEmpty(\neg B_{\overline{C}}, K_m)$;
5: **if** $|\tau| \neq 0$ **then**
6: $ts = ts \cup \{\mathcal{V}(\tau)\}$
7: **else**
8: **return** $\emptyset$;
9: **end if**
10: **end for**
11: **return** ts;

Algorithm 2 generates a test suite strongly covering all the ASCs of a Büchi automaton B. It makes use of ASC excluding automata. For each of B's ASCs, Algorithm 2 constructs an ASC excluding GBA $B_{\overline{C}}$ with respect to C. ASC_Gen is a sub-routine computing all the ASCs for a GBA. The algorithm uses a model checker to search for a successful run τ on the production of $\neg B_{\overline{C}}$ and K_m, and τ is a successful run of $\neg B_{\overline{C}}$ accepting K_m. $MC\_isEmpty$ refers to the emptyness checking algorithm in an off-the-shelf linear temporal model checker. If a run exists, it returns with a test set containing $t = \mathcal{V}(\tau)$, which is a word accepted by $\neg B_{\overline{C}}$. Consequently, t cannot be accepted by $B_{\overline{C}}$. Note that τ is a successful run of the production of B and K_m, therefore based on Theorem 1, for every successful run ρ that $t \vdash \rho$, $\inf(\rho) \supseteq C$. Based on Definition 7, ts is a test suite that strongly covers C.

Theorem 2 *If the test suite ts returned by Algorithm 2 is not empty, then (i) K_m passes ts and (ii) ts strongly covers all the ASCs of B.*

Proof (i) For each test $t \in ts$, there is a related ASC C and $MC\_isEmpty(\neg B_{\overline{C}}, K_m))$ returns a successful run τ of the product of $\neg B_{\overline{C}}$ and K_m such that $\mathcal{V}(\tau) = t$. It follows that τ is also a successful run on K_m, and K_m shall pass $t = \mathcal{V}(\tau)$. Therefore, K_m passes every test case in ts.

(ii) As shown in (i), for each $t \in ts$, there is a related ASC C and a successful run τ of the production of $\neg B_{\overline{C}}$ and K_m such that $\mathcal{V}(\tau) = t$. We show that t strongly covers the ASC C.

First, since τ is also a trace of K_m and K_m satisfies B by the precondition of Algorithm 2, $\tau \models B$.

Second, we will prove by contradiction that every successful run of B that is induced by the test case t shall cover C, i.e., every state in C shall be visited infinitely often. Suppose that it were not the case. Let ρ be a successful run of B that is induced by t, and ρ does not cover C.

We may now construct a run ρ' by "tracing" ρ's states on $B_{\overline{C}}$ as follows: since ρ does not cover C, there must exist at least one state $s \in C$ and $s \notin \inf(\rho)$. As ρ is a lasso-shaped trace, we label every state in the non-circular prefix of ρ with $\perp$, and every state in the circular subtrace of ρ with s. By this construction and Definition 9, ρ' is a successful run on $B_{\overline{C}}$.

Since t induces ρ, and ρ' is obtained by adding labels to the states on ρ, t also induces ρ' on $B_{\overline{C}}$. Therefore, t shall be accepted by $B_{\overline{C}}$. However, we have shown that t is accepted by $\neg(B_{\overline{C}})$ which is a complement of $B_{\overline{C}}$, and thus should have no common words in their languages. If t can be accepted by both automata, then $t \in L(\neg B_{\overline{C}}) \cap L(B_{\overline{C}}) \neq \emptyset$. Therefore, every successful run of B that accepts t shall cover C. □

Compared with constructing an ASC excluding automaton, constructing a Büchi automaton accepting the runs *weakly* covering an ASC is relatively straightforward: the new automaton may be obtained by removing from the acceptance condition the states *not* in the ASC, that is, replacing the acceptance condition with C. Definition 10 describes the process.

Definition 10 (*ASC Marking Automaton*) Given a GBA $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F} \rangle$ and an ASC C, $B_C = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_C \rangle$ is the ASC-Marking (ASC-M) Büchi automaton for B with respect to C, in which $\mathcal{F}_C = \{F \cap C | F \in \mathcal{F}\}$.

Clearly $L(B_C) \subseteq L(B)$, since the acceptance condition of B_C is a refinement of that of B, that is, $\forall F \in \mathcal{F}, \exists F' \in \mathcal{F}_C, (F' \subseteq \mathcal{F}')$ and $\forall F' \in \mathcal{F}_C, \exists F \in \mathcal{F}, (F' \subseteq F)$. Note that, $\mathcal{F}_C$ in Definition 10 is a subset of $2^C - \emptyset$. By Definition 6, C has to be a minimal set of states that $\forall F \in \mathcal{F}$, $F \cap C \neq \emptyset$. Combining these two conditions, it is straightforward $\bigcup \mathcal{F}_C = C$. Based on Definition 4, this means that for a run to be successful on B_C, all states in C must be visited infinitely often. Therefore we

rewrite the acceptance condition for B_C as $\mathcal{F}_C = \{\{s\} \mid s \in C\}$, i.e., a set of singleton sets of states in C. For the rest of the paper, we consider ASC-M-GBA to be defined with the rewritten acceptance condition.

Lemma 3 *Given a GBA $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, let C be one of its ASCs and $B_C = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_C\rangle$ be its ASC-marking automaton for C, then ρ covers C if and only if ρ is also a successful run of B_C.*

Proof
($\Rightarrow$) By the construction of $\mathcal{F}_C$, for every $F' \in \mathcal{F}_C$, there exists a $F \in \mathcal{F}$ such that $F' = F \cap C$. Since $F \cap C \neq \emptyset$ by Definition 6, $F' \cap C \neq \emptyset$. Moreover, since ρ covers C, $\inf(\rho) \supseteq C$. Therefore, $\inf(\rho) \cap F' \neq \emptyset$. That is, ρ is also a successful run of B_C.
($\Leftarrow$) We will use contradition to show that ρ covers C. Suppose not, and let $s \in C$ be an acceptance state not visited by ρ infinitely often. Since ρ is a successful run of B_C, by the construction of $\mathcal{F}_C$ we have that for every $F \in \mathcal{F}$, $(F \cap C) \cap \inf(\rho) \neq \emptyset$. Moveover, since $s \notin \inf(\rho)$, $(F \cap (C - \{s\}) \cap \inf(\rho) \neq \emptyset$. Hence $F \cap (C - \{s\}) \neq \emptyset$. Moreover, since C is a ASC of B, $C \subseteq \bigcup \mathcal{F}$ and hence $(C - \{s\}) \subset C \subseteq \mathcal{F}$. Therefore, contradicting to the lemma's condition, C cannot be a ASC of B, which requires C to be a minimal set satisfying (i) $C \subseteq \bigcup \mathcal{F}$; and, (ii) $\forall F \in \mathcal{F}.(F \cap C) \neq \emptyset$. Therefore, the assumption could not be true, and hence ρ covers C. □

Algorithm 3 TestGen_WC($B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, $K_m = \langle \mathcal{S}, s_0, \rightarrow, \mathcal{V}\rangle$)

Require: B is a GBA, K_m is a system model, and K_m satisifies B.
Ensure: Return a test suite ts such that ts weakly covers every ASC in $\mathcal{C}(B)$ and K_m passes ts. Return $\emptyset$ if such a test suite is not found;

```
1: C(B) = ASC_Gen(B);
2: for every ASC C ∈ C(B) do
3:    B_C = ⟨P, S, S_0, Δ, L, F_C⟩, where F_C = {{s} | s ∈ C};
4:    τ = MC_isEmpty(B_C, K_m);
5:    if |τ| ≠ 0 then
6:       ts = ts ∪ {V(τ)};
7:    else
8:       return ∅;
9:    end if
10: end for
11: return ts;
```

Algorithm 3 generates tests that weakly cover the ASC C. We construct an ASC-M-GBA B_C in Algorithm 3, and then search for a successful run on the product of B_C and the system model K_m. If such run τ exists, the test case $t = \mathcal{V}(\tau)$ is then added to ts and return as the singleton test suite. Since $t \in L(B_C)$ and $L(B_C) \subseteq L(B)$, $t \in L(B)$. By Definitions 10 and 7, since τ is a successful run of B_C that weakly covers C on B, therefore t is a test case that weakly covers C on B.

Theorem 3 *If the test suite ts returned by Algorithm 3 is not empty, then (i) K_m passes ts and (ii) ts weakly covers all the ASCs of B.*

Proof (i) For each $t \in ts$, there is a related ASC C and $MC\_isEmpty\ (B_C, K_m))$ returns a successful run of the production of B_C and K_m such that $\mathcal{V}(\tau) = t$. Since any successful run of the production of B_C and K_m shall also be a trace of K_m, τ is also a trace of K_m. Therefore, K_m shall pass t. That is, K_m passes every test case in ts.

(ii) As shown in (i), for each $t \in ts$, there is a related ASC C and a successful run τ of the production of B_C and K_m such that $\mathcal{V}(\tau) = t$. We show that t weakly covers C, by showing that τ is also a successful run on B, and visits all states in C infinitely often.

By Definition 10, the only difference bewteen B_C and B is that B_C has the acceptance condition replaced with a set of singleton sets of states in C. By Definition 6, we have two conclusions. First, for each set $F_C \in \mathcal{F}_C$, there exists a set $F \in \mathcal{F}$ that $F_C \subseteq F$. Second, for each set $F \in \mathcal{F}$, there exists at least one state $s \in C$ that $s \in F$. Based on the two conclusions, we have $L(B_C) \subseteq L(B)$. Therefore τ must be a successful run on B as well, and it covers all states in C infinitely often. □

5 ASC-Induced Property Refinement

ASC coverage metrics measure the conformance of a design against a formal requirement in Büchi automaton. Lacking of ASC coverage may be contributed either by bugs in the design, or by the deficiency of the requirement, or sometimes by both. We develop an algorithm that identifies the deficiency of the requirement and refines the requirement, using the information collected from test case generation (Algorithm 3).

We consider the refinement in terms of language inclusion, that is, if the language of an automaton B' is a subset of that of B, we refer to B' as a refinement of B'. Given a Kripke structure K representing a system with its requirement as GBA B, we develop an algorithm to refine B if not every ASC of B can be weakly covered w.r.t. K.

Given a Kripke structure K_m as a system model and a GBA $B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$ as its requirement, the basic steps of refining B w.r.t. B are described below:

1. Identify the set of ASCs $\mathcal{C} = \{C_0, \ldots, C_n\}$ of B that are weakly covered w.r.t. K_m. $\mathcal{C}$ may be identified by Algorithm 3 during the test case generation;
2. Produce an automaton $B_{\mathcal{C}} = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_{\mathcal{C}}\rangle$, where $\mathcal{F}_{\mathcal{C}} = \{F \cap (\bigcup \mathcal{C}) | F \in \mathcal{F}\}$

Algorithm 4 TestGen_RefineWC($B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, $K_m = \langle \mathcal{S}, s_0, \rightarrow, \mathcal{V}\rangle$)

Require: B is a GBA, K_m is a system model, and K_m satisfies B.
Ensure: Return a test suite ts such that ts weakly covers all ASCs that can be covered in $\mathcal{C}_\perp$ and K_m passes ts. Also returns a Büchi automata with a refined acceptance condition;
1: $\mathcal{C}(B) = ASC\_Gen(B)$;
2: **for** every ASC $C \in \mathcal{C}(B)$ **do**
3: $B_C = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_C\rangle$, where $\mathcal{F}_C = \{\{s\} \mid s \in C\}$;
4: $\tau = MC\_isEmpty(B_C, K_m)$;
5: **if** $|\tau| \neq 0$ **then**
6: $ts = ts \cup \{\mathcal{V}(\tau)\}$
7: $\mathcal{C} = \mathcal{C} \cup \{C\}$
8: **end if**
9: **end for**
10: $B_\mathcal{C} = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_\mathcal{C} \equiv \{F \cap (\bigcup C) \mid F \in \mathcal{F}\}\rangle$
11: **return** ts and $B_\mathcal{C}$;

For example, consider a Büchi automaton B with an acceptance condition $\mathcal{F} = \{F_0, F_1, F_2\}$, in which $F_0 = \{s_1, s_2, s_4\}$, $F_1 = \{s_2, s_3, s_4\}$, $F_2 = \{s_1, s_3, s_4\}$. The ASCs for B would be $C_0 = \{s_1, s_2\}$, $C_1 = \{s_2, s_3\}$, $C_2 = \{s_1, s_3\}$ and $C_3 = \{s_4\}$. Assume that only C_0 and C_2 can be weakly covered w.r.t. a system K, we can then refine B to $B_\mathcal{C}$, where $B_\mathcal{C}$'s acceptance condition is $\{\{s_1, s_2\}, \{s_2, s_3\}, \{s_1, s_3\}\}$.

The entire process of property refinement may be automated by extending the test generation algorithm, as described in Algorithm 4 Clearly $L(B_\mathcal{C}) \subseteq L(B)$, since the acceptance condition of $B_\mathcal{C}$ is a refinement of the acceptance condition of B. Moreover, by the construction of $\mathcal{F}_\mathcal{C}$, $B_\mathcal{C}$ contains all the ASCs of B that can be weakly covered by some tests passed by K_m. Theorem 4 states that the refined automaton, $B_\mathcal{C}$, is still satisfied by K_m. In other words, by refining B to $B_\mathcal{C}$, we obtain a "restricted" version of the property that more closely specifies a requirement for K_m.

Theorem 4 *Given a GBA B and a Kripke structure K_m such that $K_m \models B$, let $B_\mathcal{C}$ be the GBA returned by $TestGen\_RefineWC(B, K_m)$, then, (i) $L(B_\mathcal{C}) \subseteq L(B)$ and (ii) $K_m \models B_\mathcal{C}$.*

Proof (i) By the construction of $B_\mathcal{C}$, it differs from B only on its acceptance condition. Therefore, each run of $B_\mathcal{C}$ is also a run of B. Moreover, $B_\mathcal{C}$'s acceptance condition $\mathcal{F}_\mathcal{C}$ is also a refinement of B's acceptance condition $\mathcal{F}$, that is, $\forall F' \in \mathcal{F}_\mathcal{C}.\exists F \in \mathcal{F}.(F' \subseteq F)$ and $\forall F \in \mathcal{F}.\exists F' \in \mathcal{F}_\mathcal{C}.(F' \subseteq F)$. It follows that each successful of $B_\mathcal{C}$ must also be a successful run of B, and hence $L(B_\mathcal{C}) \subseteq L(B)$. That is, $B_\mathcal{C}$ is a refinement of B in terms of language inclusion.

(ii) We then prove K_m satisfies $B_\mathcal{C}$ by contradiction. Assume it is not the case, then there is a trace t of K_m that does not induce a successful run of $B_\mathcal{C}$. Since $K_m \models B$. t induces at least one successful run of B, denoted as ρ. By the assumption ρ could not be a successful run of $B_\mathcal{C}$. By Lemma 1, since $\rho \models B$, there has to be at least one ASC, denoted C_ρ, that is covered by ρ. By Lemma 3, ρ is also a successful run of the ASC-marking automaton B_{C_ρ}. Therefore, $MC\_isEmpty(B_{C_\rho}, K_m)$ returns a successful run, and $C_\rho \in \mathcal{C}$.

Now we will show that for every $F' \in \mathcal{F}_\mathcal{C}$, $(C_\rho \cap F') \neq \emptyset$. By the construction of $\mathcal{F}_\mathcal{C}$, there exists at least one $F \in \mathcal{F}$ such that $F' = F \cap (\bigcup \mathcal{C})$. By Definition 6 $F \cap C_\rho \neq \emptyset$. Since $C_\rho \in \mathcal{C}$, $(C_\rho \cap F') \neq \emptyset$.

Finally, since ρ covers C_ρ, $\inf(\rho) superseteq C_\rho$. It follows that for every $F' \in \mathcal{F}_\mathcal{C}$, $(\inf(\rho) \cap F') \neq \emptyset$. That is, ρ is also a successful run of $B_\mathcal{C}$, which contradicts to our assumption. Therefore, $K_m \models B_\mathcal{C}$. □

Note that the refined $B_\mathcal{C}$ returned by $TestGen\_RefineWC(B, K_m)$ is not the optimal refinement in terms of semantic equivalency. Consider the same example mentioned above: a Büchi automaton B with an acceptance condition $\mathcal{F} = \{F_0, F_1, F_2\}$, and its refinement $B_\mathcal{C}$, where $B_\mathcal{C}$'s acceptance condition is $\{\{s_1, s_2\}, \{s_2, s_3\}, \{s_1, s_3\}\}$. Based on the given condition, only $C_0 = \{s_1, s_2\}$ and $C_2 = \{s_1, s_3\}$ can be weakly covered for B. However, a trace that covers $C_1 = \{s_2, s_3\}$ can also be accepted by $B_\mathcal{C}$, indicating that there is still space for further refinement.

We propose another alternative for ASC-induced property refinement, based directly upon the ASC-M-GBA. Algorithm 5 describes the refining process. Intuitively, Algorithm 5 collects every ASC-M-GBA generated while generating test cases towards the weak ASC coverage metric. The union of these ASC-M-GBA is a tighter refinement than $B_\mathcal{C}$ from Algorithm 4. Theorem 5 proves the legitimacy of this approach.

Algorithm 5 TestGen_RefineWCAlt($B = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}\rangle$, $K_m = \langle \mathcal{S}, s_0, \rightarrow, \mathcal{V}\rangle$)

Require: B is a GBA, K_m is a system model, and K_m satisfies B.

Ensure: Return a test suite ts such that ts weakly covers all ASCs that can be covered in $\mathcal{C}_\perp$ and K_m passes ts. Also returns a set of Büchi automata, the union of which represents the refined property;

1: $\mathcal{C}(B) = ASC\_Gen(B)$;
2: $S_{B_{ref}} = \emptyset$
3: **for** every ASC $C \in \mathcal{C}(B)$ **do**
4: $B_C = \langle P, S, S_0, \Delta, \mathcal{L}, \mathcal{F}_C\rangle$, where $\mathcal{F}_C = \{\{s\} \mid s \in C\}$;
5: $\tau = MC\_isEmpty(B_C, K_m)$;
6: **if** $|\tau| \neq 0$ **then**
7: $ts = ts \cup \{\mathcal{V}(\tau)\}$
8: $S_{B_{ref}} = S_{B_{ref}} \cup B_C$
9: **end if**
10: **end for**
11: **return** ts and $S_{B_{ref}}$;

Theorem 5 *Given a GBA B and a Kripke structure K_m such that $K_m \models B$, let $S_{B_{ref}}$ be the set of GBA returned by $TestGen\_RefineWCAlt(B, K_m)$, then, (i) $S_{B_{ref}}$ is a refinement of B and (ii) for any trace t of K_m, there exists at least one $B' \in S_{B_{ref}}$ that $t \models B'$.*

Proof First, $S_{B_{ref}}$ includes each ASC-M-GBA that was built w.r.t one of the coverable ASCs. Based on Definition 10, we know that each ASC-M-GBA is a refinement of the original GBA B. Therefore it follows the union of these ASC-M-GBAs, as represented by $S_{B_{ref}}$ are also a refinement of B.

For the second part, we prove by contradiction. By the condition of Algorithm 5, K_m satisfies B, i.e., every trace of K_m can be accepted by B. Suppose there exists a trace t that none of the $B' \in \mathcal{S}_{B_{ref}}$ is able to accept. Since $t \models B$, there must exist at least one run ρ that $t \vdash \rho$ and ρ is a successful run on B. By Lemma 1, ρ must have covered at least one ASC C of B. As a result, $MC\_isEmpty(B_C, K_m))$ in Algorithm 5 would not return empty, and B_C would be included in $S_{B_{ref}}$, and ρ is a successful run on B_C. Hence t can be accepted by $B_C \in S_{B_{ref}}$. This contradicts the previous assumption. □

6 Experiments

6.1 Experiment Settings

To obtain a close-to-reality measurement, we select the subjects of our experiments from a diversified range of applications. The first subject is a model of the general Inter-ORB Protocol (GIOP) from the area of software engineering. GIOP is a key component of the Object Management Group (OMG)'s Common Object Request Broker Architecture (CORBA) specification [14]. The second model is a model of the Needham-Schroeder public key protocol from the area of computer security. The Needham-Schroeder public key protocol intends to authenticate two parties involving with a communication channel. Finally, our third subject is a model of a fuel system from the area of control system. The model is translated by Joseph [13] from a classic fuel system example in Stateflow [17].

Each model has a set of linear temporal properties that specify behavior requirement for the underlying syste. We selected the most representative property for each model to use in the experiments. For the GIOP model, the property models the behaviors of a recipient during communication. The LTL property for the Needham-Shroeder public-key protocol is a liveness property requiring that an initiator can only send messages after a responder is up and running. Finally, the properties for the fuel system checks that under abnormal conditions, the system's fault tolerant mechanism functions properly.

Table 1 provides an overview of the models and properties, showing the size of both the models and properties in terms of the number of branches, ASCs, states/transitions of the LTL property equivalent Büchi automata, and atomic propositions in the properties. All of the information in Table 1 are of relevance to the diversified profiles of test criteria we used in the experiments for the comparison, in terms of the size of test suites generated.

Table 1 Overview of the models and properties used in the experiments

Models	Branches	ASCs	States	Transitions	Atoms
GIOP	70	2	2	6	4
Needham	43	2	2	6	3
Fuel	55	3	4	21	4

For performance comparison, we select several traditional as well as specification-based testing criteria. Based on the coverage for outcomes of a logic expression (c.f. [12]), branch coverage (BC) is one of the most commonly-used structural test criteria. We include both transition and state variations of strong coverage criteria (SC/strong, TC/strong) and weak coverage criteria (SC/weak, TC/weak) for Büchi automaton [21, 23, 31]. We also include a property-coverage criterion (PC) for Linear Temporal Logic (LTL) [22]. In our experiment, the performances of these criteria, and two ASC coverage criteria (ACC/stong and ACC/weak) are compared with each other.

6.2 Methodologies

To assess the performance of the proposed criteria, we perform an extended computational study using two different methodolgies: one uses the cross-coverage percentage as a measurement indicator, and the other adopts the fault-injection technique to analyse the sensitivity of the test cases towards manually injected errors.

[Cross-coverage analysis] The cross-coverage measures how well a test suite generated for a test criterion covers another test criterion. The cross coverage is used as an indicator for the semantic strength of a test criterion with respect to others. In [28] we developed a tool to compare the effectiveness of test criteria that are used in model-checking-assisted test case generation. This experiment uses an extension of the tool that also supports the proposed ASC criteria. We use GOAL [25] to perform graph transformation required for building ASC-E-GBAs and ASC-M-GBAs. We use SPIN [10] as the underlying model checker to assist test case generation. Figure 3 shows the workflow of model-checking-assisted test case generation under ASC coverage criteria for Büchi automaton. More details of this procedure and an earlier computational study that covers more traditional testing criteria can be found in [29], with a different set of sample models.

[Fault-injection-based sensitivity analysis] Fault-injection technique (c.f. [27]) is a classic technique used in software engineering for evaluating the sensitivity of a quality assurance tool towards injected faults. For this part, faults are systematically introduced into a system, and the effectiveness of a test suite is measured by its ability of catching these artificially injected errors. More faults being caught indicates that the underlying test criterion is more sensitive in detecting faults. The fault-injection process is achieved by mutating relational operators (e.g. changing $\geq$ to $<$) within

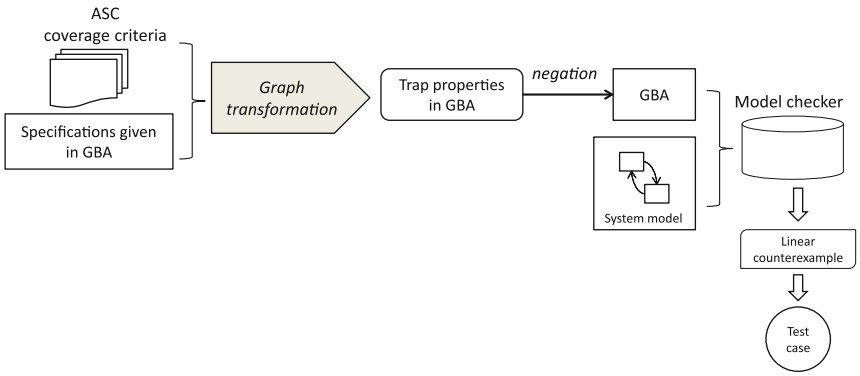


Fig. 3 The workflow of model-checking-assisted test case generation under ASC coverage criteria for Büchi automaton

the system model one operator at a time. The faulty model is then used to run a test suite generated for a test criterion. If the execution of the faulty model under a test case exhibit different behaviors from that of the original model, then the injected fault is caught by the test case. In another word, the test criterion is sensitive enough to detect the fault.

6.3 Experiment Results and Analysis

Table 2 shows the measurement of the test cases generated from the aforementioned variety of coverage criteria. We note that for the branch coverage (BC, or BC/All), tests were first specifically generated for every branch of the model. We used the coverage information to further select two more groups of test cases. We applied an Integer Linear Programming solver to obtain an optimal test suite that consists of the least number of test cases and covers the maximum number of branches that can be covered (BC/Opt.). This test suite represents the theoretical lower bound of the number of test cases needed for covering the system model under BC. The other test suite (BC/Grd.) is selected under a greedy algorithm. For example, if the first test case covers branches No. 2 and 3, then test cases for the second and third branches shall no longer be generated, and so on. The greedy algorithm represents the common practice of selecting a near-optimal test suite, to reduce the cost of test execution. "TS Size" in Table 2 indicates the number of test cases each test suite has, and "Max./Min./Avg. Length" specifies the length of the lasso-shaped test cases, i.e., the number of steps in the counterexample trace produced by the model checker. Finally, "Gen. Time" and "Exec. Time" represent the time it took to generate the traces and execute the test cases in milliseconds, respectively.

It shall be noted that for practical purpose, we enforce a time limit for the model checking process. This is due to the fact that SPIN suffers from "state space explosion

Table 2 Test suites overview

	BC			PC	SC		TC		ACC	
	All	Opt.	Grd.		Strong	Weak	Strong	Weak	Strong	Weak
GIOP										
TS Size	54	4	10	3	2	2	6	6	2	1
Max. Length	779	779	779	601	602	602	602	602	602	601
Min. Length	34	605	49	280	602	572	602	572	470	601
Avg. Length	405	664	534	494	602	587	602	588	536	601
Gen. Time	1087	27	51	332	0.02	3.7	0.06	13	13.4	300
Exec. Time	0.586	0.05	0.1	0.06	0.03	0.05	0.11	0.13	0.02	0.01
Needham protocol										
TS Size	37	7	13	3	2	2	6	6	2	2
Max. Length	70	70	62	34	43	42	43	42	43	41
Min. Length	9	22	22	33	41	41	41	41	41	41
Avg. Length	43	51	48	34	42	42	42	41	42	41
Gen. Time	360	0.065	0.122	0.03	0.02	0.02	0.06	0.06	0.02	0.02
Exec. Time	0.335	0.063	0.118	0.03	0.02	0.02	0.06	0.06	0.02	0.02
Fuel system										
TS Size	45	1	8	3	2	2	6	7	2	2
Max. Length	52,904	52,904	9261	8594	8482	538	8482	5320	9362	148
Min. Length	27	52,904	29	130	1530	254	174	192	1530	130
Avg. Length	3985	52,904	1975	4239	5006	396	4661	2003	5446	139
Gen. Time	602	0.76	0.155	0.178	600	600	780	720	300	300
Exec. Time	751	150	0.375	0.379	0.24	0.02	0.61	0.3	0.11	0.02

problem" as an explicit state model checker [7]. SPIN may run out of resources (time and/or space) before reaching a conclusive result. Subsequently, we expect three possible outcomes of the model checking process: (1) returning with a counterexample trace, (2) returning with an answer that there is no counterexample or (3) terminating without returning value. For the third case, we count the time limit towards the generation time, which explains why some entries in Table 2 takes significantly longer time than the other criteria. A specific complication involved with ACC/strong is that, as we can see from Definition 9, the construction of ASC-E-GBA essentially produces several copies of the original automaton, and the number of copies equals the size of the ASC plus one. Hence, when there are more than two states in the ASC, the ASC-E-GBA becomes too large in size, as well as too complicated in its acceptance condition. In this case, the ASC-E-GBA is too complex to be handled by GOAL, which was unable to produce the equivalent never-claims for SPIN. For the purpose of simplicity, we treated this situation the same as when SPIN could not terminate with results in our experiment.

Table 3 Cross-coverage comparison results

		BC	PC	SC		TC		ACC	
				Strong	Weak	Strong	Weak	Strong	Weak
GIOP									
BC		(77 %)	75 %	100 %	100 %	100 %	100 %	100 %	100 %
PC		66 %	(75 %)	100 %	100 %	100 %	100 %	100 %	100 %
SC	Strong	66 %	75 %	(100 %)	100 %	100 %	100 %	100 %	100 %
	Weak	66 %	75 %	100 %	(100 %)	100 %	100 %	100 %	100 %
TC	Strong	66 %	75 %	100 %	100 %	(100 %)	100 %	100 %	100 %
	Weak	66 %	75 %	100 %	100 %	100 %	(100 %)	100 %	100 %
ACC	Strong	66 %	75 %	100 %	100 %	100 %	100 %	(100 %)	100 %
	Weak	66 %	75 %	100 %	100 %	100 %	100 %	100 %	(50 %)
Needham protocol									
BC		(86 %)	100 %	100 %	100 %	100 %	100 %	100 %	100 %
PC		47 %	(100 %)	100 %	100 %	100 %	100 %	100 %	100 %
SC	Strong	47 %	100 %	(100 %)	100 %	100 %	100 %	100 %	100 %
	Weak	28 %	0 %	0 %	(100 %)	0 %	100 %	0 %	100 %
TC	Strong	47 %	100 %	100 %	100 %	(100 %)	100 %	100 %	100 %
	Weak	40 %	0 %	0 %	100 %	0 %	(100 %)	0 %	100 %
ACC	Strong	47 %	100 %	100 %	100 %	100 %	100 %	(100 %)	100 %
	Weak	30 %	0 %	0 %	100 %	0 %	100 %	0 %	(100 %)
Fuel system									
BC		(82 %)	25 %	75 %	50 %	86 %	33 %	67 %	67 %
PC		78 %	(100 %)	50 %	50 %	29 %	33 %	67 %	67 %
SC	Strong	75 %	100 %	(50 %)	50 %	29 %	33 %	67 %	67 %
	Weak	64 %	25 %	75 %	(50 %)	86 %	33 %	67 %	67 %
TC	Strong	75 %	100 %	100 %	50 %	(29 %)	33 %	67 %	67 %
	Weak	67 %	25 %	75 %	50 %	86 %	(33 %)	67 %	67 %
ACC	Strong	75 %	100 %	50 %	50 %	29 %	33 %	(67 %)	67 %
	Weak	55 %	25 %	75 %	50 %	86 %	33 %	67 %	(67 %)

Table 3 shows the results from the cross-coverage analysis. The number in each cell indicates the coverage of test cases generated for the criterion on the row w.r.t. the criterion on the column. Numbers on diagonal cells (marked with parentheses) represent the coverage of a test suite generated for the same criterion. A less-than perfect coverage on these diagonal cells indicates any of the following causes: (1) it indicates potential deficiency of a model and/or a requirement or (2) the model checker could not terminate within the time limit. For instance, the test suite of the fuel system model for ACC/weak may only reach 67 % coverage upon all the ASCs because SPIN was unable to return with a conclusive answer. As for ACC/strong for the same model, the same percentage was caused by GOAL unable to produce the equivalent never-claims for SPIN due to the ASC-E-GBA being too complex.

The results show that our proposed ASC coverage criteria, especially the strong variants, have solid and competent performances. They perform on par with the other Büchi automaton based criteria, and fall only barely behind branch coverage criterion. It shall be noted that the test suite generated for the branch coverage criterion is much larger than those generated for the property-based test criteria, including our ASC criteria, indicating that the property-based criteria can potentially make testing more effective by producing smaller and more focused test suites, as shown in Table 2. A smaller test suite, along with a good performance in cross-coverage analysis, makes the new criteria competitive alternatives to a white-box coverage criterion such as the branch coverage criterion.

It shall also be noted that the test suites for ASC coverage criteria, although competent, did not achieve full branch coverage. This is because we only use one temporal property for each model, and the property does not cover all the functional aspects of the models. For instance, the property for the GIOP model specifies the recipient's behavior at "waiting" or "receiving" modes, it does not concern other modes of operations. Therefore, the generated test suite skips some code segments, which leads to a less-than perfect branch coverage.

This observation leads to an important feature of property-based test criteria, including our ASC coverage criteria. That is, the performance of these criteria are heavily influenced by the quality of underlying requirement. A thorough requirement touching more aspects of a model may result in a test suite with better quality. In Sect. 5 we capitalized this observation via our ASC-induced property refinement. Alternatively, a more complete set of temporal properties that address multiple aspects of a model could also greatly improve the performance on this part.

Last but not least, the results above also establish that ASC coverage criteria correlate nicely with the state and transition coverage criteria. The strong variant performs exactly the same as the state and transition coverage, while the weak variant exhibits the results that are somewhat in between. Superfluously, an ASC being covered indicates that the states and transitions on the path are also covered.

Such correlation proves that we are able to strip the syntax dependency away even more thoroughly, compared with the syntax dependency that still exists for the property coverage criterion in [22]. At the same time, an ASC is also not merely an extension of states and transitions. The traces covering the ASC need to satisfy the "infinite visit" condition upon the acceptance states. Hence, it comes one step closer to the semantic essence of the temporal properties. In some cases, this makes the ASC more challenging to cover. When it does happen, such as in the case of ACC/weak for the fuel system model, it only has 55 % of coverage over the branches, lower than both SC/weak and TC/weak. On the other hand, both ACC/strong and ACC/weak tend to yield smaller size of test suites, while simultaneously have a better grasp on the semantic essence. This also means the refinement process described in Sect. 5 could result in finer tuned refined GBA that other criteria are unable to produce.

Table 4 shows the results of fault-injection-based sensitivity analysis. Faults are injected by mutating relational operators in the models. The count of such operators are specified in the parenthesis along side the name of the model at the top row of the

Table 4 Injected faults detection results

Total faults	GIOP (49)		Needham protocol (24)		Fuel system (191)	
	Detection rate (%)	SAC	Detection rate (%)	SAC	Detection rate (%)	SAC
BC	76	28,776	92	1729	66	118,355
BC(Opt.)	76	3495	79	452	N/A	N/A
BC(Grd.)	76	7026	88	709	66	23,939
PC	67	2212	75	136	76	16,733
SC/strong	67	1797	83	101	69	14,510
SC/weak	67	1752	25	336	54	1467
TC/strong	67	5391	83	304	69	40,530
TC/weak	67	5266	38	647	58	24,174
ACC/strong	73	1468	83	101	69	15,785
ACC/weak	67	897	25	328	44	632

table. The rest of the Table 4 lists out the percentage of the faults that were detected by the test suites we generated based on the different test coverage criteria.

We define a *Sensitivity Adjusted Cost* (SAC) for cost/benefit analysis:

$$SAC = \frac{\text{(Total Length of the Test Suite)}}{\text{(Percentage of Detected Faults)}}$$

Note that the cost of executing a test suite is in general proportional to the size of the test suite. The SAC essentially indicates the adjusted cost of test (execution) w.r.t. the sensitivity of the underlying test criterion, and the lower the cost is the better.

In all three models, the property-based criteria, including our ASC based coverage criteria, are able to detect a good portion of the injected faults. Comparing with branch coverage, it is to be expected that BC would have the best detection rate due to its code-based nature. For the fuel system model, however, some of the test cases are excessively long that they are not executable (the longest one exceeding 50,000, see Table 2). Both the full and greedy test suites consequently can only detect two thirds of the faults, while other test suites catch up or even surpass it with fewer and shorter test cases, as indicated by the SAC values.

While comparing with other property-based criteria (PC, SC and TC), ACC-generated test suites benefit from their smaller sizes, and their SAC values are either the lowest or very close. In particular, ACC out-performs both SC and TC on the GIOP model with both higher detection rate and much lower SAC values. While on the other two models, ACC also at least performs on par with SC and TC with competitive SAC values. It shows that among the GBA based criteria, ACC also demonstrates stronger performances.

In all three models, strong variants of Büchi-automaton-based coverage criteria outperform the related weak variants, and by a large margin in some cases (e.g.

Needham Protocol). In theory the strong variants subsume their counterparts in weak variants. In practice, the strong variants of the criteria unveil more subtle features of temporal requirements, often resulting in longer test cases. These longer test cases help find faults deeply buried in models.

7 Conclusions

We proposed a specification-based approach for testing reactive systems with requirement expressed in Büchi automata. At the core of our approach are two variants of property coverage metrics and criteria measuring how well a test suite covers the acceptance condition of a Büchi automaton. By covering the acceptance condition, which is the hallmark of a Büchi automaton defining infinite words, these metrics relate test cases to temporal patterns of infinite executions. This makes testing more effective in debugging infinite executions of the system. To provide a complete tool chain for requirement-based testing with Büchi automaton, we developed a test case generation algorithm for the proposed criteria. The algorithm utilizes the counterexample generation capability of an off-the-shelf model checker to automate the test case generation.

It shall be noted that, although specification-based testing with automata has been studied before (c.f. [4]), the specification concerned in most of these previous works is a system design modeled in a finite automaton. In comparison, we focus on behavioral requirements modeled in Büchi automaton. Moreover, existing approaches for specification-based testing for reactive systems [18, 21, 23, 24, 31] focus on the finite prefixes of its infinite executions. In contrast, our approach works with temporal patterns of its infinite executions. All of these make our approach more advanced and effective in testing the temporal patterns of a reactive system.

Our approach tests the conformance of a reactive system to its requirement in Büchi automaton. It may be used for revealing the deficiency of the system as well as its requirement. We discussed how our approach may be used to debug and even refine the requirement, using the information from the model-checking-assisted test case generation. We proposed a property-refinement algorithm that automated the process of property refinement.

To assess the effectiveness of our approach, we carried out an extended computational study using two methodologies: a cross-coverage measurement among multiple test criteria, and a fault-injection-based sensitivity analysis. Subjects for study are selected from a diversified range of fields. First, we use a cross-coverage metric to measure relative effectiveness of test criteria against each other. Then, we use fault-injection technique to measure how well test suites generated from the proposed criteria can detect faults planted in models. The experimental results indicate that our criteria exhibit competent performance over existing test criteria. These criteria are particularly effective at reducing the size of test suites, making testing more targeted and efficient. For the future work, we want to extend our approach to more complex requirements, such as those in μ-calculus.

References

1. Bieman, J., Dreilinger, D., Lin, L.: Using fault injection to increase software test coverage. In: Proceedings of Seventh International Symposium on Software Reliability Engineering, 1996, pp. 166–174 (1996). doi:10.1109/ISSRE.1996.558776
2. Dahl, O.J., Dijkstra, E.W., Hoare, C.: Structured Programming. A.P.I.C. Studies in Data Processing, vol. 8. Academic Press (1972)
3. Fraser, G., Gargantini, A.: An evaluation of model checkers for specification based test case generation. In: ICST'09: Proceedings of the 2009 International Conference on Software Testing Verification and Validation. IEEE Computer Society, Washington, DC, USA (2009)
4. Fujiwara, S., von Bochmann, G., Khendek, F., Amalou, M., Ghedamsi, A.: Test selection based on finite state models. IEEE Trans. Softw. Eng. **17**(6), 591–603 (1991)
5. Gaudel, M.C.: Software testing based on formal specification. In: Borba, P., Cavalcanti, A., Sampaio, A., Woodcock, J. (eds.) Testing Techniques in Software Engineering, Second Pernambuco Summer School on Software Engineering, PSSE 2007, 3–7 Dec 2007, Revised Lectures. Lecture Notes in Computer Science, vol. 6153, pp. 215–242. Springer, Berlin (2010)
6. Gaudel, M.C.: Checking models, proving programs, and testing systems. In: Gogolla, M., Wolff, B. (eds.) TAP 2011 Proceedings. Lecture Notes in Computer Science, vol. 6706, pp. 1–13. Springer, Berlin (2011)
7. Gerth, R., Peled, D., Vardi, M.Y., Wolper, P.: Simple on-the-fly automatic verification of linear temporal logic. In: Protocol Specification Testing and Verification. Chapman & Hall (1995)
8. Hartman, A., Nagin, K.: The agedis tools for model based testing. In: ISSTA'04: Proceedings of the ACM/SIGSOFT International Symposium on Software Testing and Analysis. ACM (2004)
9. Hierons, R.M., Bogdanov, K., Bowen, J.P., Cleaveland, R., Derrick, J., Dick, J., Gheorghe, M., Harman, M., Kapoor, K., Krause, P., Lüttgen, G., Simons, A.J.H., Vilkomir, S., Woodward, M.R., Zedan, H.: Using formal specifications to support testing. ACM Comput. Surv. **41**(2), 9:1–9:76 (2009)
10. Holzmann, G.J.: The model checker SPIN. IEEE Trans. Softw. Eng. **23**, 279 (1997)
11. Hong, H.S., Lee, I., Sokolsky, O., Ural, H.: A temporal logic based theory of test coverage and generation. In: TACAS'02 (2002)
12. Jorgensen, P.C.: Software Testing: A Craftsman's Approach, 1st edn. CRC Press Inc., Boca Raton, FL, USA (1995)
13. Joseph, S.: Fault-Injection through Model Checking via Naive Assumptions about State Machine Synchrony Semantics. Master's thesis, West Virginia University, Morgantown, West Virginia (1998)
14. Kamel, M., Leue, S.: Formalization and validation of the general inter-ORB protocol (GIOP) using PROMELA and SPIN. Int. J. Softw. Tools Technol. Transf. (STTT) **2**(4), 394–409 (2000)
15. Knight, J.: Safety critical systems: challenges and directions. In: Proceedings of the 24rd International Conference on Software Engineering. ICSE 2002, pp. 547–550 (2002)
16. MathWorks: Simulink design verifier (2015). http://www.mathworks.com/products/sldesignverifier/
17. MathWorks Inc.: Stateflow examples (2015). http://www.mathworks.com/help/stateflow/examples.html
18. Meinke, K., Sindhu, M.A.: Incremental learning-based testing for reactive systems. In: Tests and Proofs, pp. 134–151. Springer (2011)
19. Platzer, A., Quesel, J.D.: European train control system: a case study in formal verification. In: Breitman, K., Cavalcanti, A. (eds.) Formal Methods and Software Engineering. Lecture Notes in Computer Science, vol. 5885, pp. 246–265. Springer, Berlin (2009)
20. Reactive Systems Inc.: Testing and validation of simulink models with reactis (2010). http://www.reactive-systems.com/
21. Tan, L.: State coverage metrics for specification-based testing with Büchi automata. In: 5th International Conference on Tests and Proofs, Lecture Notes in Computer Science. Springer, Zurich, Switzerland (2011)

22. Tan, L., Sokolsky, O., Lee, I.: Specification-based Testing with Linear Temporal Logic. In: the proceedings of IEEE Internation Conference on Information Reuse and Integration (IRI'04). IEEE Society (2004)
23. Tan, L., Zeng, B.: Specification-based testing with Buchi automata: transition coverage criteria and property refinement. In: International Conference on Information Reuse and Integration. IEEE (2014)
24. Tretmans, J.: Model based testing with labelled transition systems. In: Formal methods and testing, pp. 1–38. Springer (2008)
25. Tsay, Y.K., Chen, Y.F., Tsai, M.H., Wu, K.N., Chan, W.C.: GOAL: A Graphical Tool for Manipulating Büchi Automata and Temporal Formulae. In: 13th Tools and Algorithms for the Construction and Analysis of Systems, vol. 02, pp. 466–471. Springer (2007)
26. Yoo, J., Jee, E., Cha, S.: Formal modeling and verification of safety-critical software. IEEE Softw. **26**(3), 42–49 (2009)
27. Young, M., Pezze, M.: Software Testing and Analysis: Process, Principles and Techniques. Wiley (2005)
28. Zeng, B., Tan, L.: Test criteria for model-checking-assisted test case generation: a computational study. In: International Conference on Information Reuse and Integration. IEEE (2012)
29. Zeng, B., Tan, L.: A unified framework for evaluating test criteria in model-checking-assisted test case generation. Inf. Syst. Front. **16**(5), 823–834 (2014)
30. Zeng, B., Tan, L.: Test reactive systems with buchi automata: acceptance condition coverage criteria and performance evaluation. In: 2015 IEEE International Conference on Information Reuse and Integration, IRI 2015, San Francisco, CA, USA, August 13–15, pp. 380–387 (2015)
31. Zeng, B., Tan, L.: Testing with buchi automata: transition coverage metrics, performance analysis, and property refinement. Advances in Intelligent Systems and Computing (2015)

Capturing and Verifying Dynamic Systems Behavior Using UML and π-Calculus

Aissam Belghiat, Allaoua Chaoui and Mokhtar Beldjehem

Abstract UML is a semi-formal modeling language for object oriented systems. It is widely accepted and its applications become more and more widespread in real word projects. The UML diagrams suffer from lack of formal semantics which hinders their automatic verification. It is actually a problem that always arises. Formal methods can be used to overcome it. π-calculus is a flexible formal theory with several applications. It offers a rich theory and tools for verification purposes. Thus, this paper presents an approach for capturing and verifying the dynamic behavior of systems using UML diagrams and π-calculus. We illustrate our approach by an example in order to explain it. Then we tackle another small example to show the verification capabilities provided by the approach. An implementation of the approach is presented.

Keywords UML · π-calculus · Formalization · Dynamic behavior · Verification

1 Introduction

UML (Unified Modeling Language) is a semi-formal language to visualize, specify, build and document all the artifacts of object-oriented software systems [1]. It is adopted as a standard in software development by the industry body OMG (Object Management Group). It provides multiple graphical notations to describe static and dynamic aspects of object-oriented software systems as well as different levels of

A. Belghiat (✉) · A. Chaoui
MISC Laboratory, Department of Computer Science, University of Mentouri,
25000 Constantine, Algeria
e-mail: belghiatissam@gmail.com

A. Chaoui
e-mail: allaoua.chaoui@univ-constantine2.dz

M. Beldjehem
University of Ottawa, Ottawa, Canada
e-mail: mbeldjeh@uOttawa.ca

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_3

detail (e.g. design vs. implementation). This makes it suitable to assist in all phases of the software development process and consequently there are several CASE tools and workbenches that have been emerged which support this language. Despite of all these advantages, the problem of imprecise semantics of this language still hinders all the verification tasks. Thus, the formalization of UML diagrams using formal methods for verification purposes has been adopted largely in order to deal with this problem.

The π-calculus [2] is a flexible formal language with several applications especially for concurrent and distributed systems. It can be used to rigorously specify and verify these systems. It provides a rich theory and tools which can be used to enhance the development process by detecting errors in early phases, and thereby reducing the cost of software development and maintenance while ensuring their correctness and reliability.

UML provides interaction diagrams to represent the communications with and within the software. There are two common variants of interaction diagrams; the sequence diagram and the communication diagram. Whereas the sequence diagram shows temporal representation of the interactions between the objects and the chronology of the exchanged messages between the objects and with the actors, the communication diagram displays a spatial representation of the objects and their interactions. Little research effort has been devoted at tackling the formalization of UML communication diagrams; due the fact that large numbers of designers claim that the other UML interaction diagram (i.e. the sequence diagram) is more appropriate in the modeling task. Unfortunately, this is not always true because the UML specification [1] tells us that each type of the proposed diagrams provides slightly different capabilities that make it more appropriate and adapted for certain situations. Furthermore, communication diagrams are more suitable [3, 4] and often used to provide a glance-view of a collection of collaborating objects, in particular within a real-time environment, offer an alternate view of interaction with sequence diagrams, add functionality to classes by exploring the behavior results from the interaction of its objects, model the implementation logic of a complex operation; in particular when it interacts with several other objects, and to describe the roles taken by objects in a system, and the different relationships involved in those roles.

UML provides also class diagrams. A class diagram defines the static structure and types of objects and methods. Although class diagrams are not invented to represent the dynamic behavior of systems, their use provides multiple advantages (e.g. placing some basic information in them...etc). This allows verifying the compatibility of lifelines (objects) and messages (method call) of communication diagram with their definition in the class diagram. Links (associations), parameter types, return values must all be correctly defined.

In this paper (which is an extension of our previous work [5]), we have proposed to translate UML models (class and communication diagrams) to the π-calculus. We focus on the communication diagrams because we are more interested in modeling and verifying the dynamic aspects of software systems. To this goal, we examine the graphical syntax of such diagrams, which is precisely specified as well as the semantic that is imprecisely defined. Then we try to develop an incremental semantic

correspondence between UML diagrams and the π-calculus using the abilities of this later in capturing the way in which the objects interact [2]. A π-calculus tool (Mobility Workbench MWB [6, 7], in our case) is then used to verify these models. A tool suite is developed in order to maximize the potential impact of our approach.

The rest of the paper is structured as follows. In Sect. 2, we present related work. In Sect. 3, we present basic notions about UML diagrams and the π-calculus. In Sect. 4, we propose a formalization of UML diagrams using the π-calculus. In Sect. 5, we illustrate our approach through an example. In Sect. 6, an example of using the approach in verification is provided. In Sect. 7, a tool suite is presented. In Sect. 8, our work is discussed. Section 9 concludes the work by remarks and future work.

2 Related Work

In the literature, there is a considerable body of work on formalization of UML diagrams using formal methods. But just a few work has addressed directly UML communication diagrams. Lano et al. [8] have formalized collaboration diagrams using Structured Temporal Theories in an effort to describe semantics for a subset of UML diagrams. Övergaard, in [9], developed a sequence-based formalization of collaboration diagrams in terms of roles and interactions. In [10] a Colored Petri Nets-based approach is proposed to represent collaboration diagrams. In [3] an integrated approach graph transformation rules and graph processes is used to formalize collaboration diagrams. In [11] the authors use Object Petri Net Models to formalize UML statechart and collaboration diagrams for analysis purposes. In [12] the authors propose an approach for integrating UML statechart and collaboration diagrams by their formalization using Hierarchical Predicate Transition Nets (HPrTNs). In [13] a graph transformation based approach is developed for the automatic generation of Colored Petri Net Models from UML statechart and collaboration diagrams. Merah et al. [14] translate UML2 communication diagrams to Buchi automata using the ATL transformation language. In [15] the authors transform the communication diagrams of Fuzzy UML [16] (which is a modeling language that combines the UML with Fuzzy logic) to Fuzzy Petri nets. In [17], the authors translate concurrent UML models into Maude formal specification for model checking purposes. In [18], the authors provide a graph transformation based approach for model checking UML diagrams and generate the code from them.

With regard to previous studies, we notice the following concerns:

- The works in [10–13, 17, 18] have not addressed directly the formalization of communication diagrams, but as part of their contributions to attain other objectives.
- The authors in [14] propose a non-persistent mapping which neglects the most essential features those that reflect the behavioral-semantics of communication diagrams such as asynchronous communication, conditional messages, concurrent messages and concurrent loops. Thus, the approach proposed by [14] is very limited and does not fully conform to the semantics of UML.

In contrast to all these works, our contribution provides multiple benefits over them:

- Our study is a first attempt in regard to the formalization of UML communication diagrams using the π-calculus. The target semantic domain chosen in our translation provides a rich theory and tools, which allow and automate formal analysis and verification of communication diagrams such as model checking and equivalence checking.
- Our study provides a full formal definition of the semantic mapping between UML communication diagrams and the π-calculus, especially in contrast to [3, 14], which will allow easily the automation of the translation for rigorous analysis tasks.
- We provide an exhaustive approach in our formalization (Unlike in [14]), so that all systems modeled in such diagrams can be perfectly described in our process algebras.
- Our approach covered the aspect of using collaboration diagrams invented in [3], i.e., the specification of system's state transformation and this is the reason why we have omitted state machine diagrams (see the discussion section for further details).

3 Background

3.1 UML Models

3.1.1 UML Class Diagrams

A class diagram is used to model the internal static structure of systems. It contains classes and relationships. A class contains attributes and operations. Relationships including associations, aggregations and generalizations relate classes to each other. Figure 1 shows a simple class diagram that gathers the basic elements, i.e. class, association, attributes, operations...etc.

3.1.2 UML Communication Diagrams (CDs)

A communication diagram (collaboration diagrams in UML 1.x) is one of interaction diagrams that display a spatial representation of the objects and their interactions. We present in this section the syntax and semantics of these diagrams.

Fig. 1 Basic elements of UML class diagrams

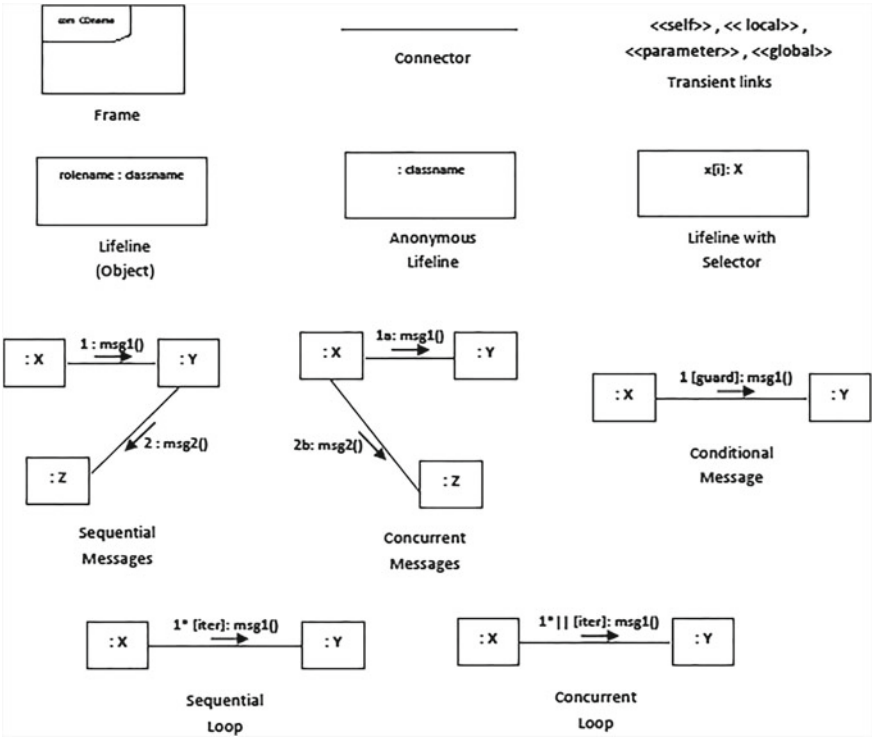


Fig. 2 Structural elements of UML CDs

Structural elements of UML CDs

We present above the notational elements of UML communication diagrams [1, 19] in Fig. 2 and we show the different combinations of these elements that are used to build the diagrams.

Other visual stereotypes symbols of the robustness diagram can be considered since they are used to improve the readability of the communication diagrams [4]. Figure 3 depicts these symbols.

Semantics of UML CDs

A Communication Diagram shows the interactions through an architectural view where the arcs between the communicating lifelines are decorated with description of the exchanged messages and their sequencing [1].

In Fig. 2, several constituents of a communication diagram are depicted. In fact, it is often described within a frame. It contains multiple lifelines (objects) which are related by means of connectors and which interact using messages exchanging. A lifeline can be anonymous (has no name), and with/without selector (the lifeline is selected with selector). A message represents the entity of interaction. It can be conditional message (with a guard condition), with sequential loop (one by one iteration), and with/without concurrent loop (parallel iteration). Furthermore, we can

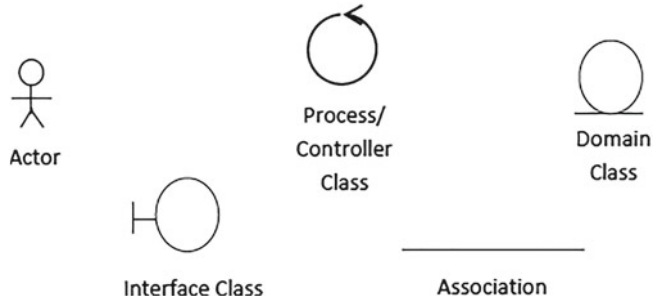


Fig. 3 Visual stereotypes symbols used in UML CDs

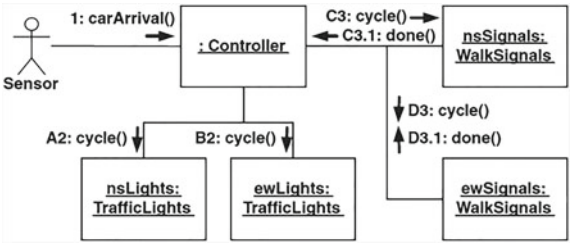


Fig. 4 An instance-level UML communication diagram

have sequential messages (one by one) or concurrent messages (in parallel) in the communication diagram.

With regard to the visual stereotypes symbols in Fig. 3, an actor represents all systems with which the modeled system interacts. Process/Controller classes implement logic which corresponds to multiple business entities. Domain classes implement basic business entities. Interface classes allow actors to interact with the system described via an interface. An association is revealed whenever an actor interacts with a class, or two classes interact.

Communication diagrams can be used on two different levels [4]; Instance-level UML communication diagrams and Specification-Level Diagrams.

- Instance-level UML communication diagrams: they are the most common used style of UML communication diagram. They display the interactions between instances (objects). They are usually created to describe and explore the internal design of object-oriented system. We focus on these diagrams since they provide both structural and interaction aspects of systems. Figure 4 shows an example of such diagrams.
- Specification-level Diagrams: they are not the common used style of UML communication diagram due to the suitability of UML class diagrams which are extensively used by modelers to identify the roles. They are typically used to describe and explore the roles that domain classes take in a system. Figure 5 shows an example of such diagrams.

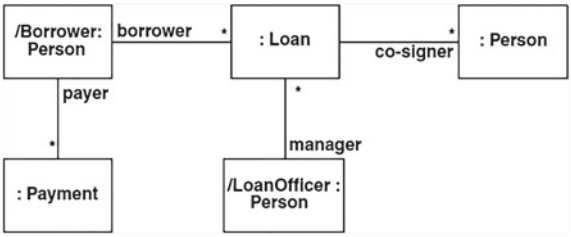


Fig. 5 A specification-level UML communication diagram

3.2 π-calculus

The π-calculus [2] has been introduced as a new and fundamental way of thinking about concurrent interactive processes, and one which is amenable to rigorous treatment [2]. It is a process algebra developed to cover the limitation of the process calculus CCS (Calculus of Communicating Systems) in terms of expression power by authorizing the passage of "channels" between processes; it can be used for the representation, the analysis, the verification and simulation of concurrent systems. The abstract syntax for the π-calculus is built from the following BNF grammar (x and y are any names in the set of names N) [2]:

$\mathbf{P} ::= \mathbf{0}$	Nil; empty process
$\mid \mathbf{x}(\mathbf{y}) \cdot \mathbf{P}$	Input prefix; receive y along x
$\mid \mathbf{x} < \mathbf{y} > \cdot \mathbf{P}$	Output prefix; send y along x
$\mid \tau \cdot \mathbf{P}$	Silent prefix; an internal action
$\mid \mathbf{P} \mid \mathbf{P}$	Parallel composition
$\mid \mathbf{P} + \mathbf{P}$	non-deterministic choice
$\mid (\nu \mathbf{x})\, \mathbf{P}$	Restriction of name x to process P
$\mid !\mathbf{P}$	Replication of process P
$\mid [\mathbf{x} = \mathbf{y}]\, \mathbf{P}$	Match; if x = y then P
$\mid [\mathbf{x} \neq \mathbf{y}]\, \mathbf{P}$	Mismatch; if x $\neq$ y then P
$\mid \mathbf{A}(\mathbf{y}_1, \ldots, \mathbf{y}_n)$	Process Identifier

There are several extensions of the π-calculus, in our paper we choose the polyadic version that extends the monadic π-calculus in which a message consists of multiple names rather than one. This is because with this version we can demonstrate and illustrate sufficiently our formalization approach.

For the convenience, we define the following shortcuts [20]: ① to represent the summation of all processes, ② to represent the composition of all processes, ③ to represent a series of channels and ④ the restriction operator for multiple names in a process as follow:

$$\sum_{i\in I} P_i \overset{def}{=} \mathbf{P_1} + \mathbf{P_2} + ... + \mathbf{P_n} \quad ...①$$

$$\prod_{i\in I} P_i \overset{def}{=} \mathbf{P_1}|\mathbf{P_2}|...|\mathbf{P_n} \quad ...②$$

$$\overrightarrow{\mathbf{x}}_1 \overset{def}{=} \mathbf{x_1}, \mathbf{x_2},..., \mathbf{x_n} \quad ...③$$

$$(\nu\ \mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x}_n)\ \boldsymbol{P} \overset{def}{=} (\nu\ \mathbf{x}_1)(\nu\ \mathbf{x_2}) \ldots (\nu\ \mathbf{x_n})\ \boldsymbol{P} \quad ...④$$

4 The Proposed Approach

Our approach consists in providing a formal mapping of the elements of UML models (class and communication diagrams) into the π-calculus. As our driving type of diagrams is the communication diagram, we focus on it, but without forgot the other interesting diagram. The architecture of our approach is presented in Fig. 6.

We start with the translation of UML class diagram. In fact, since the class diagram only represents the static view of a system, we don't proceed to a full formal description of its mapping. We only show informally how it is going and we focus on essential elements. The systems behavior is what interests us here, so we provide a full formal mapping for communication diagrams to enable analysis. We consider a class diagram because it defines the static structure and types of objects and methods. We can, for example, verify the compatibility of lifelines (objects) with their definition in the class diagram.

In order to translate a class diagram (see Table 1), we have to use a variety of π-calculus names, for example names to represent the names of classes "c_i", names to represent different attributes "a_{ij}", names to represent different methods "m_{ij}", names to represent identities of objects created "id_io_j".

A class "C_i" which is the main element of the class diagram is represented by a process $C_i(c_i, a_{i1}..., m_{i1}...)$. This last creates a new object with a new unique name "id_io_j" whenever accessed at "c_i" by executing the output action $(\nu id_io_j)\bar{c}_i <$

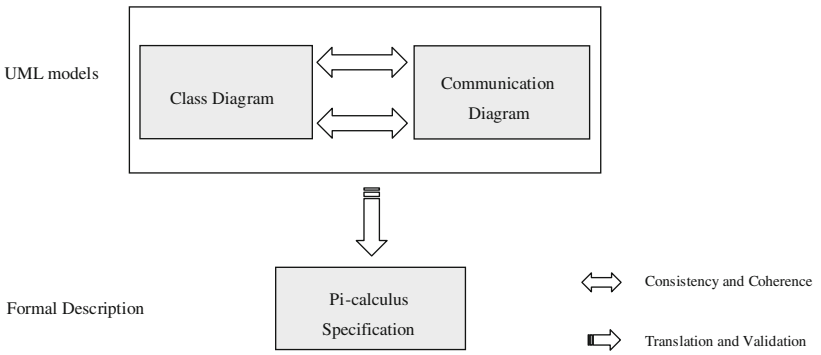


Fig. 6 Architecture of the approach

Table 1 Translation of class diagram

Class diagram	Pi-calculus	Comments
Class name c_i	c_i	Translation to a name
Attribute name a_{ij}	a_{ij}	Translation to a name
Methode name m_{ij}	m_{ij}	Translation to a name
Class C_i	$C_i(c_i, a_{i1}\ldots, m_{i1}\ldots) \stackrel{def}{=} !(\nu id_i o_j)\bar{c}_i<id_i o_j>.$ $Oj(id_i o_j, a_{i1}\ldots, m_{i1}\ldots)$	Translation to a process

$id_i o_j >$ and behaving like the process that represents the object. The attributes and methods represented as parameters "$a_{i1}\ldots$, $m_{i1}\ldots$" pass to the process that represents the object. This guarantees that each object created containing its own attributes and methods. The replication symbol "!" is used to allow creating infinity of objects.

Regarding communication diagrams, the next section, inspired by [20–22], proposes a formal definition of UML communication diagrams. First, we define a communication diagram in terms of sets and functions. Then, we start to formally defining the translation mapping between the source and target models.

4.1 Formal Definition of UML CDs

Definition 1 (*CDs definition*) We suppose the types of notational elements of communication diagrams as:

$$\textbf{Elements} = \{\textbf{Lifelines}, \textbf{Links}, \textbf{Messages}, \textbf{Conds}, \textbf{Msgs}, \textbf{Vals}, \textbf{Prms}\}$$

A communication diagram is a 8-tuple:

$$\textbf{CD} = (\textbf{CDname}, \textbf{Elements}, \boldsymbol{\alpha}_{\textbf{cond}}, \boldsymbol{\alpha}_{\textbf{msg}}, \boldsymbol{\alpha}_{\textbf{val}}, \boldsymbol{\alpha}_{\textbf{prm}}, \boldsymbol{\alpha}_{\textbf{in}}, \boldsymbol{\alpha}_{\textbf{out}})$$

where:

- **CDname** is the communication diagram name.
- **Lifelines** represents the set of lifelines
- **Links** represents the set of links
- **Messages** represents the set of messages
- **Conds** represents the set of conditions.
- **Msgs** represents the set of messages names.
- **Vals** represents the set of return values.
- **Prms** represents the set of parameters.

$-\alpha_{\mathbf{cond}}$: **Messages** $\longrightarrow$ **Conds**	Defines for a message its condition.
$-\alpha_{\mathbf{msg}}$: **Messages** $\longrightarrow$ **Msgs**	Specifies for a message its name.
$-\alpha_{\mathbf{val}}$: **Messages** $\longrightarrow$ **Vals**	Specifies for a message its return value.
$-\alpha_{\mathbf{prm}}$: **Messages** $\longrightarrow$ **Prms**	Specifies for a message its parameters.
$-\alpha_{\mathbf{in}}$: **Lifelines U Messages** $\longrightarrow$ **Links**	Relates a lifeline (resp. message) to links (considered as entering links).
$-\alpha_{\mathbf{out}}$: **Lifelines U Messages** $\longrightarrow$ **Links**	Relates a lifeline (resp. message) to links (considered as leaving links).

Definition 2 (*Process expression function*) In order to capture the semantics of communication diagrams, we define a function Ω for representing UML communication diagrams as process expressions in the π-calculus. The function Ω is defined as follows:

$$\mathbf{\Omega}_{Elements} : \mathbf{Elements} \longrightarrow \mathbf{Pi-calculus}$$

$\forall$ E $\in$ Elements, $\exists$ P $\in$ Pi-calculus, where $\Omega^{E \in Elements}$ (E) = P. Which means that each elements of the communication diagram has it's correspond process expression "P" in the pi-calculus.

Using this function, we can map each notational element of the communication diagram into the adequate π-calculus specification as process expressions.

4.2 Formalization of UML CDs

The technique adopted to formalize UML communication diagrams is to define the appropriate π-calculus representation for each of their notational elements. The task is repeated until no elements are left and a complete π-calculus specification for a communication diagram is generated. The lifelines are modeled as processes, the messages as processes and the links as connectors.

- **Rule 1**:**(lifeline "object")**

Suppose O1 $\in$ Lifelines, α_{in} (O1) = {IN_i}, α_{out} (O1) = {OUT_j}, $\Omega_{\text{Links}}(\text{IN}_\text{i}) = \text{in}_\text{i}\text{o}_1$, $\Omega_{\text{Links}}(\text{OUT}_\text{j}) = \text{out}_\text{j}\text{o}_1$, for i = 1,…, n. j = 1,…, m. f = n + m is the number of links associated with the lifeline. "seq" is a channel for evaluating the sequence number of the next message that will be sent. Ω_{Liflines} (O1) = O1($\text{in}_\text{i}\text{o}_1$, seq, $\text{out}_\text{j}\text{o}_1$). We model the semantics of a lifeline by the behavior of the parameterized process O1($\text{in}_\text{i}\text{o}_1$, seq, $\text{out}_\text{j}\text{o}_1$) as follows (While i and j represent different inputs and outputs respectively of the object):

$$\mathbf{O1(in_io_1, seq, out_jo_1)} \stackrel{def}{=} \mathbf{in_io_1}.\tau.(\nu \mathbf{x})\overline{\mathbf{seq}} < \mathbf{x} > .\mathbf{x(s)}.!\overline{\mathbf{out_jo_1}} < \mathbf{s} > .\mathbf{O1(in_io_1, seq, out_jo_1)}$$

An event that occurs in the process modeling the object is specified using the internal action "τ". We use the "seq" channel to evaluate the sequence number of the message generated in response to the event produced. The output action "$\overline{\text{seq}}$ <x>"

and the input action "x(s)" specify the sequence number of the next message. The concerned message process will be fired using the output action "$\overline{out_j o_1}$ <s>" in the process. The replication operator "!" is used to indicate that the process modeling the object will trigger multiple messages processes (towards different objects), by outputting multiple copies of the "s" channel, if they have letters on messages i.e. different threads concurrently (in parallel). The recurrence of the process $O1(in_i o_1$, seq, $out_j o_1)$ in the end of the expression is to deal with sequential messages (one by one).

- **Rule 2**:**(message)**

We take the following notation that gives us a general representation of messages and summarizes those described in Fig. 2.

```
[[<seq>] [<cond>] [* [ || ] [ [ <iter> ] ] ]:] [<var>:=] <msg> ( [<prm>] )
```

Suppose M1 ∈ Messages, $\alpha_{in}(M1) = \{IN_i\}$, $\alpha_{out}(M1) = \{OUT_j\}$, $\Omega_{Links}(IN_i) = in_i m_1$, $\Omega_{Links}(OUT_j) = out_j m_1$, for i = 1,…, n. j = 1,…, m. f = n + m is the number of links associated with the message. α_{msg} (M1) = {MSG1}, $\Omega_{Msgs}(MSG1) = msg_1$, "$msg_1$" is a channel which represents the message that will be sent.

– **A simple message**:

When we have a simple message, $\Omega_{Messages}(M1) = M1(in_i m_1, msg_1, out_j m_1)$, the semantics of the message is represented in the π-calculus as a process with parameters $M1(in_i m_1, msg_1, out_j m_1)$ as follow:

$$\mathbf{M1(in_i m_1, msg_1, out_j m_1)} \overset{def}{=} \mathbf{in_i m_1(s).\overline{out_j m_1} < msg_1 > .M1(in_i m_1, msg_1, out_j m_1)}$$

The process modeling the message waits its turn to be executed (i.e. message sending), this is represented by an input on the "$in_i m_1$" channel. The output action "$\overline{out_j m_1} < msg_1 >$" sends the message.

– **A message with return value**:

When we have a message with return value VAL1∈ VAL, α_{val} (M1) = {VAL1}, $\Omega_{Vals}(VAL1) = val_1$, $\Omega_{Messages}(M1) = M1(in_i m_1, msg_1, val_1, out_j m_1)$, the semantics of the message is represented in the π-calculus as a process with parameters $M1(in_i m_1$, msg_1, val_1, $out_j m_1)$ as follows:

$$\mathbf{M1(in_i m_1, msg_1, val_1, out_j m_1)} \overset{def}{=} \mathbf{in_i m_1(s).\overline{out_j m}_1 < msg_1, val_1 > .M1(in_i m_1, msg_1, val_1, out_i m_1)}$$

When the message has a return value, the process modeling the message outputs the "val_1" channel on the channel "$out_i m_1$" which will be used to get back the returned value.

– **A message with parameters and a return value**:

When we have a message with a return value and parameters PRM1∈ PRM, $\alpha_{prm}(M1) = \{PRM1\}$, $\Omega_{Prms}(PRM1) = prm_1$, $\Omega_{Messages}(M1) = M1(in_i m_1, msg_1, val_1, prm_1, out_j m_1)$, the semantics of the message is represented in the π-calculus as a process with parameters $M1(in_i m_1, msg_1, val_1, prm_1, out_j m_1)$ as follows:

$$\mathbf{M1(in_i m_1, msg_1, val_1, prm_1, out_j m_1)} \overset{def}{=} \mathbf{in_i m_1}(s).((\nu \mathbf{p})\overline{\mathbf{prm_1}} < \mathbf{p} > |\mathbf{p}(\overrightarrow{\mathbf{pts}})).\overline{\mathbf{out_j m_1}} < \mathbf{msg_1}, \overrightarrow{\mathbf{pts}}\ \mathbf{val_1} > .\mathbf{M1(in_i m_1, msg_1, val_1, prm_1, out_j m_1)}$$

The "prm_1" channel is used to obtain the list of parameters modeled as "$\overrightarrow{pts}$" channels. When the message has some parameters, the process modeling the message creates a channel "p" and executes the output action "$\overline{prm}_1$ <p>" and the input action "$\overrightarrow{p}$ (pts)" to retrieve the parameters. The channel "msg_1" will be thereafter sent with multiple channels which represent the parameters and the returned value.

– **If there is a sequential iteration**:

When we have a message with a sequential iteration k (k may be specified or unspecified i.e. "*"), $\Omega_{Messages}(M1) = M1(in_i m_1, msg_1, out_j m_1)$, the semantics of the message is represented in the π-calculus as a process with parameters $M1(in_i m_1, msg_1, out_j m_1)$ as follows:

$$\mathbf{M1(in_i m_1, msg_1, out_j m_1)} \overset{def}{=} \mathbf{in_i m_1}(s).\underbrace{\overline{\mathbf{out_j m}}_{\mathbf{1}} < \mathbf{msg_1} > .\ldots.\overline{\mathbf{out_j m_1}} < \mathbf{msg_1} >}_{\mathbf{k}}.\ \mathbf{M1(in_i m_1, msg_1, out_j m_1)}$$

The message process will send sequentially (one by one) a specified or unspecified number of messages "msg_1" to the object Oj.

– **If there is a parallel iteration**:

When we have a message with a parallel iteration k (k may be specified or unspecified i.e. "*"), $\Omega_{Messages}(M1) = M1(in_i m_1, msg_1, out_j m_1)$, the semantics of the message is represented in the π-calculus as a process with parameters $M1(in_i m_1, msg_1, out_j m_1)$ as follows:

$$\mathbf{M1(in_i m_1, msg_1, out_j m_1)} \overset{def}{=} \mathbf{in}_i \mathbf{m_1}(s).\prod_{k=1}^{k} \overline{\mathbf{out_j m_1}} < \mathrm{msg}_\mathbf{1} > .\ \mathbf{M1(in_i m_1, msg_1, out_j m_1)}$$

The message process will send concurrently a specified or unspecified number of message "msg_1" to the object Oj.

– **If it is a conditional message**:

When we have a message with a condition COND1 $\in$ COND, α_{cond} (M1) = {COND1}, Ω_{Conds} (COND1) = $guard_1$, $\Omega_{Messages}$(M1) = M1(in_im_1, $guard_1$, msg_1, out_jm_1), the semantics of the message is represented in the π-calculus as a process with parameters M1(in_im_1, $guard_1$, msg_1, out_jm_1) as follows:

$$\mathbf{M1(in_im_1, guard_1, msg_1, out_jm_1)} \overset{def}{=} \mathbf{in_im_1}(s).(\nu \mathbf{g})\,\overline{\mathbf{guard_1}} < \mathbf{g} > .\mathbf{g(y)}.([\mathbf{y = true}]\,\overline{\mathbf{out_jm_1}} < msg_1 > .$$
$$\mathbf{M1(in_im_1, guard_1, msg_1, out_jm_1)} + [\mathbf{y = false}]\,\mathbf{M1(in_im_1, guard_1, msg_1, out_jm_1)})$$

The message process creates a channel "g" and executes the output action "$\overline{guard}_1$" and the input action "g(y)" to retrieve the current evaluation of the condition. If the condition is verified, the matching construct "[y=true]" allows the submission of the message along the output action "$\overline{out_i m_1}$ $<msg_1>$" to the target process which models "Oj". In the other case the message will not be sent and the process will wait until the condition is verified.

– **A full message**:

When we have a full message M1, $\Omega_{Messages}$(M1) = M1(in_im_1, $guard_1$, msg_1, val_1, prm_1, out_jm_1), the semantics of the message is represented in the π-calculus as a process with parameters M1(in_im_1, $guard_1$, msg_1, val_1, prm_1, out_jm_1) as follows:

$$\mathbf{M1(in_im_1, guard_1, msg_1, val_1, prm_1, out_jm_1)} \overset{def}{=} \mathbf{in_im_1}(s).(\nu \mathbf{g})\,\overline{\mathbf{guard_1}} < \mathbf{g} > .\mathbf{g(y)}.$$
$$([\mathbf{y = true}]((\nu \mathbf{p})\overline{\mathbf{prm}}_1 < \mathbf{p} > |\mathbf{p}(\overrightarrow{\mathbf{pts}}))\underbrace{(\overline{\mathbf{out_jm}}_1 < msg_1, \overrightarrow{\mathbf{pts}}, val_1 > .\cdots .\overline{\mathbf{out_jm}}_1 < msg_1, \overrightarrow{\mathbf{pts}}, val_1 > .\cdots}_{\mathbf{K\ times}} +$$
$$\textstyle\prod_{k \in K} \overline{\mathbf{out_jm}}_1 < msg_1, \overrightarrow{\mathbf{pts}}, val_1 >).\mathbf{M1(in_im_1, guard_1, msg_1, val_1, prm_1, out_jm_1)} + [\mathbf{y = false}]$$
$$\mathbf{M1(in_im_1, guard_1, msg_1, val_1, prm_1, out_jm_1)})$$

The process modeling the message is executed when the "$in_i m_1$" channel is fired, after that, the condition will be evaluated using the "$guard_1$" channel. If the condition is not verified, the message will not be sent. If the condition is verified, the message process proceeds to send either one message, multiple messages consequently or multiple messages concurrently.

Definition 3 (*Processes communication*)
The objects are related using the connectors, which are consequently represented as links between processes representing objects and processes representing messages and vice versa. They relate output ports of source processes with input ports of target processes. Here, we can use the communication reduction rule defined in [2]:

$$\mathrm{COMM} : (\ldots + x(y).P)|(\ldots + \bar{x}z.Q) \rightarrow P\{z/y\}|Q$$

This rule represents the communication between two complementary processes (have complementary subjects) and consequently all free occurrence of y in P will be replaced by z using the substitution {z / y} after the communication. Based on this rule, the author in [2] has introduced a linking operator relation "∩" on two π-calculus processes as follow:

$$\mathbf{P} \cap \mathbf{Q}(\mathbf{t}/\mathbf{p}, \mathbf{t}/\mathbf{q}) \stackrel{def}{=} \nu\mathbf{t}(\{\mathbf{t}/\mathbf{p}\}\mathbf{P}|\{\mathbf{t}/\mathbf{q}\}\mathbf{Q})$$

This relation indicates that port p of the process P is linked with the port q of the process Q and then the channel t will be internalized.

- **Rule 3: (links)**

We aspire from the interesting relation defined in (definition 3) to facilitate the expression of the translation from an object to message and vice versa represented as processes. Furthermore, we define a process called "*Connector*" that links all object processes "O_i" and message processes "M_j" of a system as bellow:

$$\textbf{Connector} \stackrel{def}{=} \prod_{i,j \in I} \nu \overrightarrow{c} (\{c/o_i\}O_i|\{c/m_j\}M_j)$$

- **Rule 4: (CDs)**

Suppose an UML communication diagram CD = (CDname, Elements, α_{cond}, α_{msg}, α_{val}, α_{prm}, α_{in}, α_{out}). Ω_{CDname}(CD) = CDname. The semantics of this communication diagram is modeled in the π-calculus by the process expression:

$$\textbf{CDname} \stackrel{def}{=} \prod_{i \in I} O_i | \prod_{j \in I} M_j | \mathit{Connector}$$

Where "O_i" and "M_j" represent respectively the objects and messages processes resulting from applying the function defined in (Definition 2) in (rule 1) and (rule 2) on the communication diagram. Thus the model can be seen as π-calculus concurrent processes which are running in parallel.

5 Example: Online Bookshop

To illustrate our approach, we consider an example of an UML communication diagram for an Online Bookshop drawn from [19], which is described in Fig. 7.

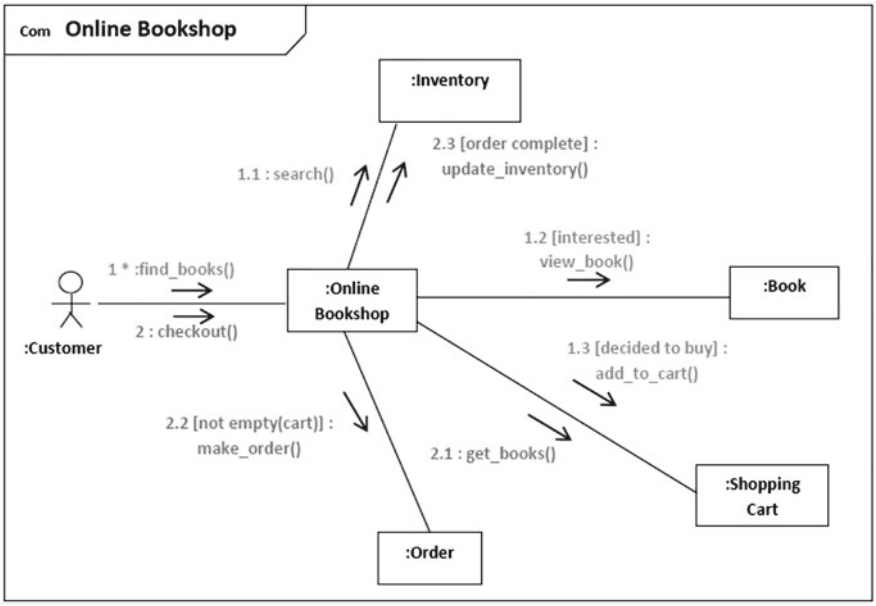


Fig. 7 An example of UML communication diagram for online bookshop

Web customer which is depicted as actor can search, view, select and buy books. Communication starts with the iterative message "1 *: find_books()" which could be repeated some unspecified number of times. Client searches inventory of books "1.1 : search()", and if he is interested in some book, he can view description of the book "1.2 [interested]: view_book()". If client decides to buy, he can add the book to the shopping cart "1.3 [decided to buy]: add_to_cart()".

Checkout "2 : checkout()" includes getting list of books "2.1 : get_books()" from shopping cart, creating order "2.2 [not empty(cart)] : make_order()", and updating inventory "2.3 [order complete] : update_inventory()", if order was completed.

The execution semantics of the Online Bookshop interaction modeled as a communication diagram is given by the following π-calculus specification:

$$\mathbf{Customer\ (in_i Customer,\ seq,\ out_j Customer)} \stackrel{def}{=} \mathbf{in_i Customer}.\ \tau.(\nu\ \mathbf{x})\overline{\mathbf{seq}}\langle \mathbf{x}\rangle.\mathbf{x}(\mathbf{s}).$$
$$\overline{\mathbf{out_j Customer}}\langle \mathbf{s}\rangle.\mathbf{Customer}\ (\mathbf{in_i Customer},\ \mathbf{seq},\ \mathbf{out_j Customer})$$

$$\mathbf{Find\_books\ (in_i Find\_books,\ find\_books(),\ out_j Find\_books)} \stackrel{def}{=} \mathbf{in_i Find\_books}(\mathrm{s}).$$
$$\underbrace{\overline{\mathbf{out_j Find\_books}}\langle \mathrm{find\_books}()\rangle.\ldots.\overline{\mathbf{out_j Find\_books}}\langle \mathrm{find\_books}()\rangle}_{*\ \text{times}}.$$
$$\mathbf{Find\_books\ (in_i find\_books,\ find\_books(),\ out_j find\_books)}$$

OnlineBookshop (in$_i$OnlineBookshop, seq, out$_j$OnlineBookshop) $\stackrel{def}{=}$ **in$_i$OnlineBookshop. τ.**
(ν x) $\overline{\text{seq}}$ <x>.x(s).$\overline{\text{out}_j\text{OnlineBookshop}}$<s>.OnlineBookshop(in$_i$OnlineBookshop, seq, out$_j$OnlineBookshop)

Search (in$_i$search, search(), out$_j$search) $\stackrel{def}{=}$ **in$_i$search** (s).$\overline{\text{out}_j\text{search}}$ <search() >**.Search (in$_i$search, search(), out$_j$search)**

Inventory (in$_i$Inventory) $\stackrel{def}{=}$ **in$_i$Inventory.Inventory (in$_i$Inventory)**

View_book (in$_i$View_book, guard$_1$, view_book(), out$_j$View_book) $\stackrel{def}{=}$ **in$_i$View_book**(s).(νg)$\overline{\text{guard}_1}$ <g>**.**
g(y).([y=true] $\overline{\text{out}_j\text{View\_book}}$<view_book()>.
View_book (in$_i$View_book(), guard$_1$, view_book(), out$_j$ View_book) +
[y=false] View_book (in$_i$View_book(), guard$_1$, view_book(), out$_j$View_book))

Book (in$_i$Book) $\stackrel{def}{=}$ **in$_i$Book .Book (in$_i$Book)**

Add_to_cart (in$_i$Add_to_cart, guard$_1$, add_to_cart(), out$_j$Add_to_cart) $\stackrel{def}{=}$ **in$_i$Add_to_cart**(s).
(ν g)$\overline{\text{guard}_1}$ <g>**. g(y).([y=true]** $\overline{\text{out}_j\text{Add\_to\_cart}}$<**add_to_cart**()>.
Add_to_cart (in$_i$Add_to_cart, guard$_1$, add_to_cart(), out$_j$Add_to_cart) +
[y=false] Add_to_cart (in$_i$Add_to_cart, guard$_1$, add_to_cart(), out$_j$Add_to_cart))

ShoppingCart (in$_i$ShoppingCart) $\stackrel{def}{=}$ **in$_i$ShoppingCart . ShoppingCart (in$_i$ShoppingCart)**

Checkout (in$_i$Checkout, checkout(), out$_j$Checkout) $\stackrel{def}{=}$ **in$_i$Checkout** (s). $\overline{\text{out}_j\text{Checkout}}$ <**checkout()**> **. Checkout (in$_i$Checkout, checkout(), out$_j$Checkout)**

Get_books (in$_i$Get_books, get_books(), out$_j$Get_books) $\stackrel{def}{=}$ **in$_i$Get_books**(s).
$\overline{\text{out}_j\text{Get\_books}}$<**get_books()**>**.Get_books (in$_i$Get_books, get_books(), out$_j$Get_books)**

Make_order (in$_i$Make_order, make_order(), out$_j$Make_order) $\stackrel{def}{=}$ **in$_i$Make_order**(s).
$\overline{\text{out}_j\text{Make\_order}}$<**make_order()**>**.Make_order (in$_i$Make_order, make_order(), out$_j$Make_order)**

Order (in$_i$ Order) $\stackrel{def}{=}$ **in$_i$Order . Order (in$_i$Order)**

Update_inventory (in$_i$Update_inventory, guard$_1$, update_inventory(), out$_j$Update_inventory) $\stackrel{def}{=}$
in$_i$Update_inventory(s).(νg)$\overline{\text{guard}_1}$ <g>**. g(y).**
([y=true] $\overline{\text{out}_j\text{Update\_inventory}}$<**update_inventory()**>.
Update_inventory (in$_i$Update_inventory, guard$_1$, update_inventory(), out$_j$Update_inventory) +
[y=false] Update_inventory (in$_i$Update_inventory, guard$_1$, update_inventory(), out$_j$Update_inventory))

Connector $\stackrel{def}{=}$ ν**c$_1$, c$_2$, c$_3$, c$_4$, c$_5$, c$_6$, c$_7$, c$_8$, c$_9$, c$_{10}$, c$_{11}$, c$_{12}$, c$_{13}$, c$_{14}$, c$_{15}$, c$_{16}$({c$_1$, c$_9$ / out$_1$Customer, out$_2$Customer}Customer | {c$_1$, c$_2$ / in$_1$ Find_books, out$_1$ Find_books} Find_books | {c$_2$, c$_{10}$, c$_3$, c$_5$, c$_7$, c$_{11}$, c$_{13}$, c$_{15}$ / in$_1$ OnlineBookshop, in$_2$OnlineBookshop, out$_1$ OnlineBookshop, out$_2$ OnlineBookshop, out$_3$ OnlineBookshop, out$_4$ OnlineBookshop, out$_5$OnlineBookshop, out$_6$OnlineBookshop} OnlineBookshop | {c$_3$, c$_4$ / in$_1$search, out$_1$search} Search | {c$_4$, c$_{16}$ / in$_1$ Inventory, in$_2$Inventory}Inventory | {c$_5$, c$_6$ / in$_1$ View_book, out$_1$ View_book}View_book | {c$_6$ / in$_1$Book}Book | {c$_7$, c$_8$ / in$_1$Add_to_cart, out$_1$Add_to_cart}Add_to_cart | {c$_8$, c$_{12}$ / in$_1$ShoppingCart, in$_2$ShoppingCart}ShoppingCart | {c$_9$, c$_{10}$ / in$_1$ Checkout, out$_1$ Checkout}Checkout | {c$_{11}$, c$_{12}$ / in$_1$ Get_books, out$_1$ Get_books}Get_books | {c$_{13}$, c$_{14}$ / in$_1$ Make_order, out$_1$ Make_order}Make_order | {c$_{14}$ / in$_1$ Order}Order | {c$_{15}$, c$_{16}$ / in$_1$ Update_inventory, out$_1$ Update_inventory}Update_inventory**

OnlineBookshop $\stackrel{def}{=}$ **Customer | Inventory | Book | Order | OnlineBookshop | Find_books | Search | View_book | Add_to_cart | ShoppingCart | Checkout | Get_books | Make_order | Update_inventory | Connector**

Figure 8 represents the flow graph of the Online Bookshop interaction. It is, as appeared, very similar to the communication diagram which describes the system. This informal flow graph facilitates the comprehension of the π-calculus processes and links of the specification.

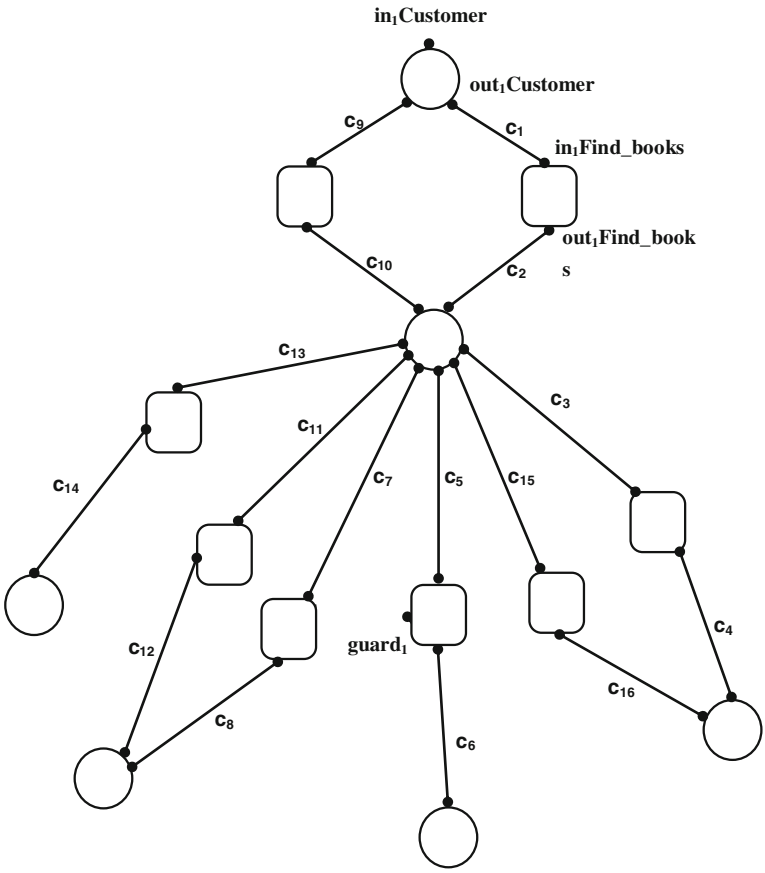


Fig. 8 Flow graph of the Online Bookshop interaction

6 Model analysis

We can elegantly proceed to the analysis and verification of UML communication diagrams modeled as π-calculus process expressions. Indeed, we can check the following:

- Equivalence checking between different communication diagrams which represent various interactions by verifying the equivalence between the corresponding π-calculus process expressions.
- Model checking of communication diagrams to check the correctness from certain properties such as deadlock, livelock, inconsistencies…etc.

We can automatically perform that using special analysis tool such as the mobility workbench MWB [6, 7]. It just needs to import the corresponding process expression in the tool, then applying some instructions to check automatically what we want to verify.

In fact, we have some results which prove the validity and correctness of our formalization. We have actually applied our approach on the communication diagram generated from a Java program by reverse engineering in [23] (see Fig. 9), and we have obtain a π-calculus specification which allowed us to verify (using MWB) that the dynamic behavior of two methods was the same, and to check possible deadlocks (no out-going transitions) in their execution such as illustrated in Fig. 10.

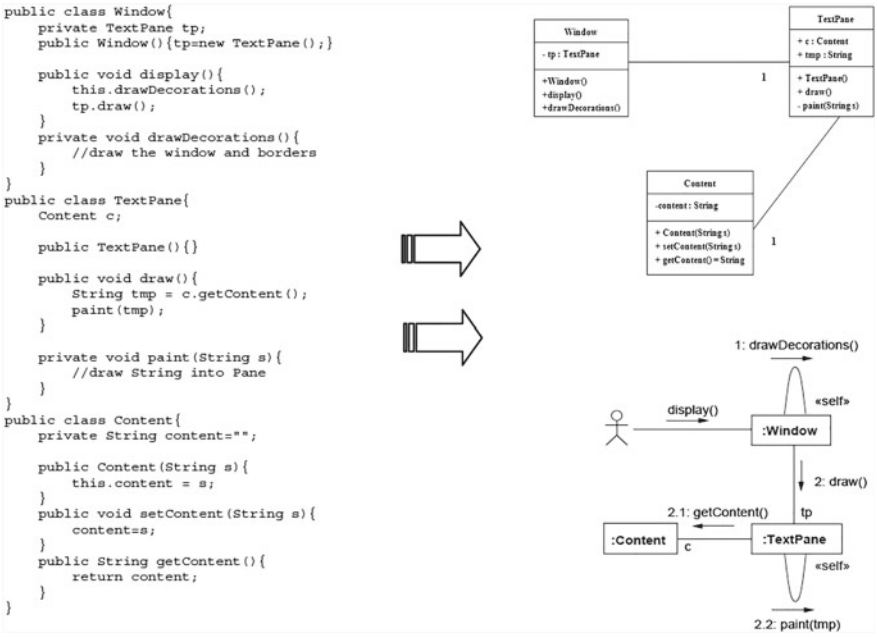


Fig. 9 Capturing a Java program behavior with UML diagrams

Fig. 10 Model checking and equivalence checking of a communication diagram in MWB

By applying our approach on the communication diagram in Fig. 9, we obtain the π-calculus specification loaded in the MWB (see Fig. 10) where some textual representations are replaced in it as follows: the restriction $\boldsymbol{\nu}$ as $\hat{}$, the output action $\overline{\mathbf{x}}$ as **'x**, the internal action $\boldsymbol{\tau}$ as **t** and each process identifier expression in the MWB will starts with the keyword **agent**. We can import the generated π-calculus specification using the command **input "picalculus.ag"**, or type the agent declarations manually. Below, we specification is generated and, thus, imported and displayed using the command **env**.

Although all preceding remarks, more work is needed for a detailed model analysis. This is a subject of a future paper that will profit from the rich theory of the π-calculus tools that is available in the literature for the automatic analysis, verification and reasoning on more complex systems modeled in UML.

7 Tool Suite Implementation

7.1 Overview

In order to automate our approach, we have developed a tool suite that implements the mapping described above. It is based on model transformation technique. Modeling and model transformation play an essential role in the MDA (Model Driven Architecture) [24]. MDA recommends the massive use of models in order to allow

a flexible and iterative development, thanks to refinements and enrichments by successive transformations. A model transformation is a set of rules that allows passing from a meta-model to another, by defining for each one of elements of the source their equivalents ones among the elements of the target. These rules are carried out by a transformation engine; this last read the source model which must be conform to the source meta-model, and apply the rules defined in the model transformation to lead to the target model which will be itself conform to the target meta-model. The principle of model transformation is illustrated by Fig. 11.

Graph transformation was largely used for the expression of model transformation [25]. Particularly transformations of visual models can be naturally formulated by graph transformation, since the graphs are well adapted to describe the fundamental structures of models. The set of graph transformation rules constitutes what is called the model of graph grammar. A graph grammar is a generalization, for graphs, of Chomsky grammars. Each rule of a graph grammar is composed of a graph of left side (LHS) and of a graph of right-sided (RHS). Therefore, the graph transformation is the process that choosing a rule among the graph grammar rules, apply this rule on a graph and reiterate the process until no rule can be applied [25].

AToM^3 [26] "A Tool for Multi-formalism and Meta-Modeling" is a visual tool for model transformation, written in Python [27] and is carried out on various platforms (Windows, Linux,…). It implements various concepts like multi-paradigm modeling, meta-modeling and graph grammars. It can be also used for simulation and code generation. AToM^3 provides the possibility to propose meta-models and building visual models according to them, and using a graph grammar to go from a model to another. It has proven that it is a very powerful tool in dealing with model transformation problems and this is what encourages us to use it.

For the realization of the tool suite, we have proposed a meta-model for class diagrams and a meta-model for communication diagrams, these meta-models will allow us to edit visually and with simplicity class and communication diagrams on

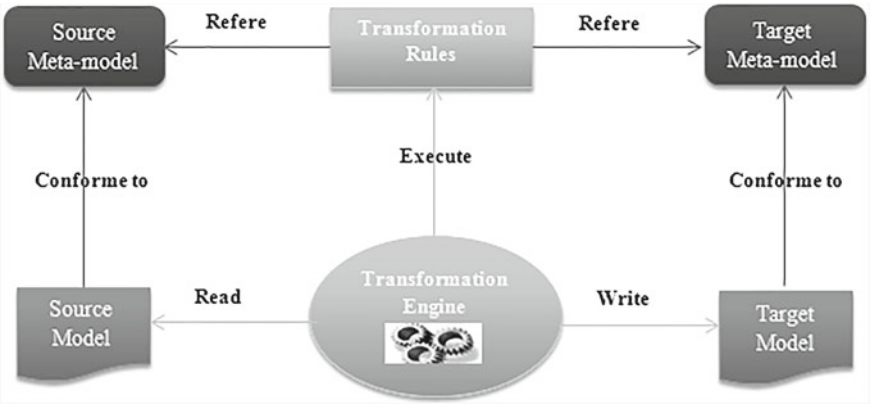


Fig. 11 Model transformation principle

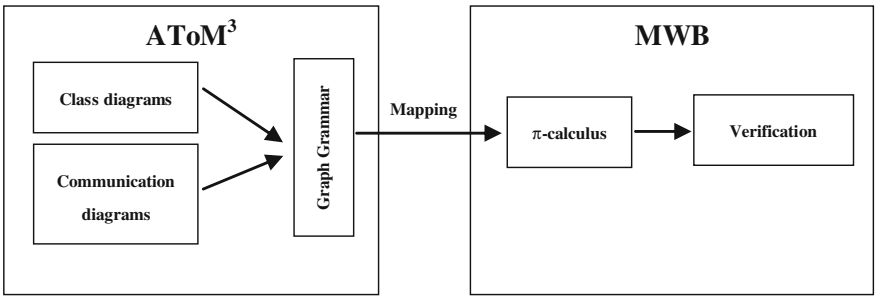


Fig. 12 Tool for editing, mapping and verifying UML diagrams

the canvas of the AToM^3 tool. In addition, we have developed a graph grammar made up of multiple rules that allows transforming progressively all what will be modeled on the canvas towards a π-calculus specification in textual format stored in a disk file. The graph grammar is based on some transformation rules; each rule deals with some constructs in the left hand side (LHS) i.e. class and communication diagrams, to transform them to others constructs in the right hand side (RHS) i.e. π-calculus specifications. This last will be directly imported to the mobility workbench MWB [6, 7] for analysis purposes. Figure 12 presents an overview of the implementation task.

To build these two meta-models, we have used the meta-meta-model (*CD_class DiagramsV3*) provided by AToM^3 and the constraints are expressed in Python code.

7.2 Class Diagram Meta-Model

To build UML class diagrams models in AToM^3, we have defined a meta-model (see Fig. 13 on the left) for them composed of 2 classes and 4 associations. The classes are "Class" and "Package". The associations are "Association", "Generalization", "Dependency" and "Association Class". The generated tool will allow the description of UML class diagrams quite simply (see Fig. 13 on the right).

7.3 Communication Diagram Meta-Model

To build UML communication diagrams models in AToM^3, we have defined a meta-model (see Fig. 14 on the left) for them composed of one class "Object" and one association "Link". The generated tool will allow the description of UML communication diagrams quite simply (see Fig. 14 on the right).

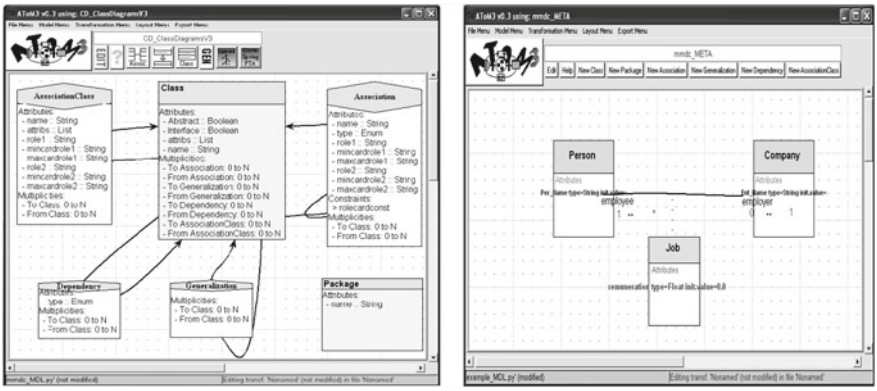


Fig. 13 Meta-model and Generated tool for UML class diagrams

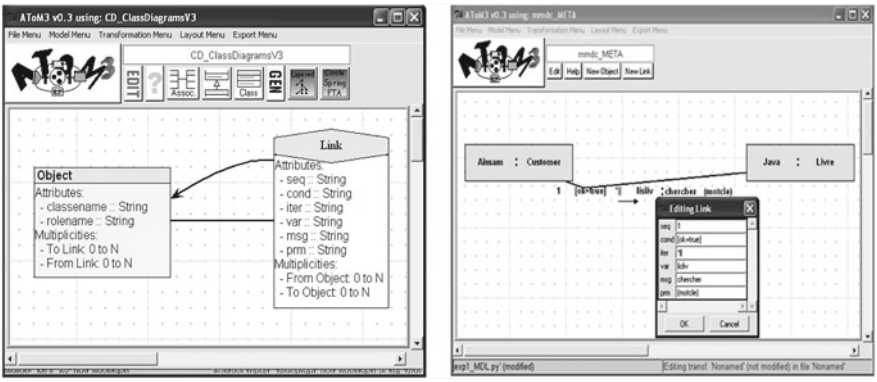


Fig. 14 Meta-model and Generated tool for UML communication diagrams

7.4 The Graph Grammar

We have developed a graph grammar that consists of multiple rules. It allows the formal translation of UML models (composed of class and communication diagrams) to π-calculus according to the mapping proposed above. Each rule of the graph grammar has, besides the LHS and RHS described above, a name, a priority i.e. an order of execution, a condition that must be verified to execute the rule and an action to be performed. It is noted that one rule of the graph grammar can implement multiple rules of the mapping, and vise-versa. In addition, since the AToM^{3}is multi-paradigm, we can describe all the models on the same canevas. We present here (see Fig. 15) just one rule to illustrate the translation since the other rules have the same implementation.

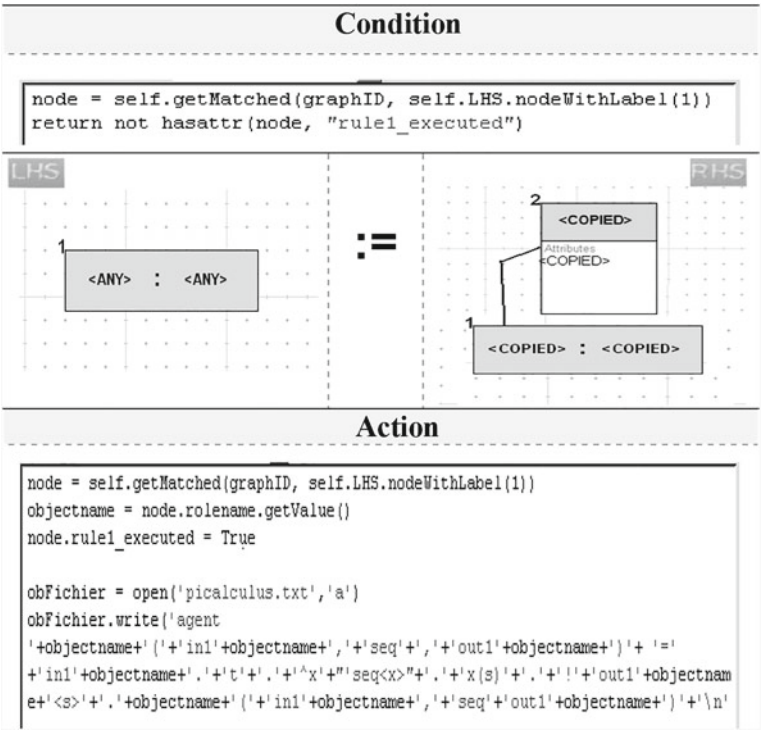


Fig. 15 Lifelines mapping in the graph grammar

Rule 1 : *Lifeline mapping*
Name : *lifeline2process*
Priority : 1
Role : *This rule transforms a lifeline (i.e. object) towards a pi-calculus process. In the condition of the rule we test if the lifeline is already transformed. If not, we proceed to relate the lifeline to its class in the class diagram in the RHS. In the action of the rule we open a file "picalculus.ag" and we add to it the corresponding pi-calculus code of the lifeline and its corresponding class as illustrated in the mapping.*

8 Discussion

The translation presented herein is generally done in a straightforward manner; it is implementation-oriented. Thus, the individual correspondences, which are being expressed as rules can straightforwardly be taken and easily coped with to be imple-

mented (and this is what we have seen using graph grammars). The approach provides a simple mapping for class diagrams and a full translation for communication diagrams. In fact, it defines a function that takes a communication diagram and then produces the corresponding π-calculus expression. The formal definition of such a mapping then typically requires the definition of a textual abstract syntax which is described in the beginning of the section. The formalization allows the mapping to be defined using structural induction/recursion which is very appropriate for the implementation.

Our translation maintain multiple aspects in a communication diagram modeling task, it guarantees that each object is distinguishable from others by its channels since it is represented as a process with special channels. The mapping assures the uniqueness of every message since it is modeled as an independent full entity i.e. a process with particular channels. Furthermore, the scheduling and the synchronization of the messages between the objects of the diagram of communication are perfectly specified using channels exchange object/message and message/object respectively in the generated π- calculus specification .

Besides the system structure and interaction aspects modeled here, our approach also covered elegantly the aspect of using collaboration diagrams for specifying system's state transformation invented by [3] without affecting the modeling of the other aspects. In fact, in our case we use a full π-calculus process expression with special channels to model the state transformation of an object during the interaction. Thus, the state of each object is conserved and provided by the π-calculus process that models the object during all its lifecycle in the interaction.

We have omitted some others elements which could appear in communication diagrams such as transient links because they do not affect the behavior of the interaction.

9 Concluding Remarks

In this paper we have proposed, elaborated and validated a new approach for capturing and verifying the dynamic behavior of systems using UML models and π-calculus. To do so, we have considered that we have used class and communication diagrams to model these systems. Then we have proposed a mapping between these diagrams and π-calculus in order to use to capabilities of this later to verify these models. The MWB (Mobility Workbench) tool is used for this purpose.

In order to illustrate the applicability of our approach, we have applied it to an example of an online bookshop system modeled in a communication diagram and we have shown how to generate the π-calculus specification from it. Using our approach, we also show how analyzing the dynamic behavior of a program like indicated in the second example.

We have also developed a tool suite based on a combined meta-modeling and graph grammar approach using the AToM^3 tool. This integrated tool allows any user to describe a system using class and communication diagrams, then it can translate

them immediately to the π-calculus and starts the analysis and verification tasks using the MWB tool.

In our future work, we plan to completely hide implementations details and make the verification process transparent to the users so as they won't be aware of the details of the approach. The long-term objective of our research is to create a framework that supports systematic verification of all UML diagrams including interaction overview diagrams, develop a formal foundation and provide a full formal semantics based π-calculus for all UML diagrams. It is hoped that the paper will stimulate further work in a field whose importance will increasingly be recognized.

Acknowledgments The authors would like to thank Mr. Rachid Echahed (CNRS and University of Grenoble, France) for his advices and comments regarding this work.

References

1. Object Management Group (OMG), Unified Modeling Language (2012). http://www.omg.org/spec/UML/
2. Milner, R.: Communicating and Mobile Systems: The π-calculus. Cambridge University Press (1999)
3. Heckel, R., Sauer, S.: Strengthening UML collaboration diagrams by state transformations. Fundamental Approaches to Software Engineering, pp. 109–123. Springer, Berlin (2001)
4. Ambler, S.W.: The Elements of UML^{TM} 2.0 Style. Cambridge University Press (2005)
5. Belghiat, A., Chaoui, A., Beldjehem, M.: Capturing and verifying dynamic program behaviour using UML communication diagrams and pi-calculus. In: IEEE International Conference on Information Reuse & Integration (IRI), pp. 318–325. IEEE (2015)
6. Victor, B.: A Verification Tool for the Polyadic π-Calculus. Licentiate thesis, Department of Computer Systems. Uppsala University (1994)
7. Victor, B., Moller, F.: The Mobility Workbench - A Tool for the π-calculus. In Dill, D. (eds.) Proceedings of the Conference on Computer-Aided Verification (CAV'94). LNCS, vol. 818, pp. 428–440. Springer Verlag, (1994)
8. Lano , K., Bicarregui, J.: Formalizing the UML in Structured Temporal Theories. In Rumpe, B., Kilov, H. (eds.) Proceedings of Second ECOOP Workshop on Precise Behavioral Semantics. ECOOP'98. Munich, Germany (1998)
9. Overgaard, G.: A formal approach to collaborations in the Unified Modeling Language. In: Proceedings of the UML'99—Beyond the Standard. LNCS, vol. 1723, pp. 99–115. Springer-Verlag (1999)
10. Gomaa, H.: Validation of Dynamic Behavior in UML Using Colored PetriNets. In: Proceedings of the UML2000 (2000)
11. Saldhana, J., Shatz, S.M.: UML diagrams to object petri net models: an approach for modeling and analysis. In: Proceedings of the International Conference on Software Engineering and Knowledge Engineering (SEKE). pp. 103–110. Chicago (2000)
12. Dong, Z., He, X.: Integrating UML statechart and collaboration diagrams using hierarchical predicate transition nets. In: Proceedings of pUML, pp. 99–112 (2001)
13. Elmansouri, R., Chaoui, A., Kerkouche, E., Khalfaoui, K.: From UML statechart and collaboration diagrams to colored petri net models: a graph transformation based approach for modeling and analysis of business processes in virtual enterprises. In: Proceedings of the Fourth South-East European Workshop on Formal Methods. IEEE, Washington, DC, USA (2009)
14. Merah, E., Messaoudi, N., Saidi, H., Chaoui, A.: Design of ATL rules for transforming UML 2 communication diagrams into buchi automata. Int. J. Softw. Eng. Appl. **7**(2), 1–15 (2013)

15. Motameni, H., Ghassempouri, T.: Transforming fuzzy communication diagram to fuzzy petri net. Am. J. Sci. Res. **16**, 62–73 (2011)
16. Haroonabadi, A., Teshnehlab, M.: A novel method for behavior modeling in uncertain information systems. World Acad. Sci. Eng. Technol. **41**, 959–966 (2008)
17. Gagnon, P., Mokhati, F., Badri, M.: Applying model checking to concurrent UML models. J. Object Technol. **7**(1), 59–84 (2008)
18. Chama, W., Elmansouri, R., Chaoui, A.: Model checking and code generation for UML diagrams using graph transformation. Int. J. Softw. Eng. Appl. (IJSEA), **3**(6) (2012)
19. Fakhroutdinov, K.: (2013). http://www.uml-diagrams.org/
20. Belghiat, A., Chaoui, A., Maouche, M., Beldjehem, M.: Formalization of mobile UML statechart diagrams using the π-calculus: an approach for modeling and analysis. In Dregvaite, G., Damasevicius, R. (eds.) ICIST 2014, CCIS 465, pp. 236–247. Springer (2014)
21. Yang, D., Zhang, S.S.: Using π-calculus to formalize UML activity diagrams. In: 10th International Conference and Workshop on the Engineering of Computer-based Systems, pp. 47–54. IEEE Computer Society (2004)
22. Lam, V.S.W.: On π-calculus semantics as a formal basis for UML activity diagrams. Int. J. Softw. Eng. Knowl. Eng. **18**(4), 541–567 (2008)
23. Kollmann, R., Gogolla, M.: Capturing dynamic program behaviour with UML collaboration diagrams. In Sousa, P., Ebert, J. (eds.) Proceedings of the 5th European Conference on Software Maintenance and Reengineering. IEEE, Los Alamitos (2001)
24. Object Management Group (OMG), Model Driven Architecture (MDA) (2004). http://www.omg.org/mda/
25. Karsai, G., Agrawal, A.: Graph transformations in OMG's model-driven architecture. Applications of Graph Transformations with Industrial Relevance. LNCS, vol. 3062, pp. 243–259. Springer, Berlin (2004)
26. AToM^3 (2002). Home page: http://atom3.cs.mcgill.ca
27. Python. Home page: http://www.python.org

A Real-Time Concurrent Constraint Calculus for Analyzing Avionic Systems Embedded in the IMA Connected Through TTEthernet

Sardaouna Hamadou, John Mullins and Abdelouahed Gherbi

Abstract The Integrated Modular Avionics (IMA) architecture and the Time-Triggered Ethernet (TTEthernet) network have emerged as the key components of a typical architecture model for recent civil aircrafts. In this paper, we present a first approach to model and verify avionic systems embedded in the Integrated Modular Architecture (IMA) connected through the TTEthernet Network, by using TTCC, a real-time concurrent constraint process calculus with an operator to define infinite periodic behaviors specific to IMA and TTEthernet. We argue that the operational constructs for interacting processes with one another of TTCC provide a suitable language to describe the time triggered architecture while the declarative aspects of this calculus provide a simple and elegant way to specify requirements of avionic systems. We also illustrate how TTCC may provide a *unified framework* for the analysis of avionic systems embedded in the IMA connected through the TTEthernet by modeling, specifying and verifying a case study developed in collaboration with an industrial partner, the landing gear system.

Keywords Concurrent constraint programming · Real-time systems · TTEthernet · IMA · Optimization of communication configuration · Functional verification

S. Hamadou · J. Mullins (✉)
Department of Computer and Software Engineering,
École Polytechnique de Montréal, Montreal, Quebec, Canada
e-mail: John.Mullins@polymtl.ca

S. Hamadou
e-mail: Sardaouna.Hamadou@polymtl.ca

A. Gherbi
Department of Software and IT Engineering,
École de Technologie Supérieure, Montreal, Quebec, Canada
e-mail: abdelouahed.gherbi@etsmtl.ca

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_4

1 Introduction

A new standard of architecture called *Integrated Modular Avionics* (*IMA*) [1] has recently emerged to cope with the growing complexity of avionic embedded systems. This type of architecture is characterized essentially by the sharing of distributed computing resources, called modules. Sharing these resources requires to guarantee some safety and liveness properties. In order to achieve this, the *Avionic Full Duplex Switched Ethernet* (*AFDX*) [2] has been adopted as a networking standard for the avionic systems. However, *AFDX* underuses the physical capacities of the network. Also, a new standard of the avionic network, the *Time-Triggered Ethernet* (TTEthernet) has been proposed [3]. This standard enables to achieve a best usage of the network and is more deterministic since the schedule is established offline. Both *IMA* and TTEthernet segregate mixed-criticality components into partitions for a safer integration. *IMA* enables applications to interact safely by partitioning them in time and space over distributed Real-Time Operating Systems (RTOS). TTEthernet, on the other hand, allows these distributed RTOS to communicate safely with each other by partitioning bandwidth into time slots.

Although a considerable effort has recently been devoted for the validation of TTEthernet (e.g. [9, 23, 38]), the analysis and validation of TTEthernet usage in model-based development for the integration on *IMA* has been so far relatively ignored. Concurrent Constraint Programming (CCP) [30, 33] is a well-established formalism for reasoning about concurrent and distributed systems. It is a matured model of concurrency with several reasoning techniques (e.g. [6, 14, 29]) and implementations (e.g. [21, 32, 34]). It is adopted in a wide spectrum of domains and applications such as *biological phenomena*, *reactive systems* and *physical systems*. CCP is a powerful way to define complex synchronization schemes in concurrent and distributed settings parametric in a constraint system. This provides a very flexible way to tailor data structures to specific domains and applications. We refer the reader to [25] for a recent survey on CCP-based models.

Drawing on earlier work on timed CCP-based formalisms [24, 31], we have presented in [17] the Time-Triggered Constraint-Based Calculus (TTCC) to provide a formal basis for the analysis of time-triggered architectures in avionic embedded systems. It is built around a small number of primitive operators or combinators parametric in a *constraint system*. It extends the Timed Concurrent Constraint Programming (TCC) [31] in order to define infinite periodic behaviors specific to *IMA* and TTEthernet. Like all CCP-based calculi, it enjoys a dual view of processes as agents interacting with one another and as logical formulas.

The main objective of this paper is to exploit the view of TTCC processes as agents interacting with one another, provided by its operational semantics to elegantly model concepts related to the *IMA* and TTEthernet architectures and to demonstrate the relevance of the calculus not only as a unifying model for time-triggered and event triggered systems embedded in IMA connected with a TTEthernet network but also as a unifying specification language to specify specific kinds of requirements depending on the nature of the traffic, by taking advantage of the alternate view of TTCC processes as logical formulas provided by its denotational semantics.

The main contributions are the following: (1) a complete description of the *landing gear system*. To the best of our knowledge, this is the first time that this important critical avionic system is modeled as an IMA system connected through the TTEthernet network; (2) the definition of generic TTCC processes for the integration of time-triggered traffic and rate constrained traffic; (3) the complete modeling of the landing gear system in TTCC calculus; and (4) the specification of the most important timing requirement of rate constrained traffic, that is the *schedulability requirement*, and the *end-to-end delay requirement of functional chains* as denotations.

The rest of the paper is organized as follows: in Sect. 3 we fix some basic notations and briefly revise the concepts of *IMA* and TTEthernet architectures; the landing gear system is introduced in Sect. 4; Sect. 5 recalls the syntax and the operational semantics of a time-triggered extension of TCC. Sections 6, 7 and 8 deliver our core technical contribution: the definition of a denotational semantics for TTCC programs, our conceptual framework illustrated by a revised and completed modeling of the landing gear system [10] and the specification in our framework of the schedulability requirement of the rate constrained traffic and the end-to-end delay requirement of functional chains involving both time-triggered and rate constrained traffic as denotations; Sect. 9 contains our concluding remarks.

2 Related Works

This paper revises and expends an earlier version [16] where we proposed the denotational semantics of TTCC calculus and illustrated its usefulness as a specification language using a simplified sub-system of the flight management system.

To our knowledge, TTCC is the first calculus to provide a comprehensive framework for the end-to-end delay requirements of functional chains in a TTEthernet setting. Previous process algebraic models do not deal with both *IMA* and TTEthernet [13, 26, 35], or only accounted for the *IMA* concept without providing a comprehensive set of reasoning techniques for the verification of the requirements of avionic systems. Similarly, formal calculi such as the Network Calculus [20] and the Real-Time Calculus [28] fall short of accounting for the time triggered architecture while maintaining a good accuracy in specifying the system designs [22].

While several worst case end-to-end analyses for rate constrained traffic have been proposed, including analyses based on Network Calculus [11, 12], Finite State Machines [27], Timed Automata [4] or Trajectory Approach [7, 8], none of these methods is applicable to TTEthernet since they do not take in account the impact of TT messages on schedulability of rate constrained traffic. The proposed timing analysis is the first to eliminate the pessimism of the very few previous analyses [36, 39] of the impact of time-triggered traffic on the schedulability of rate constrained traffic as it computes the exact worst-case end-to-end delay.

3 Preliminaries

In order to make this paper as self-contained as possible, we describe briefly in this section the main concepts of *IMA* and TTEthernet architectures, which both underpin the work presented in this paper.

3.1 Integrated Modular Avionics

Avionic systems are safety-critical systems which should meet stringent safety, reliability and performance requirements. The design of these systems is based on The Integrated Modular Avionics (*IMA*) architecture [5]. The main feature of this architecture is that the resources are shared between the system functionalities reducing consequently the cost associated with large volume of wiring and equipment. In the same time, this architecture ensures the isolation of the system functionalities to meet the safety requirements.

The *IMA* architecture is a modular real-time architecture for the engineering of avionic systems defined in ARINC653 [1]. In this architecture, a system is composed of a set of modules each of which is basically a computing resource. Each functionality of the system is implemented by a set of functions distributed across different modules. Each module hosts the execution of several functions. The functions that are deployed on the same module may have different criticality levels. An IMA-based avionic system is therefore a mixed-criticality system. In order to meet safety requirements, these functions should be strictly isolated using different partitions of the module. The partitioning of these functions is two dimensional: spatial partitioning and temporal partitioning. The spatial partitioning is implemented using a static exclusive assignment of all the module resources for the partition being executed. The temporal partitioning is implemented using an allocation of a periodic time window dedicated for the execution of each partition.

3.2 Time-Triggered Ethernet

Ethernet is now a well established standard network (IEEE STD 802.3). Even though, Ethernet is increasingly used to support industrial and embedded systems with high bandwidths requirements, it does not, however, meet strict timing and safety requirements of critical applications such as avionic systems. Essentially, Ethernet uses an event-triggered transfer principle where an end system can access the network at arbitrary points in time. Service to the end systems is on a first come first serve scheme. Consequently, this can substantially increases the transmission delay and jitter when several end systems need to communicate over the same shared medium.

TTEthernet is a new SAE standard [3] which specifies a set of (time-triggered) services extending the Ethernet IEEE standard 802.3. TTEthernet is based on the Time-triggered communication paradigm [19]. Therefore, it establishes a network-wide common time base implemented using a robust and precise synchronization of the clocks of the different end systems and switches in the network. This results in a bounded latency and a low jitter. TTEthernet integrates both time-triggered and event-triggered communication on the same physical network. TTEthernet limits latency and jitter for time-triggered (TT) traffic, limits latency for rate constrained (RC) traffic, while simultaneously supporting the best-effort (BE) traffic service of IEEE 802.3 standard. This allows the application of Ethernet as a unified networking infrastructure. Therefore, TTEthernet supports the deployment of mixed-criticality applications at the network level.

The physical topology of a TTEthernet network is a graph $\mathbf{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices composed of end systems and switches and $\mathcal{E}$ is the set of the edges connecting vertices and representing the physical links. Each physical link connecting two vertices defines *two directed "dataflow links"*. The set of dataflow links is denoted $\mathcal{L}$. We denote by $[u, v]$ the dataflow link from vertex u to vertex v and by

$$p = [[v_1, v_2], [v_2, v_3], \ldots [v_{m-2}, v_{m-1}], [v_{m-1}, v_m]]$$

the dataflow path connecting one end system (the sender) v_1 to exactly one other end system (the receiver) v_m. In accordance with the Ethernet convention, information between the sender and receiver is communicated in form of messages f called *frames*. $\mathcal{F}$ denotes the set of all frames. Frames may be delivered from a sender to multiple receivers where the individual dataflow paths between the sender and each single receiver together form a *virtual link*. Hence, a virtual link vl is the union of the dataflow paths that link the sender to each receiver. We denote by $\mathcal{DP}$ (resp. $\mathcal{VL}$) the set of dataflow paths (resp. virtual links).

4 Landing Gear System

This section introduces the landing gear system, a leading example used to illustrate our conceptual framework.

4.1 System Description

The landing gear system is a critical avionic sub-system that controls the maneuvering of the landing gears and their associated doors. It is composed of three distinct parts: (1) mechanical and hydraulic system, (2) digital system, and (3) the cockpit interface. There are three landing sets that compose the mechanical and hydraulic system: the nose (aka front), the right and the left landing sets. Each landing set

also has three components: a box, a door, and a gear. The digital part is a control software that provides commands for extension/retraction to the actuating hydraulic cylinders governing the motion of the landing sets. It also informs the pilot about the global state of the mechanical and hydraulic system. The pilot interface has two components: an Up/Down handle to command the retraction/extension of the gears and a set of lights which inform the pilot about the current position of the gears.

Moving the handle from up to down triggers the extension of the landing gear. The full extension process is composed of the following sequence of events: door unlocking, door opening, gear unlocking, gear extending, and gear locking. The reverse movement executes the retraction process according to the following sequence: gear unlocking, gear retracting, gear locking, door closing, and door locking. These processes can be reversed at any time when the pilot reverses the handle from down to up and vice versa. The logical behavior of a gear (resp. door) is illustrated by Fig. 1 (resp. by Fig. 2) and its temporal behavior is shown in Table 1.

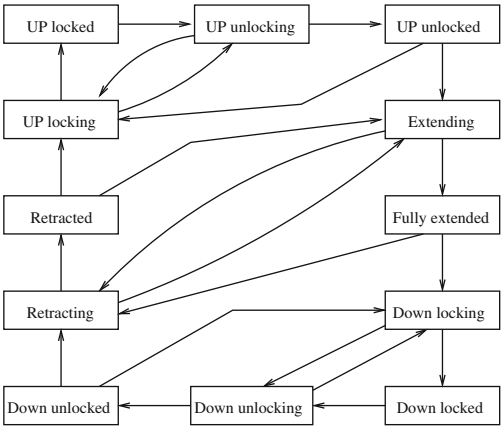


Fig. 1 Logical behavior of a gear

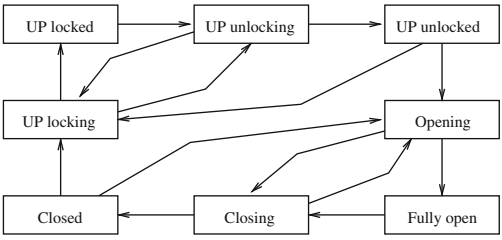


Fig. 2 Logical behavior of a door

Table 1 Temporal behavior of the landing gears

Duration (in second) of ...	NLG		MLG	
	Gear	Door	Gear	Door
Unlocking in down position	0.4	–	0.4	–
Retracting/closing	1.6	1.2	2	1.6
Locking in high position	0.8	0.4	0.8	0.4
Unlocking in high position	0.8	0.4	0.8	0.4
Extending/opening	1.6	1.2	2	1.6
Locking in down position	0.4	–	0.4	–

4.2 System Architecture

In order to study the behavior of the landing gear system, we model it as an IMA system interconnected through a TTEthernet network as shown in Fig. 3. The digital part of the system is modeled by two identical computing modules (ES_1 and ES_2) executing the same control software. Each module is composed of two partitions: the electro-valve command (EVC) partition which provides the pilot's commands to the mechanical part, and the landing gear status (LGS) partition which informs the pilot

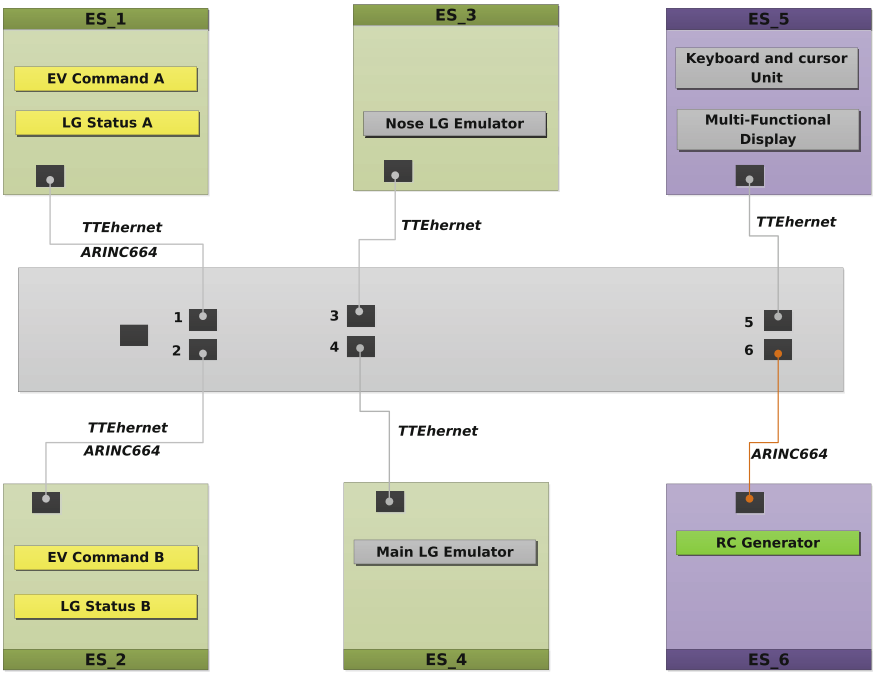


Fig. 3 LGS

about the status of the landing sets. We emulate the mechanical component by two modules: one module for the nose landing set (ES_3) and one module (ES_4) for the main (viz. left and right) landing sets. Each of these modules is composed of one single partition which actuates the gear position according to the extension/retraction command received from EVC and informs LGS about the current position of the gear. One single module (ES_5) composed of two partitions emulates the pilot interface. The keyboard and cursor unit (KU) partition emulates the Up/Down handle and the multi-functional display partition emulates the lights set. The final modele (ES_6) contains one single partition that tries to "saturate" the network with rate constrained traffic. All these distributed modules are interconnected through one single TTEthernet switch.

5 The TTCC Process Calculus

This section describes the syntax and the operational semantics of the Time-Triggered Concurrent Constraint Programming (ttcc). We start by recalling a fundamental notion in CCP-based calculi: *constraint systems*.

5.1 Constraint Systems

The **TTCC** model is parametric in a constraint system specifying the structure and interdependencies of information that processes can ask of and add to a *central shared store*. A constraint system provides a signature (a set of constants, functions and predicates symbols) from which constraints, which represent pieces of information upon which processes may act, can be constructed as well as an entailment relation denoted $\vdash$, which specifies the interdependencies between these constraints. Formally, a constraint system is a pair (Σ, Δ) where Σ is a signature and Δ is a first order theory over Σ. Constraints are first-order formulas over $\mathcal{L}(\Sigma)$, the underlying first-order language under Σ. We shall denote by $\mathcal{C}$ the set of constraints in the underlying constraint system with typical elements $c, d, \ldots$. Given two constraints (i.e. two pieces of information) c and d, we say that c entails d, and write $c \vdash d$, if and only if, in all models of Δ, either c is not true or both c and d are true. In other words, d can be deduced from c.

Throughout the rest of this paper, we shall consider the widely used Finite-Domain Constraint System **FD**$[max]$ proposed [18] where Σ is given by the constants symbols $0, 1, 2, \ldots, max - 1$ and the relation symbols $=, \neq, <, \leq, >, \geq$. Δ, the first order theory over Σ, is given by the axioms in Number Theory. The intuitive meaning of **FD**$[max]$ is that variables range over a finite domain of values $\{0, 1, 2, \ldots, max - 1\}$.

5.2 Process Syntax

To model real time, we assume a discrete global clock where time is divided into *discrete intervals* or *time units*. A common store is used as communication medium by **TTCC** processes to post and read constraints. We use $\mathcal{P}roc$ to denote the set of all **TTCC** processes, with typical elements $P, Q, \ldots$. They are built from the following primitive operators:

$$P, Q, \ldots ::= \mathbf{0} \mid \mathbf{tell}(c) \mid \mathbf{when}\ c\ \mathbf{do}\ P \mid P \mid Q \mid (\mathbf{local}\ x; c)\ \mathbf{in}\ P \mid \mathbf{next}\ P$$
$$\mid \mathbf{unless}\ c\ \mathbf{next}\ P \mid \mathbf{catch}\ c\ \mathbf{in}\ P\ \mathbf{finally}\ Q \mid !_T P \mid A(\tilde{x})$$

The null process $\mathbf{0}$ does nothing. The process $\mathbf{tell}(c)$ adds constraint c to the store within the current time. The process $\mathbf{when}\ c\ \mathbf{do}\ P$ executes P if its guard constraint c is entailed by the store in the current time. Otherwise, P is discarded. $P \parallel Q$ represents P and Q acting concurrently. We write $\prod_{1 \le i \le n} P_i$ for the parallel composition $P_1 \parallel P_2 \parallel \cdots \parallel P_n$. The process $(\mathbf{local}\ x; c)\ \mathbf{in}\ P$ behaves like P, except that it declares variable x private to P (the information about the variable x, represented as c, is hidden to other processes). In $\mathbf{next}\ P$, the process P will be activated in the next time unit. The weak time-out $\mathbf{unless}\ c\ \mathbf{next}\ P$ represents the activation of P the next time unit if c cannot be inferred from the current store in the current time. Otherwise, P will be discarded. Modeling avionic systems may require detecting anomalies and react *instantaneously*. Consider for instance the software which is in charge of controlling gears and doors, it is also responsible of detecting anomalies and informing the pilots. The strong preemption $\mathbf{catch}\ c\ \mathbf{in}\ P\ \mathbf{finally}\ Q$ models this situation where a process P is aborted to immediately execute a process Q when the store can entail a specific constraint c. The operator $!_T$ is used to define infinite periodic behavior. $!_T P$ represents $P \parallel \mathbf{next}^T P \parallel \mathbf{next}^{2T} P \parallel \cdots$ where $\mathbf{next}^T P$ is the abbreviation of $\mathbf{next}\ (\mathbf{next}\ (\cdots (\mathbf{next}\ P) \cdots))$ where $\mathbf{next}$ is repeated T times. The process $A(\tilde{x})$ is an *identifier* with arity $|\tilde{x}|$. We assume that every such an identifier has a unique (recursive) definition of the form $A(\tilde{x}) \stackrel{\text{def}}{=} P$.

5.3 Operational Semantics

The dynamics of the calculus is specified by means of two transition relations between configurations $\longrightarrow, \Longrightarrow \subseteq \mathcal{C}onf \times \mathcal{C}onf$ obtained by the rules in Table 2. A configuration is a pair $\langle P, d\rangle \in \mathcal{P}roc \times \mathcal{C}$ where d represents the current store. $\mathcal{C}onf$ denote the set of all configurations with typical elements $\Gamma, \Gamma', \ldots$.

An *internal transition* $\langle P, d\rangle \longrightarrow \langle P', d'\rangle$ means that P under the current store d evolves internally into P' and produces the store d' and corresponds to an operational step that take place during a time-unit. Rules in upper part of Table 2 define the internal transitions. Rule (R-Tell) means that a **tell** process adds information (viz. a constraint) to the current store and terminates. Rules (R-Ask) specify that the guard

Table 2 Internal transition rules $\longrightarrow$ (upper part) and the observable transition rule $\Longrightarrow$ (lower part)

$$\text{(R-Tell)}\ \frac{}{\langle \mathbf{tell}(c),d\rangle \longrightarrow \langle \mathbf{0}, d\wedge c\rangle} \qquad \text{(R-Ask)}\ \frac{d\vdash c}{\langle \mathbf{when}\ c\ \mathbf{do}\ P,d\rangle \longrightarrow \langle P,d\rangle}$$

$$\text{(R-Par)}\ \frac{\langle P,d\rangle \longrightarrow \langle P',d'_p\rangle \quad \langle Q,d\rangle \longrightarrow \langle Q',d'_q\rangle}{\langle P \parallel Q,d\rangle \longrightarrow \langle P' \parallel Q', d'_p \wedge d'_q\rangle} \qquad \text{(R-Per)}\ \frac{}{\langle !_T P,d\rangle \longrightarrow \langle P \parallel \mathbf{next}^T(!_T P),d\rangle}$$

$$\text{(R-UNL)}\ \frac{d\vdash c}{\langle \mathbf{unless}\ c\ \mathbf{next}\ P,d\rangle \longrightarrow \langle \mathbf{0},d\rangle} \qquad \text{(R-PRE1)}\ \frac{d\vdash c}{\langle \mathbf{catch}\ c\ \mathbf{in}\ P\ \mathbf{finally}\ Q,d\rangle \longrightarrow \langle Q,d\rangle}$$

$$\text{(R-PRE2)}\ \frac{\langle P,d\rangle \longrightarrow \langle P',d'\rangle \quad d\not\vdash c}{\langle \mathbf{catch}\ c\ \mathbf{in}\ P\ \mathbf{finally}\ Q,d\rangle \longrightarrow \langle \mathbf{catch}\ c\ \mathbf{in}\ P'\ \mathbf{finally}\ Q,d'\rangle}$$

$$\text{(R-Loc)}\ \frac{\langle P,c\wedge\exists x d\rangle \longrightarrow \langle P',c'\wedge\exists x d\rangle}{\langle (\mathbf{local}\ x;c)\ \mathbf{in}\ P,d\rangle \longrightarrow \langle (\mathbf{local}\ x;c')\ \mathbf{in}\ P', d\wedge\exists x c'\rangle}$$

$$\text{(R-Def)}\ \frac{A(\tilde{x}) \stackrel{\text{def}}{=} P \quad \langle P[\tilde{v}/\tilde{x}],d\rangle \longrightarrow \langle P',d'\rangle}{\langle A(\tilde{v}),d\rangle \longrightarrow \langle P',d'\rangle}$$

$$\text{(R-Obs)}\ \frac{\langle P,c\rangle \longrightarrow^* \langle Q,d\rangle \not\longrightarrow \quad Fu(Q)=R}{P \stackrel{(c,d)}{\Longrightarrow} R}$$

constraint of an ask process must be entailed by the current store when it is triggered. Rule (R-Par) specifies the concurrent execution of multiple processes and assumes *maximal parallelism* since, typically in avionic systems, agents running concurrently are located on different modules. In rule (R-Loc), the process (**local** x; c) **in** P behaves like P, except that it distinguishes between the external (viz. d) and the internal (viz. c) points of view.[1] Rule (R-Per) states that in $!_T P$, the process P is activated in the current time and then repeated periodically. In Rule (R-UNL), the process **unless** c **next** P evolves into **0** if its guard can be entailed from the current store. The strong preemption **catch** c **in** P **finally** Q (R-PRE) interrupts P in the current time if c is entailed by the store and continues with process Q. Otherwise, P continues. If P finishes Q is discarded. Finally, rule (R-Def) states that the identifier process $A(\tilde{x})$ behaves like P. Process $P[\tilde{v}/\tilde{x}]$ denotes P where each variable $x_i \in \tilde{x}$ inside P is substituted by the value $v_i \in \tilde{v}$.

In order to unfold the timed operator **next**, we consider *observable transitions*. An observable transition $P \stackrel{(c,d)}{\Longrightarrow} R$, means that the process P under the current store c evolves in *one time-unit* to R and produces the store d. We say that R is an *observable evolution* of P. Rule (R-Obs) in lower part of Table 2 defines the observable transitions. The transition $P \stackrel{(c,d)}{\Longrightarrow} R$ is obtained from a finite sequence of internal transitions $\langle P, c\rangle \longrightarrow^* \langle Q, d\rangle \not\longrightarrow$ where $Fu(Q) = R$ and $Fu : \mathcal{P}roc \to \mathcal{P}roc$, the *future function* is defined as

[1] See [25] for more details.

$$Fu(Q) = \begin{cases} Q' & \text{if } Q = \textbf{next } Q', \\ & \text{or } Q = \textbf{unless } c \textbf{ next } Q', \\ Fu(Q_1) \parallel Fu(Q_2) & \text{if } Q = Q_1 \parallel Q_2, \\ (\textbf{local } x) \textbf{ in } Fu(Q') & \text{if } Q = (\textbf{local } x; c) \textbf{ in } Q', \\ \textbf{catch } c \textbf{ in } Fu(R) \textbf{ finally } S & \text{if } Q = \textbf{catch } c \textbf{ in } R \textbf{ finally } S, \\ Fu(Q') & \text{if } Q = A(\tilde{v}) \text{ and } A(\tilde{v}) \stackrel{\text{def}}{=} Q', \\ \textbf{0} & \text{otherwise.} \end{cases}$$

$\Gamma \not\longrightarrow$ means that there is no Γ' such that $\Gamma \longrightarrow \Gamma'$.

Example Let P be as in Example 1 and consider a store with a pair of variables $(pReq, wpId)$: a boolean one and an integer one. Now assume the following initial stores: $c = (\textbf{false}, 0)$ and $d = (\textbf{true}, 0)$. Then we have

$$P \stackrel{(c,c)}{\Longrightarrow} \textbf{0}$$

and

$$P \stackrel{(d,d)}{\Longrightarrow} \textbf{next}^3\ R \stackrel{(d,d)}{\Longrightarrow} \textbf{next}^2\ R \stackrel{(d,d)}{\Longrightarrow} \textbf{next}\ R \stackrel{(d,e)}{\Longrightarrow} \textbf{0},$$

where $e = (\textbf{true}, 1)$.

Note that in the transition $P \stackrel{(c,d)}{\Longrightarrow} R$, d is not automatically transferred to the next time unit. However, as it is often the case in avionic systems, the adding and the querying do not happen within the same time unit. For instance, a partition may produce a frame at time unit t, but the frame is only scheduled at time unit $t + \delta$. Hence, the computing process must ensure that the values that are needed in the following time units are maintained. For this purpose, following [24], we introduce some auxiliary processes which provide a basis for the specification of mutable and persistent data. We start by extending the signature Σ with a unary predicate *change*. The following process defines a structure (or a variable) x that has a current value z which has to be maintained in the future unless it is stated in the current store (by the predicate $change(x)$) that it needs to be assigned a new value in the future.

$$pres[x : z] \stackrel{\text{def}}{=} \textbf{tell}(x = z) \parallel \textbf{unless } change(x) \textbf{ next } pres[x : z] \tag{1}$$

The following process assigns the value v in the next time unit to the structure x. The value v is maintained until the next update.

$$update[x : v] \stackrel{\text{def}}{=} \textbf{tell } change(x) \parallel \textbf{next } pres[x : v] \tag{2}$$

In the following section, we define the denotational semantics which is a compositional semantics approximating the operationals semantics. Operators of the language are modeled in equations of the denotational semantics by simple set-theoretic and fixed-point operators, which will be conveniently used as specification language in Sect. 8.

6 Denotational Semantics and Denotations

Let $\mathcal{C}^*$ and $\mathcal{C}^\omega$ denote the set of finite and infinite sequences of constraints in $\mathcal{C}$, respectively. The interpretation of a sequence of observable transitions

$$P = P_1 \overset{(c_1,d_1)}{\Longrightarrow} P_2 \overset{(c_2,d_2)}{\Longrightarrow} P_3 \overset{(c_3,d_3)}{\Longrightarrow} \cdots$$

which we denote as $P \overset{(\alpha,\beta)}{\Longrightarrow^\omega}$ where $\alpha = c_1c_2c_3\ldots$ and $\beta = d_1d_2d_3\ldots$, is that at time unit i the environment provides a *stimulus* c_i and P_i produces d_i as *response*. The *input-output behavior* of P is defined as

$$io(P) = \{(\alpha, \beta) : P \overset{(\alpha,\beta)}{\Longrightarrow^\omega}\}.$$

The strongest postcondition (or quiescent) behavior of P defined as

$$sp(P) = \{\alpha : P \overset{(\alpha,\alpha)}{\Longrightarrow^\omega}\}$$

corresponds to the set of input sequences of which P can run without adding any information, therefore what we observe is that the input and the output coincide. The notation $\exists_x\alpha$ denotes the sequence obtained from α by replacing for any i, the ith element c_i of α by $\exists_x c_i$.

The denotational semantics is specified by mean of a function $[\![\cdot]\!] : \mathcal{P}roc \to \mathcal{P}(\mathcal{C}^\omega)$, defined by the equations in Table 3. The denotation $[\![P]\!]$ is meant to capture $sp(P)$. The process **0** add no information to any sequence (Eq. D0). The set of sequences to which **tell**(c) cannot add information are those whose first element is stronger then c (Eq. D1). Eq. D2 states that either d enables the guard c, then a quiescent trace of **when** c **do** P is d followed by a quiescent trace of P, or d does not enable c, **when** c **do** P suspends and then a quiescent trace is d followed by any trace. Eq. D3 states that a sequence is a quiescent trace of $P \parallel Q$ if and only if it is a quiescent trace of both P and Q. Eq. D4 states that If $d' \cdot \alpha'$ is a quiescent trace of P and $d \cdot \alpha$ and $d' \cdot \alpha'$ differ only on the information about the variable x (i.e. $\exists_x d \cdot \alpha = \exists_x d' \cdot \alpha'$) then $d \cdot \alpha$ is a quiescent trace of (**local** x; c) **in** P. Process **next** P does not constraint the first element of a sequence, hence $d \cdot \alpha$ is quiescent for **next** P if α is quiescent for P (Eq. D5). Eq. D6 states that either d does not enable the guard c, then a quiescent trace of **unless** c **next** P is d followed by a quiescent trace of P, or d enables c, **unless** c **next** P suspends and then a quiescent trace is d followed by any trace. Eq. D7 states that a quiescent trace of **catch** c **in** P **finally** Q is either a quiescent trace of P that contains no element entailing c, or a prefix of a quiescent trace of P whose elements do not entail c followed by a quiescent trace of Q whose first element entails c. We say that α' is k-suffix of α if $\alpha'_i = \alpha_{k+i}$ for all $i \geq 1$. A sequence α is quiescent for $!_T P$ if any kT-suffix of α ($k \geq 0$) is quiescent for P (Eq. D8). Alternately, $[\![!_T P]\!]$ could be defined as the greatest fixpoint of $F(X) = [\![P]\!] \cap \{\beta \cdot \alpha : \beta \in \mathcal{C}^T \text{ and } \alpha \in X\}$ on the complete lattice $(\mathcal{P}(\mathcal{C}^\omega), \subseteq)$.

Table 3 Denotational semantics of TTCC

$D0$	$[\![\mathbf{0}]\!]$	$=$	$\mathcal{C}^\omega$
$D1$	$[\![\mathbf{tell}(c)]\!]$	$=$	$\{d \cdot \alpha : d \vdash c \text{ and } \alpha \in \mathcal{C}^\omega\}$
$D2$	$[\![\mathbf{when}\ c\ \mathbf{do}\ P]\!]$	$=$	$\{d \cdot \alpha \in [\![P]\!] : d \vdash c\} \cup \{d \cdot \alpha \in \mathcal{C}^\omega : d \nvdash c\}$
$D3$	$[\![P \parallel Q]\!]$	$=$	$[\![P]\!] \cap [\![Q]\!]$
$D4$	$[\![(\mathbf{local}\ x; c)\ \mathbf{in}\ P]\!]$	$=$	$\{d \cdot \alpha$: there exists $d' \cdot \alpha' \in [\![P]\!]$ s.t $d' \vdash c$ and $\exists_x d \cdot \alpha = \exists_x d' \cdot \alpha'\}$
$D5$	$[\![\mathbf{next}\ P]\!]$	$=$	$\{d \cdot \alpha : d \in \mathcal{C} \text{ and } \alpha \in [\![P]\!]\}$
$D6$	$[\![\mathbf{unless}\ c\ \mathbf{next}\ P]\!]$	$=$	$\{d \cdot \alpha : d \vdash c \text{ and } \alpha \in \mathcal{C}^\omega\} \cup \{d \cdot \alpha : d \nvdash c \text{ and } \alpha \in [\![P]\!]\}$
$D7$	$[\![\mathbf{catch}\ c\ \mathbf{in}\ P\ \mathbf{finally}\ Q]\!]$	$=$	$\{\beta \cdot \alpha : \alpha \in [\![Q]\!]$ and there exists γ s.t $\beta \cdot \gamma \in [\![P]\!]$ and $\forall i \leq \lvert\beta\rvert,\ \beta_i \nvdash c$ and $\alpha_1 \vdash c\}$ $\cup \{\beta \in [\![P]\!] : \forall i \leq \lvert\beta\rvert,\ \beta_i \nvdash c\}$
$D8$	$[\![!_T P]\!]$	$=$	$\nu X.([\![P]\!] \cap \{\beta \cdot \alpha : \beta \in \mathcal{C}^T \text{ and } \alpha \in X\})$
		$=$	$\{\alpha : \forall \beta \in \mathcal{C}^{T*}, \forall \alpha' \in \mathcal{C}^\omega,\ \text{if } \alpha = \beta \cdot \alpha' \text{ then } \alpha' \in [\![P]\!]\}$
$D9$	$[\![A(\tilde{x})]\!]$	$=$	$[\![P]\!]$ for $A(\tilde{x}) \stackrel{\text{def}}{=} P$

Finally, process calls $A(\tilde{x})$ are interpreted according to the interpretation $[\![P]\!]$ that gives meaning to the process definition $A(\tilde{x}) \stackrel{\text{def}}{=} P$ (Eq. D9).

The following example illustrates the $[\![\cdot]\!]$ operator and the way denotations can be used as specifications.

Example Consider a controller that must trigger an alert signal in case of failure while a component $P =!_1 \mathbf{tell}(c)$ is currently executing (i.e. the environment introduces as stimulus the constraint *failure*) and suppose that $c \nvdash failure$. The following process intends to implement such a system:

$$controller = \mathbf{catch}\ failure\ \mathbf{in}\ P\ \mathbf{finally}\ \mathbf{tell}(alert).$$

$[\![controller]\!]$ is computed from the bottom as follows:

$$\begin{aligned}
[\![\mathbf{tell}(c)]\!] &= \{e : e \vdash c\} \cdot \mathcal{C}^\omega \\
[\![P]\!] &= \{e : e \vdash c\}^\omega \text{ s.t.} \\
[\![\mathbf{tell}(alert)]\!] &= \{f : f \vdash alert\} \cdot \mathcal{C}^\omega \\
[\![controller]\!] &= [\![\mathbf{catch}\ failure\ \mathbf{in}\ P\ \mathbf{finally}\ \mathbf{tell}(alert)]\!]
\end{aligned}$$

$$
\begin{aligned}
&\{\beta\cdot\alpha : \alpha \in [\![\mathbf{tell}(alert)]\!], \exists\gamma, \beta\cdot\gamma \in [\![P]\!],\\
&\quad \forall i \leq |\beta|,\ \beta_i \nvdash c, \alpha_1 \vdash failure\} \cup \{e : e \vdash c \text{ and } e \nvdash failure\}^\omega\\
&= \{e : e \vdash c \text{ and } e \nvdash failure\}^\omega \cup \{\beta\cdot\alpha : \alpha \in f\cdot\mathcal{C}^\omega,\\
&\quad \exists\gamma \text{ s.t. }, \beta\cdot\gamma = e^\omega, \forall i \leq |\beta|,\ \beta_i \nvdash c,\ \alpha_1 \vdash failure, e \vdash c,\ f \vdash alert\}\\
&= \{\beta\cdot\alpha : \alpha \in f\cdot\mathcal{C}^\omega, \beta \in e^*,\\
&\quad \forall i \leq |\beta|,\ \beta_i \nvdash failure, \alpha_1 \vdash failure,\ e \vdash c,\ f \vdash alert\}\\
&= \{e : e \vdash c \text{ and } e \nvdash failure\}^*\cdot\\
&\quad \{f : f \vdash failure \wedge alert\}\cdot\mathcal{C}^\omega \cup \{e : e \vdash c \text{ and } e \nvdash failure\}^\omega
\end{aligned}
$$

Denotations can be used not only to model but also to specify. Consider, for example, the informal requirement

In case of failure detection, an alert signal must be immediately triggered.

If it is understood as the inequation

$$[\![controller]\!] \subseteq \{\alpha \in \mathcal{C}^\omega : \forall_i \alpha_i \vdash (failure \rightarrow alert)\},$$

meaning that

Anytime along any execution of P, if an error is detected then an alert signal is immediately triggered,

then it is not verified since

$$(failure \wedge alert) failure^\omega \in [\![controller]\!].$$

However, it makes better sense to understand this requirement as

$$
\begin{aligned}
[\![controller]\!] \subseteq\ & \{\alpha \in \mathcal{C}^\omega : \forall_i \alpha_i \nvdash failure\}\\
& \cup \{\alpha \in \mathcal{C}^\omega : \exists_i \forall_{j<i} (\alpha_i \vdash (failure \wedge alert) \text{and } \alpha_j \nvdash failure)\}
\end{aligned}
$$

meaning that

The first failure detection triggers immediately an alert signal.

This formal interpretation of the requirement is verified. Indeed, for any

$$
\begin{aligned}
&\alpha \in \{e : e \vdash c \text{ and } e \nvdash failure\}^* \cup \{e : e \vdash c \text{ and } e \nvdash failure\}^\omega \text{ and}\\
&\beta \in \{f : f \vdash failure \wedge alert\}\cdot\mathcal{C}^\omega
\end{aligned}
$$

one has, for any $i \in \mathbb{N}$, $\alpha_i \nvdash failure$ since $c \nvdash failure$ and $\beta_1 \vdash failure \wedge alert$.

7 Modeling the Landing Gear System

This section shows the use of our calculus to model IMA and TTEthernet concepts using the *landing gear system* [10] introduced in Sect. 4 as case study. We start by modeling IMA related components of the system. Later, we address the integration of the network component of the system.

7.1 Architecture Modeling

As stated earlier, the IMA architecture is a set of distributed computing resources, called modules, that enable mixed-criticality applications to interact safely. Each module is composed of a set of partitions. Each of partition implements an avionic application that is executed periodically. Therefore, we assume that each partition is a black box function g. The function g takes the current value(s) v of the input variable(s) var_in of the partition and stores the result $g(v)$ into the partition's output variable(s) var_out. Moreover, since applications (viz. partitions) implemented on the same module share its computing resources, their scheduling must satisfy the so-called *contention-free* property, that is the mutual-exclusion of their execution times. Hence, each partition is fully characterized by the following elements:

- g the function that models the application implemented within the partition;
- o the offset time, that is the scheduling time of the partition;
- τ the duration, that is the execution time of the partition;
- π the period of the partition.

We therefore model a partition by the following generic process

$$Part \stackrel{\text{def}}{=} !_{\pi}\Big(\mathbf{next}^{o} \prod_{v} \mathbf{when}\ (var\_in = v)\ \mathbf{do}\ \mathbf{next}^{\tau-1} update[var\_out : g(v)]\Big). \tag{3}$$

Note that we use $\tau - 1$ in the above definition because of the *update* (see Eq. 2) process which must stop maintaining the old value of the variable currently being updated one time-unit earlier.

The local scheduling of partitions on each module is defined and validated statically offline. Therefore, we assume that the IMA schedules are well-defined, that is the contention-free property is satisfied. Hence, we model each computing module as the parallel composition of its partitions. For our case study, the temporal parameters of its partitions are given in Table 4 and the complete model of its computing modules is given in Table 5. To complete the description of the model, we give below the definitions of the functions that implement the partitions.

KU. The KU partition, which emulates the handle, simply copies the current value of its input $handle \in \{\text{Up}, \text{Down}\}$ into its output $command_{ES5}$. Hence, its associate function is the identity function.

Table 4 IMA-schedule

Partition	π	τ	o	*var_in*	*var_out*	Module
KU	80	5	0	handle	$command_{ES5}$	ES_5
MFD	80	5	40	*LG_status*	*light*	ES_5
EVCA	40	5	20	$command_{ES1}$	$EVcommand_{ES1}$	ES_1
LGSA	40	5	0	$(NLG\_stat_{ES1}, MLG\_stat_{ES1})$	LG_status_{ES1}	ES_1
EVCB	40	5	20	$command_{ES2}$	$EVcommand_{ES2}$	ES_2
LGSB	40	5	0	$(NLG\_stat_{ES1}, MLG\_stat_{ES2})$	LG_status_{ES2}	ES_2
NLGE	40	10	0	$(EVcommand_{ES3}, NLG\_stat_{ES3})$	NLG_stat_{ES3}	ES_3
MLGE	40	10	0	$(EVcommand_{ES4}, MLG\_stat_{ES4})$	NLG_stat_{ES4}	ES_4
RCG	20	5	0	$RCmsg_{ES6}$	$RCmsg_{ES6}$	ES_6

Table 5 Landing gear system model: IMA modules

$$
\begin{aligned}
\text{ES\_5} \overset{\text{def}}{=}\ & !_{80}\Big(\prod_{v\in\{\text{Up, Down}\}} \textbf{when } (handle = v) \textbf{ do next}^4 update[command_{ES5} : v]\Big) \parallel \\
& !_{80}\Big(\textbf{next}^{40} \prod_{v\in\{\text{extended, retracted, maneuvering}\}} \textbf{when } (LG\_status = v) \textbf{ do} \\
& \textbf{next}^4 update[light : g_{MFD}(v)]\Big) \\
\text{ES\_1} \overset{\text{def}}{=}\ & !_{40}\Big(\textbf{next}^{20} \prod_{v\in\{\text{Up, Down}\}} \textbf{when } (command_{ES1} = v) \textbf{ do} \\
& \textbf{next}^4 update[EVcommand_{ES1} : g_{EVC}(v)]\Big) \parallel \\
& !_{40}\Big(\prod_{(v_1,v_2)\in\{0,\cdots,500\}\times\{0,\cdots,500\}} \textbf{when } \big((NLG\_stat_{ES1}, MLG\_stat_{ES1}) = (v_1, v_2)\big) \\
& \textbf{do next}^4 update[LG\_status_{ES1} : g_{LGS}(v_1, v_2)]\Big) \\
\text{ES\_2} \overset{\text{def}}{=}\ & !_{40}\Big(\textbf{next}^{20} \prod_{v\in\{\text{Up, Down}\}} \textbf{when } (command_{ES2} = v) \textbf{ do} \\
& \textbf{next}^4 update[EVcommand_{ES2} : g_{EVC}(v)]\Big) \parallel \\
& !_{40}\Big(\prod_{(v_1,v_2)\in\{0,\cdots,500\}\times\{0,\cdots,500\}} \textbf{when } \big((NLG\_stat_{ES2}, MLG\_stat_{ES2}) = (v_1, v_2)\big) \\
& \textbf{do next}^4 update[LG\_status_{ES2} : g_{LGS}(v_1, v_2)]\Big) \\
\text{ES\_3} \overset{\text{def}}{=}\ & !_{40}\Big(\prod_{(u,v)\in\{\text{extend, retract}\}\times\{0,\cdots,500\}} \textbf{when } \big((EVcommand_{ES3} = u, NLG\_stat_{ES3} = v)\big) \\
& \textbf{do next}^9 update[NLG\_stat_{ES3} : g_{NLGE}(u, v)]\Big) \\
\text{ES\_4} \overset{\text{def}}{=}\ & !_{40}\Big(\prod_{(u,v)\in\{\text{extend, retract}\}\times\{0,\cdots,500\}} \textbf{when } \big((EVcommand_{ES4} = u, MLG\_stat_{ES4} = v)\big) \\
& \textbf{do next}^9 update[MLG\_stat_{ES4} : g_{MLGE}(u, v)]\Big) \\
\text{ES\_5} \overset{\text{def}}{=}\ & !_{20}\Big(\prod_{v\in\{\text{msg\_0, msg\_1}\}} \textbf{when } (RCmsg_{ES6} = v) \textbf{ do next}^4 update[RCmsg_{ES6} : g_{RCG}(v)]\Big)
\end{aligned}
$$

MFD. The MFD partition, which emulates the cockpit lights, displays a light according to $LG\_status \in$ {extended, retracted, maneuvering} the current global status of the landing gear as follows:

$$g_{MFD}(v) = \begin{cases} \text{green} & \text{if } v = \text{extended} \\ \text{yellow} & \text{if } v = \text{maneuvering} \\ \text{no light} & \text{otherwise.} \end{cases}$$

EVCA and EVCB. The EVCA (resp. EVCB) partition is the control software that provides commands for extension/retraction of the gears according to the current state of the handle. Hence the following function.

$$g_{EVC}(v) = \begin{cases} \text{retract} & \text{if } v = \text{Up} \\ \text{extend} & \text{if } v = \text{Down} \end{cases}$$

LGSA and LGSB. The LGSA (resp. LGSB) partition is the control software that informs the pilot about the current global state of the landing sets according to the current positions of the doors and gears that it receives from NLGE and MLGE. As stated in Sect. 4, the full extension process is a five steps sequence of doors unlocking, doors opening, gears up unlocking, gears extension and gears down locking. The retraction process does the opposite actions following the reverse order. Therefore, we express the state of a landing set by a variable $v =\in \{0, \ldots, 500\}$ and interpret its values as follows:

- $\lfloor \frac{v}{100} \rfloor$ expresses the number of extension steps that are fully completed;
- $(v \mod 100)$ expresses the percentage of the current extension step that is completed.

Now, let v_1 be the current state of the nose landing gear and v_2 be the current state of the main landing gears, then we have the following:

$$g_{LGS}(v_1, v_2) = \begin{cases} \text{extended} & \text{if } v_1 = v_2 = 500 \\ \text{retracted} & \text{if } v_1 = v_2 = 0 \\ \text{maneuvering} & \text{otherwise.} \end{cases}$$

NLGE and MLGE. The NLGE partition emulates the actuation of the extension/retraction process according to the electro-valve signal $EVcommand_{ES3} \in$ {extend, retract} and $NLG\_stat_{ES3} \in \{0, \ldots, 500\}$ the current state of the nose landing set.

Now, let $v \in \{0, \ldots, 500\}$ be the current state of the nose landing set, $\pi = 40$ its period, $k = \lfloor \frac{v}{100} \rfloor$ be the number of extension steps that are fully completed and ρ_{k+1} be the duration of the $(k + 1)$th step currently being executed. We remind the reader that the duration ρ_i are given in Table 1. The extension/retraction actuation of the nose landing set is:

$$g_{NLGE}(\text{extend}, v) = \begin{cases} \max\big((k+1) \times 100, v(1 + \frac{\pi}{\rho_{k+1}})\big) & \text{if } v < 500 \\ 500 & \text{otherwise.} \end{cases}$$

$$g_{NLGE}(\text{retract}, v) = \begin{cases} \max\big(k \times 100, v(1 - \frac{\pi}{\rho_{k+1}})\big) & \text{if } v > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Intuitively, each cycle period $\pi = 40$, $\frac{40}{\rho_{k+1}}\%$ of the extension/retraction process is actuated.

RCG. Finally, the RCG partition sends alternately messages msg_0 and msg_1. Hence $g_{RCG}(\text{msg\_i}) = \text{msg\_j}$, where $i \in \{0, 1\}$ and $j = i + 1 \mod 2$.

7.2 Communication Modeling

We proceed with the modeling of TTEthernet concepts. TTEthernet offers three traffic classes: static *time-triggered* (TT) traffic, dynamic traffic with bounded transmission rate known as *rate constrained* (RC) and dynamic unbounded traffic called *best-effort* (BE).

Before going into detail of the modeling, we note that there are some similarities between the concepts of IMA modules and TTEthernet datalinks. Indeed, like a module that enables mixed-criticality applications (partitions) to share safely its computing resources, a datalink enables mixed-criticality frames transmitted over the datalink to share safely its bandwidth. Hence, we follow the same principle as in the previous section. We start by modeling frames on each datalink. Then we model datalink as a composition of frames. Finally, in order to build the complete network, we piece together all datalinks.

We start with TT traffic. Then we present the integration of RC traffic. But we do not consider the no critical BE traffic.

Time-triggered traffic. According to the aerospace standard AS6802 [3], a TT frame f on a datalink $[u, v]$, denoted $f^{[u,v]}$, is fully temporally specified by its *offset time*, *length* and *period*:

$$f^{[u,v]} = (f^{[u,v]} \cdot offset, f \cdot length, f \cdot period).$$

The length and the period of a frame are given a priori and remain fixed along the virtual link. It is the task of the tt-scheduler to assign values to the frame's offset times on all dataflow links belonging to the frame's virtual link. Table 6 gives the virtual links of the landing gear system and the TT-scheduler is shown in Table 7. Note that these three temporal parameters are the same that fully characterize a partition on a module.[2] Hence, there is a perfect similarity between the temporal characterization

[2] Although they might differ in orders of magnitude (duration).

Table 6 Virtual links description

ID	TT/RC	Source	Destination	Period/BAG (ms)	Frame size (ms)
1	TT	KU	{EVCA, EVCB}	80	1
2	TT	EVCA	{NLGE, MLGE}	40	1
3	TT	EVCB	{NLGE, MLGE}	40	1
4	TT	NLGE	{LGSA, LGSB}	40	2
5	TT	MLGE	{LGSA, LGSB}	40	2
6	TT	LGSA	{MFD}	40	1
7	TT	LGSB	{MFD}	40	1
8	RC	RCG	{EVCA, EVCB}	20	3

Table 7 TT-scheduler

Frame	Offset	*var_src*	*var_target*	Datalink
1	10	$command_{ES5}$	$command_{NS}$	[ES_5,NS]
1	15	$command_{NS}$	$command_{ES1}$	[NS,ES_1]
1	15	$command_{NS}$	$command_{ES2}$	[NS,ES_2]
2	25	$EVcommand_{ES1}$	$EVcommand\_1_{NS}$	[ES_1,NS]
2	30	$EVcommand\_1_{NS}$	$EVcommand_{ES3}$	[NS,ES_3]
2	30	$EVcommand\_1_{NS}$	$EVcommand_{ES4}$	[NS,ES_4]
3	25	$EVcommand_{ES2}$	$EVcommand\_2_{NS}$	[ES_2,NS]
3	35	$EVcommand\_2_{NS}$	$EVcommand_{ES3}$	[NS,ES_3]
3	35	$EVcommand\_2_{NS}$	$EVcommand_{ES4}$	[NS,ES_4]
4	15	NLG_stat_{ES3}	NLG_stat_{NS}	[ES_3,NS]
4	20	NLG_stat_{NS}	NLG_stat_{ES1}	[NS,ES_1]
4	20	NLG_stat_{NS}	NLG_stat_{ES2}	[NS,ES_2]
5	15	MLG_stat_{ES4}	MLG_stat_{NS}	[ES_4,NS]
5	25	MLG_stat_{NS}	MLG_stat_{ES1}	[NS,ES_1]
5	25	MLG_stat_{NS}	NLG_stat_{ES2}	[NS,ES_2]
6	10	LG_status_{ES1}	$LG\_status\_1_{NS}$	[ES_1,NS]
6	15	$LG\_status\_1_{NS}$	LG_status	[NS,ES_5]
7	10	LG_status_{ES1}	$LG\_status\_2_{NS}$	[ES_2,NS]
7	20	$LG\_status\_2_{NS}$	LG_status	[NS,ES_5]

of a partition on a module and the temporal characterization of a frame on a datalink. The main difference is that a frame is not computing process as it simply transmits its current value at the source node to the target node. Therefore, we model a TT frame $f^{[u,v]} = (o, \tau, \pi)$ transmitted over the datalink $[u, v]$ by the following generic process

$$f^{[u,v]} \stackrel{\text{def}}{=} !_{\pi}\Big(\mathbf{next}^{o} \prod_{v} \mathbf{when}\ (var\_src = v)\ \mathbf{do}\ \mathbf{next}^{\tau-1} update[var\_target : v]\Big) \qquad (4)$$

Rate constrained traffic. An RC frame is characterized by two fixed temporal parameters: its *length*, that is the duration of the frame's transmission and its *bandwidth allocation gap* (BAG), that is the minimum time interval between two consecutive transmissions of the RC frame The BAG is decided by the system engineer to ensure that there is enough bandwidth allocated for the transmission of the frame on its virtual link and is enforced by the sending ES only. An RC frame is transmitted on a datalink only if there is no highest priority TT frame scheduled for transmission. Moreover, we consider the *timely block* integration [3] of TT and RC traffic which ensures that an RC frame is transmitted only if there is enough time to finish the transmission before a TT frame is scheduled. Hence, we need to compute the *timely block window times* where an RC frame f is blocked by a TT frame over a datalink L.

Let f be an RC frame, τ be its length, L be a datalink and $\mathcal{F}^L_{TT} = \{f_i = (o_i, \tau_i, \pi_i) : 1 \leq i \leq n\}$ be the set of TT frames transmitted over L. We denote $MAF(L) = LCM_{1\leq i\leq n}(\pi_i)$ the major frame of L, that is the least common multiple the periods. Then the timely block window times of f over L is

$$I^L_{tb}(f) = \bigcup_{1\leq i\leq n, 0\leq j\leq \frac{MAF(L)}{\pi_i}-1} \Big[o_i + j \times \pi_i - \tau + 1; o_i + j \times \pi_i + \tau_i\Big] \tag{5}$$

We define an RC frame f over a datalink L as a composition of two processes. The first process adds the unary predicate $busy(L)$ to the store all over the timely block window times of f over L to inform the frame that the network is busy. The second one is a process that continuously checks if there is a new frame (i.e. $new(f, L) = \textbf{true}$) and if the link is free then it sends the frame, sets the $new(f, L)$ to **false**, and if its target is a NS, informs the successor link $succ(f, L)$ of L on the virtual link of f about the arrival of a new frame.

$$\begin{aligned} f^L \stackrel{\text{def}}{=}\ & !_{MAF(L)} \prod_{i\in I^L_{tb}(f)} \textbf{next}^i\ \textbf{tell}\ busy(L) \\ & !\Big(\textbf{when}\ available(f, L)\ \textbf{do}\ \big(update[new(f, L) : \textbf{false}] \parallel \\ & \prod_v \textbf{when}\ (var\_src = v)\ \textbf{do}\ (\textbf{next}^{\tau-1} update[var\_target : v] \parallel Q)\big)\Big) \end{aligned} \tag{6}$$

where

$$Q = \begin{cases} \textbf{0} & \text{if the target of L is an ES} \\ update[new(f, succ(f, L)) : \textbf{true}] & \text{otherwise.} \end{cases}$$

and $c \vdash available(f, L)$ if and only if $c \vdash (new(f, L) = \textbf{true})$ and $c \nvdash busy(L)$.

Data link. Again, we assume that the TT scheduler which is computed and validated offline is well-defined. Therefore, we model each datalink as the parallel composition of the frames that cross over the datalink. For our case study, the modeling of the datalinks from ESs to NS is given in Table 8. In Table 9, we give the model of the link $[NS, ES\_1]$ to illustrate the integration of the RC frame f_8 and the TT

Table 8 Landing gear system model: network links (part 1)

$$
\begin{aligned}
L^{[\text{ES\_5, NS}]} \stackrel{\text{def}}{=}\ & !_{80}\Big(\mathbf{next}^{10} \prod_{v\in\{\text{Up, Down}\}} \mathbf{when}\ (command_{ES5} = v)\ \mathbf{do} \\
& update[command_{NS} : v]\Big) \\
L^{[\text{ES\_1, NS}]} \stackrel{\text{def}}{=}\ & !_{40}\Big(\mathbf{next}^{25} \prod_{v\in\{\text{extend, retract}\}} \mathbf{when}\ (EVcommand_{ES1} = v)\ \mathbf{do} \\
& update[EVcommand\_1_{NS} : v; 1]\Big) \parallel \\
& !_{40}\Big(\mathbf{next}^{10} \prod_{v\in\{\text{extended, retracted, maneuvering}\}} \mathbf{when}\ (LG\_status_{ES1} = v)\ \mathbf{do} \\
& update[LG\_status\_1_{NS} : v]\Big) \\
L^{[\text{ES\_2, NS}]} \stackrel{\text{def}}{=}\ & !_{40}\Big(\mathbf{next}^{25} \prod_{v\in\{\text{extend, retract}\}} \mathbf{when}\ (EVcommand_{ES2} = v)\ \mathbf{do} \\
& update[EVcommand\_2_{NS} : v]\Big) \parallel \\
& !_{40}\Big(\mathbf{next}^{10} \prod_{v\in\{\text{extended, retracted, maneuvering}\}} \mathbf{when}\ (LG\_status_{ES2} = v)\ \mathbf{do} \\
& update[LG\_status\_2_{NS} : v]\Big) \\
L^{[\text{ES\_3, NS}]} \stackrel{\text{def}}{=}\ & !_{40}\Big(\mathbf{next}^{15} \prod_{v\in\in\{0,\ldots,500\}} \mathbf{when}\ (NLG\_stat_{ES3} = v)\ \mathbf{do} \\
& update[NLG\_stat_{NS} : v]\Big) \\
L^{[\text{ES\_4, NS}]} \stackrel{\text{def}}{=}\ & !_{40}\Big(\mathbf{next}^{15} \prod_{v\in\in\{0,\ldots,500\}} \mathbf{when}\ (MLG\_stat_{ES4} = v)\ \mathbf{do} \\
& update[MLG\_stat_{NS} : v]\Big) \\
L^{[\text{ES\_6, NS}]} \stackrel{\text{def}}{=}\ & !_{20}\Big(\mathbf{next}^{5} \prod_{v\in\{\text{msg\_0, msg\_1}\}} \mathbf{when}\ (RCmsg_{ES6} = v)\ \mathbf{do}\ \mathbf{next}^{2} \\
& \big(update[RCmsg_{NS} : v] \parallel update[new(f_8, [\text{NS, ES\_2}]) : \mathbf{true}] \parallel \\
& update[new(f_8, [\text{NS, ES\_2}]) : \mathbf{true}]\big)\Big)
\end{aligned}
$$

Table 9 Landing gear system model: network links (part 2)

$$
\begin{aligned}
L^{[\text{NS, ES\_1}]} \stackrel{\text{def}}{=}\ & !_{80}\Big(\mathbf{next}^{5} \prod_{v\in\{\text{Up, Down}\}} \mathbf{when}\ (command_{NS} = v)\ \mathbf{do} \\
& update[command_{ES1} : v]\Big) \\
& !_{40}\Big(\mathbf{next}^{5} \prod_{v\in\{0,\ldots,500\}} \mathbf{when}\ (NLG\_stat_{NS} = v)\ \mathbf{do} \\
& update[NLG\_stat_{ES1} : v]\Big) \\
& !_{40}\Big(\mathbf{next}^{5} \prod_{v\in\{0,\ldots,500\}} \mathbf{when}\ (MLG\_stat_{NS} = v)\ \mathbf{do} \\
& update[MLG\_stat_{ES1} : v]\Big) \\
& !_{80} \prod_{i\in\{[12;19]\cup[22,26;]\cup[52,59]\cup[22,26;]\cup[52,59]\}} \mathbf{next}^{i}\ \mathbf{tell}\ busy([\text{NS, ES\_1}]) \parallel \\
& !\Big(\mathbf{when}\ available(f_8, [\text{NS, ES\_1}])\ \mathbf{do}\ \big(update[new(f_8, [\text{NS, ES\_1}]) : \mathbf{false}] \parallel \\
& \prod_{v\in\{\text{msg\_0, msg\_1}\}} \mathbf{when}\ (RCmsg_{NS} = v)\ \mathbf{do}\ \mathbf{next}^{2} update[RCmsg_{ES1} : v]\big)\Big)
\end{aligned}
$$

frames f_1, f_4 and f_5 over the same datalink. Note that within the major frame $MAF([NS, ES\_1]) = 80$, we have one single instance of f_1 and two instances of both f_4 and f_5. The remaining datalinks from NS to ESs can be easily modeled following the same principle.

8 Specifying the Requirements

Our objective is to provide a comprehensive framework for the behavioral analysis for IMA systems deployed throughout a TTEthernet network. This section presents two of the most important requirements of avionic systems as well as their specifications in our framework.

8.1 Rate Constrained Traffic Schedulability

One of the most important requirement of avionic systems is the *schedulability requirement* which ensures that the worse case delay of each frame, that is the elapsed time between the sending of a message at the source ES and its arrival at the targets ESs, is within its deadline. Formally, let $WCD(f)$ and $\Delta(f)$ denote respectively the worse case delay and the deadline of f. Then f is schedulable if and only if

$$WCD(f) \leq \Delta(f)$$

A system that does not guarantee the schedulability requirement is not *certifiable*. It is easy to verify the schedulability of TT traffic since they are transmitted based on static schedule tables. However, due to the integration of TT and RC messages in TTEthernet, the schedule of TT messages may severely impacts the schedulability of RC traffic. For example, consider the RC/TT integration shown in Fig. 4. In the

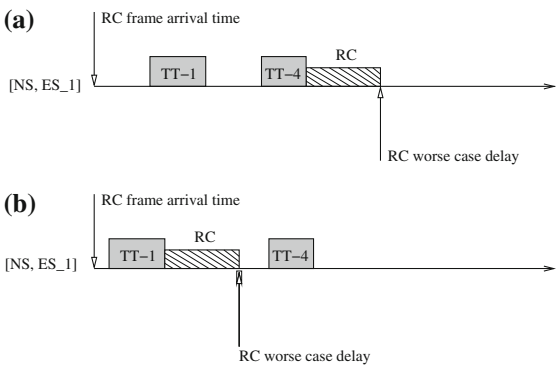


Fig. 4 TT/RC integration

case of (a), both TT frames block the RC frame while in the case of (b) only the first TT frame blocks it leading to a better worse case delay. As stated earlier, the few work [36, 39] that studies the impact the schedule of TT traffic on the schedulability of RC traffic has some limitation as the proposed approaches overestimate the worse case delay. Due to this pessimism, a schedulable system may be rejected when the exact WCD of an RC frame is closed enough to its deadline. Below, we propose a validation approach of the schedulability requirement that eliminates the above limitation.

Specification of the schedulability requirement. The rate constrained traffic schedulability for f_8, the single RC frame of our case study, can be specified in our framework as follows. Let $\Delta(f_8) = 20$ be the deadline of f_8. The landing gear system ensures the schedulability of the rate constrained traffic if and only if for all k in {0; 1},

$$[\![\mathbf{LGS}]\!] \cap (\mathcal{C}\backslash\{\bot\})^{\omega} \neq \emptyset \text{ and } [\![\mathbf{LGS}]\!] \subseteq$$
$$\{\alpha \in \mathcal{C}^{\omega} : \forall i \geq 1, \text{ if } \alpha_i \vdash (RCmsg_{ES6} = msg\_k) \text{ then}$$
$$\exists 0 \leq j \leq \Delta(f_8)\ s.t.\ \alpha_{i+j} \vdash (RCmsg_{ES1} = msg\_k) \vee (RCmsg_{ES2} = msg\_k)\}$$

8.2 Functional Chain End-to-End Delay

Another important property of avionic systems is the *end-to-end delay of functional chains* that involve multiple end systems. The extension process, in our case study, is an example of such a functional chain which involves all the end systems except ES_6. These kind of properties cannot be verified by the existing static approaches to the real-time analysis of TTEthernet mainly for four reasons:

- they often depend on the exact values of the messages communicated between the involved ESs. For example, when the pilot reverses the extension/retraction process, the time for the process to complete will depend on which step of the extension/retraction process is currently executed and how much of the step is already completed;
- the periodic cycles of the involved ESs are not necessary synchronized;
- they may involve both TT and RC traffic.
- they may require a large number of periodic executions of the involved ESs. For example, the total extension time may be up to 12 s while the periods of the involved ESs are either 80 or 40 ms.

Below we give one such requirement for the landing gear system introduced in [10] and show how it can be easily specified and validate in our approach.

Requirement [10] when the command line is working (i.e. in the absence of failure), if the landing gear command handle has been pushed Down and stays Down, then the gears will be locked down and the doors will be seen closed less than 15 s after the handle has been pushed.

Specification of the end-to-end delay requirement. Let $\Delta_{lgs} = 15000$ ms be the above 15 s threshold of the gear extension requirement. Then the landing gear system ensures the end-to-end delay requirement of the gear extension process if and only if

$$[\![\mathbf{LGS}]\!] \cap (\mathcal{C} \setminus \{\bot\})^{\omega} \neq \emptyset \text{ and } [\![\mathbf{LGS}]\!] \subseteq \{\alpha \in \mathcal{C}^{\omega} : \text{ if } \forall i \leq \Delta_{lgs}, \alpha_i \vdash (handle = \text{Down}) \text{ then } \exists 0 \leq j \leq \Delta_{lgs}\ s.t.\ \alpha_j \vdash (light = \text{green})\}$$

These properties illustrate the applicability of our framework to the specification of important avionic requirements.

9 Concluding Remarks

In this paper we have presented the TTCC calculus, a simple and elegant CCP-based calculus for the analysis of real-time systems tailored for the time-triggered processes. We have shown how the denotational semantics can be conveniently used as specification language of the requirements of avionic systems. In particular, three generic TTCC processes are defined for the modeling of the main concepts of IMA and TTEthernet: partitions, time-triggered frames and rate constrained frames. We have also illustrated the usefulness of our approach through the specification of the schedulability and the end to end delay of functional chains properties, two of the most important requirements of avionic systems.

In future work, we plan to develop a proof system for denotational inclusion of the form $P \subseteq F$ establishing that a given process P satisfies a given property (or requirement) F, following the lines proposed in [24] for linear-temporal properties of TCC processes. We also expect to take advantage of the prototype tool developed by the AVISPA Research Group[3] for an automatic verification of avionic systems modeled as TTCC processes. We finally plan to develop a general methodology and an associated tool for translating AADL [15] (Architecture Analysis and Design Language) and Annexes specification (e.g. [37]) into the TTCC process calculus to allow a comprehensive analysis for avionic systems specified in this aerospace standard for model-based specification of complex real-time embedded systems.

Acknowledgments This work is supported by the CRIAQ-NSERC RDC Project VerITTAS (Verification and Integration of multi-critical Time-Triggered Avionics Systems) (AVIO613) No. 435325-12.

[3] URL: http://avispa.univalle.edu.co:8080/REACT-PLUS/.

References

1. Integrated Modular Avionics (IMA). Aeronautical Radio, Inc., ARINC 653 (2009)
2. Avionics Full Duplex Switched Ethernet (AFDX). Aeronautical Radio, Inc., ARINC 664, Part 7, (2010)
3. Time-Triggered Ethernet (TTEthernet). SAE Aerospace, AS6802 (2011)
4. Adnan, M., Scharbarg, J., Ermont, J., Fraboul, C.: Model for worst case delay analysis of an AFDX network using timed automata. In: Proceedings of 15th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2010, pp. 1–4, Bilbao, Spain, 13–16 Sept 2010. IEEE (2010)
5. Al Sheikh, A., Brun, O., Hladik, P.-E., Prabhu, B.J.: Strictly periodic scheduling in IMA-based architectures. Real-Time Syst. **48**(4), 359–386 (2012)
6. Alpuente, M., del Mar Gallardo, M., Pimentel, E., Villanueva, A.: An abstract analysis framework for synchronous concurrent languages based on source-to-source transformation. Electr. Notes. Theor. Comput. Sci. **206**, 3–21 (2008)
7. Bauer, H., Scharbarg, J., Fraboul, C.: Applying and optimizing trajectory approach for performance evaluation of AFDX avionics network. In: Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2009, pp. 1–8. IEEE (2009)
8. Bauer, H., Scharbarg, J.-L., Fraboul, C.: Improving the worst-case delay analysis of an afdx network using an optimized trajectory approach. IEEE Trans. Ind. Inform. **6**(4), 521–533 (2010)
9. Behjati, R., Yue, T., Nejati, S., Briand, L.C., Selic, B.: Extending SysML with AADL concepts for comprehensive system architecture modeling. In: France, R.B., Küster, J.M., Bordbar, B., Paige, R.F. (eds.) Lecture Notes in Computer Science, ECMFA, vol. 6698, pp. 236–252. Springer (2011)
10. Boniol, F., Wiels, V.: The landing gear system case study. In: Boniol, F., Wiels, V., Ait Ameur, Y., Schewe, K.-D (eds) ABZ 2014: The Landing Gear Case Study, Communications in Computer and Information Science, vol. 433, pp. 1–18. Springer International Publishing (2014)
11. Bouillard, A., Jouhet, L., Thierry, E.: Tight performance bounds in the worst-case analysis of feed-forward networks. In: INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, pp. 1316–1324. IEEE (2010)
12. Boyer, M., Fraboul, C.: Tightening end to end delay upper bound for AFDX network calculus with rate latency FCFS servers using network calculus (regular paper). In: IEEE International Workshop on Factory Communication Systems (WFCS), pp. 11–20. IEEE, May 2008. www.ieee.org/
13. Brémond-Grégoire, P., Lee, I., Gerber, R.: Acsr: an algebra of communicating shared resources with dense time and priorities. In: Best, E. (ed.) CONCUR, Lecture Notes in Computer Science, vol. 715, pp. 417–431. Springer (1993)
14. de Boer, F.S., Gabbrielli, M., Meo, M.C.: A timed concurrent constraint language. Inf. Comput. **161**(1), 45–83 (2000)
15. Feiler, P.H., Gluch, D.P.: Model-Based Engineering with AADL—An Introduction to the SAE Architecture Analysis and Design Language. SEI Series inSoftware Engineering. Addison-Wesley, Boston (2012)
16. Hamadou, S., Mullins, J., Chareton, C., Gherbi, A.: Specifying avionic embedded systems by denotations of the time-triggered constraint-based calculus. In: 2015 IEEE International Conference on Information Reuse and Integration, IRI 2015, pp. 303–310, San Francisco, CA, USA, 13–15 Aug 2015. IEEE (2015)
17. Hamadou, S., Mullins, J., Gherbi, A., Beji, S.: A time-triggered constraint-based calculus for avionic systems. In: 18th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, ISORC 2015, pp. 60–67 (2015)

18. Hentenryck, P.V., Saraswat, V.A., Deville, Y.: Design, implementation, and evaluation of the constraint language cc(FD). In: Podelski, A. (ed.) Constraint Programming. Lecture Notes in Computer Science, vol. 910, pp. 293–316. Springer (1994)
19. Kopetz, H., Bauer, G.: The time-triggered architecture. Proc. IEEE **91**(1), 112–126 (2003)
20. Le Boudec, J.-Y., Thiran, P.: Network Calculus: A Theory of Deterministic Queuing Systems for the Internet. Springer-Verlag, Berlin (2001)
21. Lescaylle, A., Villanueva, A.: A tool for generating a symbolic representation of TCCP executions. Electr. Notes Theor. Comput. Sci. **246**, 131–145 (2009)
22. Li, X., Scharbarg, J.L., Fraboul, C.: Improving end-to-end delay upper bounds on an afdx network by integrating offsets in worst-case analysis. In: ETFA, 2010, pp. 1–8, Bilbao, Spain, 13–16 Sept 2010. IEEE (2010)
23. Mikucionis, M., Larsen, K.G., Rasmussen, J.I., Nielsen, B., Skou, A., Palm, S.U., Pedersen, J.S., Hougaard, P.: Schedulability analysis using Uppaal: Herschel-planck case study. In: Margaria, T., Steffen, B. (eds.) ISoLA (2), Lecture Notes in Computer Science, vol. 6416, pp. 175–190. Springer (2010)
24. Nielsen, M., Palamidessi, C., Valencia, F.D.: Temporal concurrent constraint programming: denotation, logic and applications. Nord. J. Comput. **9**(1), 145–188 (2002)
25. Olarte, C., Rueda, C., Valencia, F.D.: Models and emerging trends of concurrent constraint programming. Constraints **18**(4), 535–578 (2013)
26. Philippou, A., Lee, I., Sokolsky, O.: Pads: an approach to modeling resource demand and supply for the formal analysis of hierarchical scheduling. Theor. Comput. Sci. **413**(1), 2–20 (2012)
27. Saha, I., Roy, S.: A finite state modeling of AFDX frame management using spin. In: Brim, L., Haverkort, B.M., Leucker, M., van de Pol, J. (eds.) Formal Methods: Applications and Technology, 11th International Workshop, FMICS 2006 and 5th International Workshop PDMC 2006, Bonn, Revised Selected Papers, Lecture Notes in Computer Science, vol. 4346, pp. 227–243. Springer (2006)
28. Samarjit, L.T., Thiele, L., Chakraborty, S., Naedele, M.: Real-time calculus for scheduling hard real-time systems. In: ISCAS, pp. 101–104 (2000)
29. Sangiorgi, D.: Introduction to Bisimulation and Coinduction. Cambridge University Press, New York, NY, USA (2011)
30. Saraswat, V.A.: Concurrent constraint programming, ACM Doctoral dissertation awards. MIT Press (1993)
31. Saraswat, V.A., Jagadeesan, R., Gupta, V.: Foundations of timed concurrent constraint programming. In: LICS, pp. 71–80. IEEE Computer Society (1994)
32. Saraswat, V.A., Jagadeesan, R., Gupta, V.: jcc: integrating timed default concurrent constraint programming into java. In: Moura-Pires, F., Abreu, S. (eds.) EPIA, Lecture Notes in Computer Science, vol. 2902, pp. 156–170. Springer (2003)
33. Saraswat, V.A., Rinard, M.C., Panangaden, P.: Semantic foundations of concurrent constraint programming. In: Wise, D.S. (ed.) POPL, pp. 333–352. ACM Press (1991)
34. Smolka, G.: Concurrent constraint programming based on functional programming (extended abstract). In: Hankin, C. (ed.) ESOP, Lecture Notes in Computer Science, volume 1381, pp. 1–11. Springer (1998)
35. Sokolsky, O., Lee, I., Clarke, D.: Process-algebraic interpretation of aadl models. In: Kordon, F., Kermarrec, Y. (eds.) Ada-Europe, Lecture Notes in Computer Science, vol. 5570, pp. 222–236. Springer (2009)
36. Tamas-Selicean, D., Pop, P., Steiner, W.: Timing analysis of rate constrained traffic for the ttethernet communication protocol. In: IEEE 18th International Symposium on Real-Time Distributed Computing, ISORC 2015, Auckland, New Zealand, pp. 119–126, 13–17 April, 2015. IEEE Computer Society (2015)

37. Tiyam, R., Kouhen, A.E., Gherbi, A., Hamadou, S., Mullins, J.: An extension for AADL to model mixed-criticality avionic systems deployed on IMA architectures with TTEthernet. In: Delange, J., Feiler, P.H. (eds.) Proceedings of the First International Workshop on Architecture Centric Virtual Integration co-located with the 17th International Conference on Model Driven Engineering Languages and Systems, ACVI@MoDELS 2014, Valencia, Spain, 29 Sept 2014. CEUR Workshop Proceedings, vol. 1233, CEUR-WS.org (2014)
38. Tămaş-Selicean, D., Pop, P., Steiner, W.: Design optimization of TTEthernet-based distributed real-time systems. Real-Time Syst. 1–35 (2014)
39. Zhao, L., Xiong, H., Zheng, Z., Li, Q.: Improving worst-case latency analysis for rate-constrained traffic in the time-triggered ethernet network. IEEE Commun. Lett. **18**(11), 1927–1930 (2014)

Case Indexing by Component, Context, and Encapsulation for Knowledge Reuse

Asmaa Chebba, Thouraya Bouabana-Tebibel, Stuart H. Rubin and Kadaouia Habib

Abstract Ontology is commonly defined as a formal specification of knowledge conceptualization in a consensual form. OWL (Web Ontology Language) is the most used and expressive ontology description language that supports handling and reasoning. However, it lacks expressivity for some specific requirements. We first proposed, in a previous work, to enhance OWL expressivity with two new relevant notions for reuse. These notions concern concepts contextualization and encapsulation. We propose, in this paper, to provide representation criterions for these two notions in a case-based reasoning (CBR) system. In of a problem-solving experience, in a given domain, includes a problem part and a solution part—each of which requires an appropriate representation. We propose specific indexing to represent these two parts by highlighting the indexing specificity for contextualization and encapsulation of concepts.

Keywords Case indexing · Context · Design knowledge · Knowledge representation · List of assemblies · Ontology · OWL

A. Chebba (✉) · T. Bouabana-Tebibel · K. Habib
École nationale Supérieure d'Informatique, LCSI Laboratory, Algiers, Algeria
e-mail: a_chebba@esi.dz

T. Bouabana-Tebibel
e-mail: t_tebibel@esi.dz

K. Habib
e-mail: k_habib@esi.dz

S.H. Rubin
Space and Naval Warfare Systems Center Pacific, San Diego 92152-5001, USA
e-mail: stuart.rubin@navy.mil

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_5

1 Introduction

Conceptual design refers to the activity of describing ideas in a symbolic format. It is a major task that impacts the quality and the overall characteristics of the created product [1]. It generates the highest stage cost in terms of effort and time. In the present work, we aim to facilitate this task for designers by deploying a reuse strategy.

Conceptual design involves different kinds of knowledge, which is not easy to arrange and use [2]. Representation of design knowledge, in conceptual design, is associated with some evaluation criteria among which are space savings, capacity of reuse, rapidity of search, and efficiency of reasoning. We use ontology to represent knowledge included in design related to a specific domain. Ontology provides high-quality expressiveness and formal specification—thus enabling reasoning and computation. It supports most of the desired knowledge representation using a well-defined syntax and semantics. It also supports sharing and reuse and is very popular, which promotes its advancement.

However, ontologies remain debatable in terms of how they should be used in computer science [3]. Many works, including those of Gómez-Pérez [4], define the main constructs of an ontology, among which are concepts, relations, functions, axioms, and instances. But, the practical syntax and semantics of an ontology are rather defined through descriptive languages. Ontologies may be mapped onto such formalisms as the predicate calculus.

Some of these languages are based on first order logics, like KIF, or frames combined with first order logics such as Ontolingua, OCML, and FLogic. Other languages are based on DL like Loom. The most popular ontology description language is currently OWL (Web Ontology Language) [5]. OWL is the current standard that came in three flavors: OWL Lite, OWL DL, and OWL Full. It has been standardized by the W3C consortium and is widely accepted within the Semantic Web community, especially for its logical basis—SHOIN(D) for OWL-DL. The Web-Ontology Working Group has taken DAML+OIL [6] features as the main input for developing OWL and has proposed the first specification of this language by extending a restricted use of Resource Description Framework (RDF) [7] and RDF Schema (RDFS) [8].

We use OWL-DL to implement the ontological model. The latter mainly supports the modeling of concepts linked with relations. No restriction is given on the type of the association ends (concepts) so far, which leaves room for different definitions for this type. However, OWL excludes the definition of groups (subsets) of concepts as composite elements at the association end. A concept may be associated with any other concept without any restriction on its matches, which considerably reduces the modeling expressivity. OWL defines, for every relation called ObjectProperty, a domain and a range. The domain is the main concept of the relation at one association end; whereas, the range is the concept at the opposite association-end. Multiple ranges may be associated to a given ObjectProperty. There is so far no simple way to specify a range as a list of range groups, rather than a simple range. Exclusive assignments are, indeed, not supported at the concept level.

We proposed in [9] to enhance OWL with a new range type. The latter supports the notion of predefined groups of concepts placed within a list, rather than simple and unstructured concepts. This range type, that we call AssemblyRange, defines the only predefined groups of ranges that can be associated to the ObjectProperty. Such modeling enhances the language expressivity. It introduces the modeling of restrictive assignments, rather than everything-allowed assignments. It evidences to be of major interest in reuse and integration, where a component may be of polymorphic form, thus compacting the ontological representation. Context and encapsulation are examples of this type of polymorphic knowledge. The context is the external environment of a concept; while, encapsulation defines all of the elements incorporated in a composite concept. The term polymorphic is related to these two notions and refers to the fact that contexts and encapsulations can vary for a given concept depending on its use case.

We propose, in this paper, to integrate the extended ontological model into a case-based reasoning (CBR) system [10]. Conceptual design is the domain of application. We, are especially interested in structuring the proposed ontology according to CBR principles. A case indexing architecture is developed for this purpose. We, furthermore offer to users the capability to specify information about the design (solution) to be retrieved in addition to describing the problem of concern. This involves new case indexing criterions, among which are the context and encapsulation representative elements.

The remainder of this paper is structured as follows. Section 2 presents works on knowledge representation, especially those related to possibilistic knowledge and situation-dependent knowledge. Section 3 highlights the issues raised through the specific domain of cooling systems. Section 4 puts emphasis on the AssemblyRange construct added on OWL for enabling the representation of polymorphic knowledge. Section 5 presents the case indexing approach using the notions of context and encapsulation. Finally, we present in Sect. 6 some results on the construct use.

2 Related Works

Design knowledge has known various definitions while inciting a broad debate on its meaning [11–13]. It can be deduced from [14] that all knowledge that leads to the creation of design is design knowledge in a given domain. Different representation models have been proposed to describe knowledge. We find, among others, logical representations [15], graphical representations as semantic networks [16], frames [17], conceptual graphs [18], and hybrid representations.

Representation of situation-dependent knowledge has been treated in previous works, each of which is motivated by a special aspect of knowledge. Context dependency is a major issue that was treated in [19] (i.e. fuzzy knowledge). The authors present a model where knowledge is processed with respect to its surrounding context. The representation is structured in layers where each layer presents a different context. To retrieve knowledge, a context selector module is used to choose the

appropriate layer. Another example concerns the description language OWL of the ontology model. The *DataRange* construct, added to the new version OWL2 [20], allows for the definition of value assignment to attributes. For this purpose, the range is defined as a list of values at the individual level. The corresponding attribute (DatatypeProperty) gets one element of the list as its value.

Context notion has also been addressed in the literature [21] and was associated with diverse definitions. Interoperability of exchanges are treated in [22], where the authors use ontology in order to share information about the services provided by communicating organizations. Authors in [23] argue that no attempt was made to design and develop generic context models. In addition, approaches not tied to specific application domains usually represent a limited amount of contextual information types. Thus, they intend to propose uniform context models, representation, and query languages, as well as reasoning algorithms that facilitate context sharing and interoperability of applications. As a final example, we cite the work presented in [24], where Context OWL (C-OWL) is proposed. C-OWL is a language whose syntax and semantics have been obtained by extending the OWL syntax and semantics to allow for the representation of contextual ontologies. The work is motivated by the fact that OWL cannot model certain situations like the directionality of information flow, interpretation of ontologies in local domains, and context mappings. In our work, we consider the context notion at its finer level of granularity, on concepts—differentiating it from the listed previous works.

On the other hand, knowledge indexing is the process of organizing the knowledge base to support knowledge key features. Retrieval and retaining, in the CBR systems for instance, are the most important processes in indexing [25]. Researches are still significantly discussing the indexing methods. Previous studies have been conducted on the subject. For example, in [26], a k-NN clustering algorithm is proposed, where generalization and fuzzification of similar cases is in the basis for improving the performance of retrieval as well as reducing the size of the case-base. Also, fuzzy indexing is an approach, which has been successfully applied by previous researchers in different applications, such as presented by the works cited in [27, 28].

3 Need for Polymorphic Knowledge Representation

Representation of design knowledge is a critical task, since the defined knowledge is complex, diverse, and varies in type and granularity. Besides, the reasoning process imposes a pertinent, complete, and well-structured representation of knowledge.

3.1 A Domain-Specific System

We illustrate the issues raised on knowledge representation through the specific domain of cooling systems. We focus on the construction of a little part of the ontology, considering only a typical two-stage refrigeration system. Multistage refrigera-

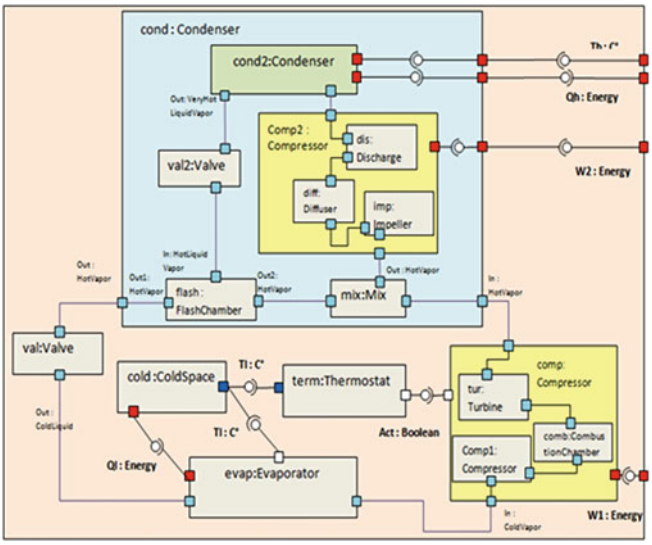


Fig. 1 IBD of a refrigeration system design

tion systems are widely used, where ultralow temperatures are required, but cannot be economically obtained through the use of a single-stage system. They employ two or more compressors, connected in series, in the same refrigeration system. Figure 1 depicts a two-stage refrigeration system modeled using the Internal Block Diagram (IBD) of SysML [29]. In the latter, all of the component in/out ports are properly connected to ensure the flow of fluid refrigerant through the whole of the system. The fluid refrigerant absorbs and removes heat from the space to be cooled and then radiates that heat elsewhere. The system is composed of:

- **Compressor**: **transforms** the low pressure vapor drawn from the evaporator to high pressure vapor.
- **Condenser**: transforms the high pressure refrigerant vapor to liquid state.
- **Expansion Valve**: **reduces** the liquid refrigerant pressure sufficiently to allow the liquid vaporization.
- **Evaporator**: **enters** the low pressure refrigerant vapor enters the evaporator to absorb heat from the Cold Space.
- **Thermostat**: regulates the temperature inside the system.
- **Flash Chamber**: allows the separation of saturated vapor and saturated liquid.
- **Mix**: mixes the two vapors from compressor and flash chamber.
- **Cold Space**: is the internal space of the refrigerator.

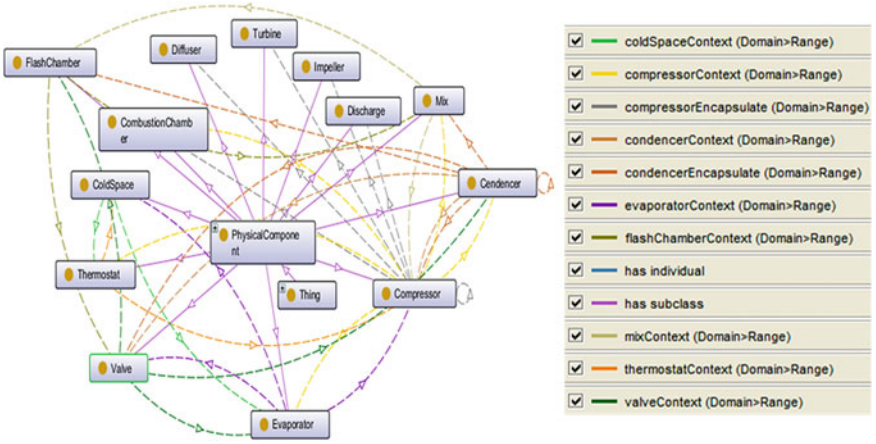


Fig. 2 Visualizing the cold systems ontology with ontograph

3.2 Polymorphic Knowledge

In conceptual design, the various forms of knowledge need be appropriately modeled. We note, on the refrigeration system of Fig. 1, two types of knowledge whose representation calls for an elaborated form. The first one refers to the context of a concept that may change depending on the latter's use. The second one concerns the encapsulation power of a concept that can also change from one situation to another. We refer to this knowledge as polymorphic since the concept definition changes according to the concept, current context, and/or encapsulation. For a specific situation, only one context and/or encapsulation is applicable. However, OWL does not provide constructs enabling the description of such a polymorphic knowledge. Hence, changes on the basic language are necessary to properly meet these specific needs.

Figure 2 gives a view of the ontology describing the refrigeration system, based on the current OWL constructs. The IBD blocs of Fig. 1 are represented by ontological concepts. The visualized graph is obtained from the graphical plug-in ontoGraph of Protégé [30] editor. As shown (refer also to the attached legend), the notion of context, for a given concept, called class in OWL, is not well expressed. An ObjectProperty "context" may be created in order to index the surrounding components of the class; but, it will point the latter in a flat form; no class grouping could be considered for describing the modeled contexts and encapsulations.

For example, let us consider the *Compressor* component. The range of its ObjectProperty *CompressorContext* is pointing all components surrounding the *Compressor* without any distinction. This does not fully match knowledge contained in the design. Indeed, we have to specify that there are three possible contexts for the *Compressor* component. The first context should index the *Condenser, Thermostat,* and *Evaporator* components; whereas, the second context indexes the *Condenser* and

Mix components; and, the third one indexes the *CombustionChamber*. Such expressivity limitations are also encountered for encapsulation. We define, in the next two subsections, the idea of polymorphic knowledge representation.
(1) *Context*: context refers to the external environment of a given concept. It defines all interactions among the concept and its surroundings. Indeed, a concept can be used to express different "things" and has then different characteristics based on its current situation.

In the refrigeration system, the concept *Valve* is used two times for two different situations. We note that connections, roles, etc., to this concept are different from one situation to another, which makes the concept use different. To represent this polymorphic knowledge, using ontology, we need to introduce the notion of context by listing all situations defining a concept.
(2) *Encapsulation*: the level of abstraction of a concept may be high or low. It is low in case the concept describes an atomic knowledge. The concept is then called *simple*. It is high when the concept encapsulates other concepts. The concept is thus called composite or aggregate. Thus, concept representation must support different levels of granularity to allow for the modeling of multiple levels of encapsulation. In addition, components encapsulated by a concept differ according to the use of this concept. These changes symbolize polymorphic knowledge, which need be appropriately modeled by listing all possible aggregations for a concept.

The *Compressor* component is an example of aggregate concept. This latter has two different encapsulations according to the refrigeration system design shown in Fig. 1. In its first use, the *Compressor* contains the sub-components: *Discharge, Diffuser,* and *Impeller*. In its second use, it contains: *CombustionChamber, Compressor,* and *Turbine*.

4 Definition of the AssemblyRange

The various contexts and encapsulations that correspond to a given concept are modeled in our work by means of an assembly list, specified by the new range type, called *AssemblyRange*. This element allows for listing all assembly possibilities with regard to context and encapsulation. In the context case, an assembly is a set of concepts surrounding the concept in question. For encapsulation, the assembly is the set of sub-concepts composing the concept of interest. OWL does not support the specification of such types of knowledge. It provides the *list* element, but this latter does not apply to the ObjectProperty and lists only simple elements rather than assembly ones. The AssemblyRange introduces a new range type for ObjectProperty. We will first describe its syntax according to the RDF/XML format. Next, we show its integrity at the semantic level by integrating it in the OWL logic basis.

4.1 AssemblyRange OWL Syntax and Semantics

We need to perform some changes on the language syntax to express the new representative element. First, recall that a relation in OWL is modeled through the ObjectProperty element and attributes are represented by the DatatypeProperty. A domain and a range are specified for both. The domain refers to the concept to which the relation or the attribute is attached (the first end). As to the range, it represents the value type for the attribute or the concept pointed to by the relation (the second end). For instance, the relation *hasComponent*, between the *Compressor* and *Turbine* classes, has the *Compressor* as domain and *Turbine* as the range. As to the attribute *size* of the class *Compressor*, it has the *Compressor* class as domain and the data-type Real as the range, since a size value is numeric.

In OWL, the list of assembly specifications will appear at the range of ObjectProperty. The context and encapsulation of a given class are respectively specified by the ObjectProperty elements *context* and *encapsulation*, where the class represents the ObjectProperty domain and the assemblies list reflects the ObjectProperty range.

We are inspired, for this specification, by the OWL DataRange defined for DatatypeProperty. DataRange enables the specification of the DatatypeProperty range as a list of possible values, where only one value is assigned to the attribute at a given time. This exclusivity is guaranteed by the use of the "oneOf" construct as defined for lists in OWL.

In the spirit of the DataRange definition, for the DatatypeProperty, we add to OWL the AssemblyRange construct, which enables specification of a list of possible assemblies for ObjectProperty.

We define the AssemblyRange syntax according to the RDF syntax as shown in Fig. 3. To express an AssemblyRange, the elements *rdf:List* and rdf:*parseType* = *"Collection"* of the RDF syntax are used. *Rdf:List* lists the context assemblies and encapsulation assemblies and *rdf:parseType*= *"Collection"* describes the structure of each assembly. We show a list with two assemblies. #Class1 and #Class2 are classes that compose the first assembly. #Class3 and #Class4 are classes that compose the second assembly. The element *owl: oneOf* is also used to signify that only one assembly is true at a time.

In addition, at the individual's level, the syntax in Fig. 4 enables representation of the chosen assembly for the specific situation. Only one assembly of the previous defined list is valid for a specific situation.

Individuals are specified from only one assembly using individual syntax: "individualID1" is of type Class1 and "individualID1" of type Class2.

4.2 AssemblyRange Logic Basis

OWL is a restricted set, using RDF syntax, whose semantics are defined by SHOIN(D). SHOIN(D) logic is a description logic (DL) that assures the consistency and integrity of OWL semantics. Let us define the following for SHOIN(D).

```
<rdf:range>
  <owl: AssemblyRange>
      <owl:oneOf>
          <rdf:List>
          <rdf:first rdf:parseType="Collection">
             <owl:Class rdf:about="#Class1"/>
             <owl:Class rdf:about="#Class2"/>
                    …
          </rdf:first>
          <rdf:rest>
          <rdf:List>
          <rdf:first rdf:parseType="Collection">
          <owl:Class rdf:about="#Class3"/>
          <owl:Class rdf:about="#Class4"/>
                    …
          </rdf:first>
          <rdf:rest rdf:resource="&rdf;nil"/>
          </rdf:List>
          </rdf:rest>
          </rdf:List>
        </owl:oneOf>
  </owl: AssemblyRange>
<rdf:range>
```

Fig. 3 AssemblyRange OWL syntax

```
<owl:NamedIndividual rdf:about="individualID">
        <relationName rdf:resource="individualID1"/>
        <relationName rdf:resource="individualID2"/>
        …
    </owl:NamedIndividual>
```

Fig. 4 AssemblyRange at the individual's level

I an interpretation that provides:

Δ^I a non-empty set called the domain of discourse, which represents existing entities (individuals) in the "world" that I presents.

$.^I$ the interpretation function that connects the vocabulary elements to Δ^I and,

- For each individual name a, a corresponding individual $a^I \in \Delta^I$
- For each concept C, a corresponding set $C^I \subseteq \Delta^I$
- For each abstract property (relation) R, a corresponding set $R^I \subseteq \Delta^I \times \Delta^I$

Besides these specifications, a set D of concrete datatypes is considered. It gives its name (D) to the SHOIN(D) logic. Each concrete datatype, $d \in D$ is associated with a set $d^D \in \Delta^I_D$, where Δ^I_D is the domain (values) of all datatypes (integer, string, etc.). The domain of interpretation Δ^I_D is disjoint from the domain of interpretation of the concept language Δ^I.

Table 1 shows the ObjectProperty definition [31] in OWL and SHOIN(D) logic.

We need to perform modifications on SHOIN(D) logic for the abstract property (ObjectPropety in OWL) to add the assembly list to OWL. These modifications con-

Table 1 Syntax and semantic of ObjectProperty

OWL abstract syntax	DL syntax	DL semantics
ObjectProperty		
(∪ super (R_1)… super (R_n)	$R \sqsubseteq R_i$	$R^I \subseteq R_i^I$
Domain (C_1)… Domain (C_n)	$\geq 1R \sqsubseteq C_i$	$R^I \subseteq C_i^I \times \Delta^I$
Range (C_1)… Range (C_n)	$T \sqsubseteq \forall R.C_i$	$R^I \subseteq \Delta^I \times C_i^I$
[InverseOf(R_0)]	$R \equiv R_0^-$	$R^I = (R_0^I)^-$
[Symmetric]	$R \equiv R^-$	$R^I = (R^I)^-$
[Functional]	$T \sqsubseteq \leq 1R$	R^I is functional
[InverseFunctional]	$T \sqsubseteq \leq 1R^-$	$(R^I)^-$ is functional
[Transitive])	Tr(R)	$R^I = (R^I)^+$

cern more specifically the range of the abstract property, which we extended to permit the specification of an assembly list, rather than only classes. The AssemblyRange DL syntax and DL semantics are defined as follows.

4.2.1 AssemblyRange DL Syntax

As stated in Table 1, the DL syntax for the ObjectProperty range is of the form: $T \sqsubseteq \forall R.C_i$—interpreted as $R^I \subseteq \Delta^I \times C_i^I$.

For the proposed AssemblyRange, the DL syntax is defined as follows.

AssemblyRange(AsmbList)➔ $T \sqsubseteq \forall R.AsmbList$, where AsmbList is a list of assemblies defined as: AsmbList ≡ {$Asmb_1$, …, $Asmb_n$}. Each $Asmb_i$ represents an assembly of a set of classes. Subsequently, let us consider $Asmb_i \equiv (C_1 \ldots C_m)$, C_i is a class.

Integration of a new element in OWL requires changes to the axiom abstract syntax [32] of OWL-DL. Changes to the ObjectProperty axiom definition are as follows.

axiom ::= '*DatatypeProperty('**datavaluedPropertyID** ['*Deprecated*']{***annotation***}
{'*super('**datavaluedPropertyID**')*'}['*Functional*']
{'*domain('**classID**'')*'} {'*range('**dataRange**')*'}')'
|'*ObjectProperty('**individualvaluedPropertyID** ['*Deprecated*'] {***annotation***}
{'*super('**individualvaluedPropertyID**')*'}
['*inverseOf('**individualvaluedPropertyID** ')*'] ['*Symmetric*']
['*Functional*' | '*InverseFunctional*' | '*Functional*' '*InverseFunctional*' | '*Transitive*']
{'*domain('**classID**'')*'} {'*range('**classID**')*'} ['*range('**AssemblyRange** ')*']')'
|'*AnnotationProperty('**annotationPropertyID** {***annotation***}')'
|'*OntologyProperty('**ontologyPropertyID** {***annotation***}')'
dataRange ::= ***datatypeID*** |'*rdfs:Literal*'
AssemblyRange ::= 'OneOf('{'('{***classID***}')'}')'

4.2.2 AssemblyRange DL Semantics

As stated in Table 1, the DL semantics defined for the range of ObjectProperty is: $R^I \subseteq \Delta^I \times C_i^I$ where C_i is the range of the ObjectProperty, R.

The DL semantics for the proposed AssemblyRange is hence: $R^I \subseteq \Delta^I \times \{Asmb_1{}^I \ldots Asmb_n{}^I\}$ with $Asmb_i{}^I ::= \cup C_{ij}{}^I$ and $\exists 1 C_{ij}{}^I.j \in [1, m]$, with m the number of concepts in $Asmb_i$. Note that the interpretation '{}' refers to the selection of only one element of the list. In addition, $\exists 1 C_{ij}{}^I$ refers to the fact that it exists for at least one individual for each class type included in the selected assembly.

We also define the semantics of the ObjectProperty associated with the AssemblyRange with regard to the inclusion (hierarchy), transitivity, inverse, symmetry, functional, inverse functional, equivalence, and disjointness, as stated for the range as shown in Table 1.

We define, for the inclusion, the following semantics.

Let R_1 be an abstract property and let R_0 be the top abstract property, which is the root for all abstract properties. Then: $R_1 \subseteq R_0$. Also, let R_i be an abstract property, C_i a class, and $AsmbList_i$ an assembly list, $\forall i \in N$.

We have $R_2 \subseteq R_1$ and $R_1 \subseteq R_0 \rightarrow R_2 \subseteq R_0$ since each abstract property is included in the root or top property by definition; hence, let us prove that for R_i abstract property ($i \in N$), if $R_n \subseteq R_{n-1}$ and $R_{n-1} \subseteq R_{n-2}$ then $R_n \subseteq R_{n-2}$:

We first define C_i^I as domain of R_i and $AsmbList_i$ as its range. In addition, we define $AsmbList_i \subseteq AsmbList_j \equiv \{\forall Asmb \in AsmbList_i \mid Asmb \in AsmbList_j\}$, where AsmbList is the list of assemblies and Asmb is an assembly.

We have: $R_n \subseteq R_{n-1} \rightarrow R_n{}^I \subseteq R_{n-1}{}^I$

$$\text{This implies: } C_n^I \subseteq C_{n-1}^I \text{ and } AsmbList_n \subseteq AsmbList_{n-1} \ldots \quad (1)$$

On the other hand, $R_{n-1} \subseteq R_{n-2} \rightarrow R_{n-1}{}^I \subseteq R_{n-2}{}^I$

$$\text{This implies: } C_{n-1}^I \subseteq C_{n-2}^I \text{ and } AsmList_{n-1} \subseteq AsmList_{n-2} \ldots \quad (2)$$

From (1) and (2), and from the inclusion proven on classes and set-features, we have:

$C_n^I \subseteq C_{n-2}^I$ and $AsmbList_n \subseteq AsmbList_{n-2} \rightarrow R_n{}^I \subseteq R_{n-2}{}^I$

Thus, the inclusion, which is the basis for hierarchy, is supported by abstract properties defined with assemblies list.

For transitivity, the semantics are as follows. The abstract property R is transitive (Tr(R)) if $R^I = (R^I)^+$, which refers to the following. If R(C1, C2) and R(C2, C3) then R(C1, C3), each C_i is a class. In the same way, we apply the transitive on the changed abstract properties as follows. Let AsmbList1 and AsmbList2 be two lists of assemblies. In addition, let C2 be a class contained in an assembly of AsmbList1 and let C3 be a class contained in an assembly of AsmbList2. Suppose that we have: R(C1, AsmbList1) and R(C2, AsmbList2); then, we can consider R(C1, AsmbList2), since R(C1, AsmbList1) implies R(C1, C2) (if C2 is a component of the list AsmbList1, then all the properties applied to this list are also valid for its components). Hence, an

abstract property R can be transitive with regard to an AsmbList specified as its range, if its domain is compatible with this AsmbList (i.e. "domain of R" $\subseteq$ AsmbList).

We define the semantics below for the inverse:

Let us consider each C_i as a class, and each R_i as an abstract property in the following. In addition, we note an assembly list as $AsmbList_i$.

$R_1 = inverseOf(R_0)$ implies that $R_1 = R_0^-$. Thus, according to the semantics defined over the assembly list, we have two cases that merge:

1. $R_0 = C_0 \times AsmbList_0$ and $R_1 = C_1 \times AsmbList_1 \rightarrow {C^I}_1 \subseteq {AsmbList^I}_0$ and $\forall {C^I}_i \in {AsmbList^I}_1, {C^I}_i \subseteq {C^I}_0$.
2. $R_0 = C_0 \times AsmbList_0$ and $R_1 = C_1 \times C_2 \rightarrow {C^I}_1 \subseteq {AsmbList^I}_0$ and ${C^I}_2 \subseteq {C^I}_0$

Let, ${C^I}_i \subseteq AsmbList_j \equiv \{\exists\ {Assembly^I}_k, \exists {C^I}_l / Assembly_k \in AsmbList_j$ and $C_l \in Assembly_k$ and ${C^I}_l = {C^I}_i\}$.

The corresponding symmetric semantics is $R \equiv R^-$. Let set_1 be the domain and $AsmbList_1$ be the range of this abstract property. Hence, the interpretation of C_1 domain of R, with $C_1 \subseteq set_1$, is ${C^I}_1 \times {AsmbList^I}_1$. According to our changes on abstract properties, $R \equiv R^- \rightarrow R^I \equiv (R^-)^I$, which implies that: $\forall C_i \subseteq set_1 \rightarrow {C^I}_i \subseteq {AsmbList^I}_1$ and $\forall C_j \in AsmbList_1 \rightarrow {C^I}_j \subseteq {set^I}_1$. Indeed, for R to be symmetric, all its domains and its ranges must be equivalent.

An abstract property cannot be defined as functional using the AssemblyRange. In fact, since the range is defined as a list of assemblies, an individual may have as the range for its ObjectProperty, an assembly of individuals that change from one instantiation to another according to the desired situation. This definition is contradictory with that of the basic OWL functional property, which states that [5]: A functional property is a property that can have only one (unique) value y for each instance x, i.e. there cannot be two distinct values y1 and y2 such that the pairs (x, y1) and (x, y2) are both instances of this property.

We presume the following for the equivalence property. Let R_1 and R_2 be abstract properties. R_1 equivalent to $R_2 (R_1 \equiv R_2)$ means that ${R_1}^I \subseteq {R_2}^I$ and ${R_2}^I \subseteq {R_1}^I ({R_1}^I = {R_2}^I)$.

${R_i}^I \subseteq {R_j}^I \rightarrow {C^I}_i \subseteq {C^I}_j$ and ${AsmbList^I}_i \subseteq {AsmbList^I}_j$ (C_i is the domain of R_i and $AsmbList_i$ is its range). In the same way, we presume the following for the disjointness. $R_1 \not\equiv R_2$ means that ${R_1}^I \cap {R_2}^I = \emptyset$, which implies: ${C^I}_1 \cap {C^I}_2 = \emptyset$ and ${AsmbList^I}_1 \cap {AsmbList^I}_2 = \emptyset$. We define ${AsmbList^I}_i \cap {AsmbList^I}_j = \{Asmb/Asmb \in AsmbList_i$ and $Asmb \in AsmbList_j\}$. So, ${AsmbList^I}_1 \cap {AsmbList^I}_2 = \emptyset$ implies that $\forall Asmb_i \in AsmbList_1$ then $Asmb_i \notin AsmbList_2$ and $\forall Asmb_j \in AsmbList_2$ then $Asmb_j \notin AsmbList_1$.

4.3 Discussion

Enabling the specification of an assemblies list as a range of the ObjectProperty enhances the expressivity of OWL. The designer is given the possibility of specifying

that an individual is linked, through the correspondent relation, exclusively to an assembly of concepts or to another. In such a conceptual design, knowledge retrieval, for the purpose of reuse, is performed on a complete and well structured knowledge design. It relies on the designed assemblies for selecting the adequate knowledge to be reused depending on the current use case. We showed that an abstract property with an AssemblyRange supports inclusion (hierarchy), transitivity, inversion, symmetry, equivalence, and disjointness; but, cannot be functional or inverse functional, since the definition of functional is not comparable with the definition of an assemblies list. These properties could be used in reasoning and inference processes, so the functional property is not useful for these processes with regards to AssemblyRange.

5 Case Representation in the CBR System

In CBR systems, a case describes an experience in a given field. We develop, in what follows, a new approach for its representation within the proposed ontological schema as well as new criterions to index cases for an efficient a posteriori search.

5.1 Case Representation in the Proposed Ontology

Ontology enables knowledge representation into two levels of abstraction. We project these levels onto the CBR system for separating the general domain of interest from its specific cases. The first level describes general knowledge about the considered domain of design, the cooling systems in our case study, whilst the second level represents the different design experiences in form of cases. Figure 5 shows the two ontological levels of abstraction that we integrate in the CBR system.

Concepts, relationships and constraints, well-known and approved by experts of the domain, form the first level. They are reported onto the ontology by means of their corresponding constructs. We use the class construct to model the concepts of the domain under study, for instance the "*Mix*" and "*Compressor*" components of the cooling systems. The ObjectProperty serves to represent the relationships among these concepts and the DatatypeProperty enables the description of the different attributes of concepts. We also use the OWL-provided constraint elements, such as cardinalities and restrictions, to model the general constraints of the domain. Knowledge at this level is always valid. It is, hence, the first reference at the moment of reuse.

The second level is an instantiation of the first-level knowledge. It includes specific knowledge about each single experience in the considered domain. The core of the CBR system is located in this second level as cases are there. We discuss, in what follows, the proposed representation for each single design experience, namely a case.

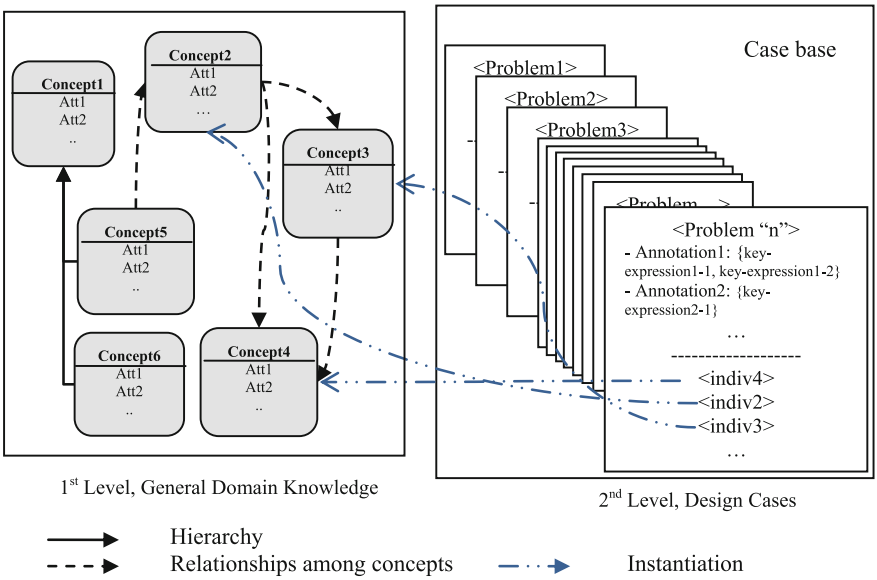


Fig. 5 Knowledge representation model

A case in the CBR system describes one design experience. It is composed of two parts: the problem description and its design solution. We propose to represent the problem description for a given experience by means of the annotation property. This latter is an OWL construct that enables the modeling of metadata on the ontology, such as comments and versions. Thus, the problem is described by means of key expressions, each key expression describing a key aspect of the problem. In a given case, a key expression is modeled by an annotation. The set of all annotations, for a given case, refers to the situation (problem) treated in the encountered experience.

As to the solution part, it is represented by the set of individuals composing the design. An individual is an instantiation of a given concept from the first abstraction level in the ontology. Accordingly, all the specific knowledge about design experiences is represented through individuals with relationships defined among them and with pairs of attribute-values for each individual, etc.

For example, the solution part of the refrigeration system case includes the individuals: “cond1”, “val”, “cold” and “term”, which are respectively instances of the concepts “Condenser”, “Valve”, “ColdSpace”, and “Thermostat”. The specific characteristics of the “cond1” individual, for instance, are represented by means of its attributes. Its interactions with the other individuals are defined through its instantiated relationships. The problem could be described with the annotations “refrigeration system, two-stage refrigerator, cooling system, absorbs heat, removes heat, rejects heat”, etc.

An OWL file describing the case is added to the case base for every new experience. This file includes a set of annotations, representing the problem description,

and a set of individuals, representing the design solution. Construction of the OWL ontology using the Protégé editor, results in the production of only one file to store all knowledge, both general and specific. All individuals are also stored at the same level without any distinction between the groups of individuals referring to the same case. This current organization of the case base (the ontology) does not satisfy the CBR system requirements. Search according to the CBR approach requires a well delimited scope for each case. We propose to store each case in a separate OWL file for this purpose. As a result, the case base is composed of a file describing the general domain knowledge, and multiple files expressing designs—each of which is relative to a case. This is performed by means of a new plug-in added to the Protégé editor. The plug-in allows for the addition of the current problem description using the existing annotation editor; and, it also enables the specification of the set of individuals composing the design solution. Cases are stored in different files and the user can visualize the set of cases present in the base through the added plug-in. A view of this plug-in is depicted in Fig. 13.

5.2 Case Indexing for Reuse

Case indexing serves as the basis for the CBR structure. In the retrieval phase, the user problem description is matched with the problem part of the different stored cases. The matching is based on the indexing approach that we propose herein. At this stage, we introduce two major contributions. First, instead of indexing cases only with their problem description part, we also index them with information about their solution part. This new indexing approach proves to be useful when the user already knows some information about his/her desired solution. Second, we define new indexing criteria based on the AssemblyRange representative element.

5.2.1 Indexing Using the Problem Description

The problem part in each case, as explained before, is described as key expressions represented by annotations. In order to accelerate search, the whole of the key expressions of the base are grouped by equivalent key expressions. For example, the two key expressions “rejects heat” and “removes heat” are included in the same set. Each constructed set serves to index the cases, which include a key expression of the set in their problem part (Fig. 6). When a new case is added to the base, each key expression composing its problem is added to the set.

Similarity between the key expressions is obtained through the use of similar keywords composing these expressions. Identification of the synonyms can be assisted by experts in the domain and supported by the WordNet [33] ontology. The latter organizes words into synonym sets, each of which represents one underlying lexical concept. In order to enable users to write their key expressions without any restriction in terms of the used keywords, a preprocessing phase, prior to the matching process,

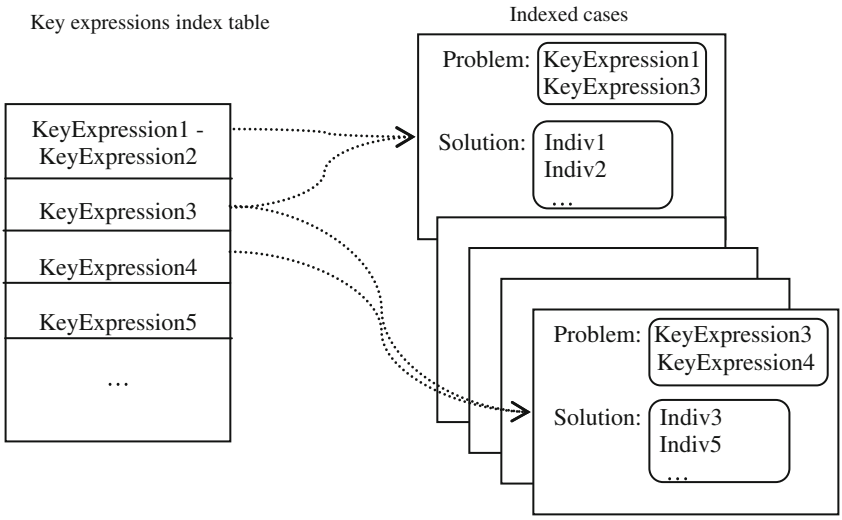


Fig. 6 Indexing with key expressions

is executed. In this phase, the synonyms of the keywords specified in the new key expressions are first identified. Next, the matching process is applied among all of the found synonyms and those of the key expressions of the index table.

5.2.2 Indexing Using the Solution Information

Indexing based on the solution information aims at improving the accuracy of the search process and better organizing the case base. It may be performed according to three different criterions. The first indexing criterion refers to the structural composition of the solution. The second and the third criterions are enabled thanks to the new representative element, AssemblyRange. These criteria concern the contextualization concepts as well as their encapsulations. We describe, in what follows, each of the proposed indexing criterions.

- *Indexing with components*
  Elements composing a solution may be reused in multiple experiences when there exists similarity between the cases. Thus, for each component, present in the ontology, we first identify the cases that make use of it. Next, an index table associates to each component with the list of cases containing it (Fig. 7). According to this criterion, search will only provide the cases including the searched components.
- *Indexing with contexts*:
  For each component existing in the ontology, we first identify its possible contexts. Next, for each context, we list the appropriate cases that make use of this context for the specified component. A two-level index table is created, where the first

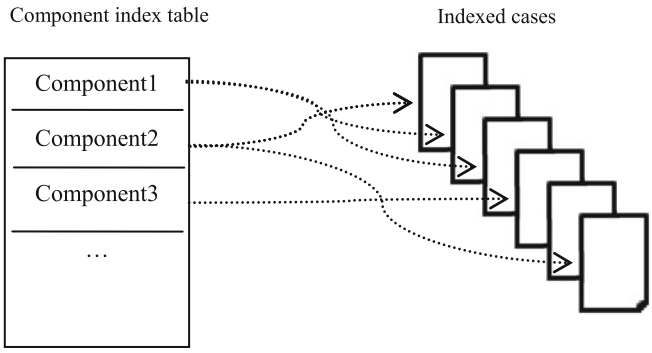


Fig. 7 Indexing with components

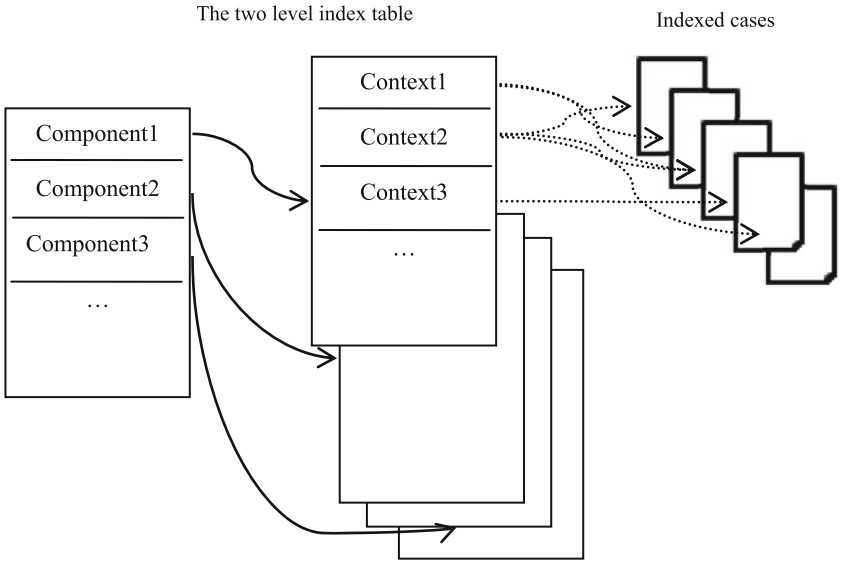


Fig. 8 Indexing with context

level indicates the contexts of a given component and the second points to the cases corresponding to each context at the first level (Fig. 8).

The context can be used in the search process where knowledge is retrieved based on its current context. This will occur when it is similar to what is annotated in one or more cases. Searching on this basis will assist users in realizing their designs by correcting and completing its initial description and even proposing the retrieved knowledge at design time.

As an example, we will consider the cooling systems domain. By including the context of "Condenser" component, the CBR system can retrieve and propose the "Condenser" component to be reused in the current design, which helps in

completing the user design. This is useful when users specify incomplete and uncertain designs as input to the CBR system in addition to the problem descriptions. The design is then completed with the missing components using all of the relevant information provided by the user.

- *Indexing with encapsulation*:
  Encapsulation is also used to index the cases through the creation of a two-level indexing table. The first level indicates the possible encapsulations for each component in the ontology; whereas, the second points to the corresponding cases for each encapsulation specified in the first level.
  Here, the retrieval process is performed based on the encapsulated components. For instance, specifying some aggregates of the "Compressor" component results in the proposition of this component to the user.

6 Application and Results

We use the OWL language to describe our ontology. One of the most popular ontology editors is Protégé. Protégé is open source, free, and supports the OWL description language.

The assembly list is implemented as a plug-in into Protégé editor. This plug-in allows users to create their own lists of assemblies when specifying contexts or encapsulations for a given concept. In addition, users can chose the adequate assembly for the current situation from previously edited lists. Figure 9 presents a view of the interface for this plug-in.

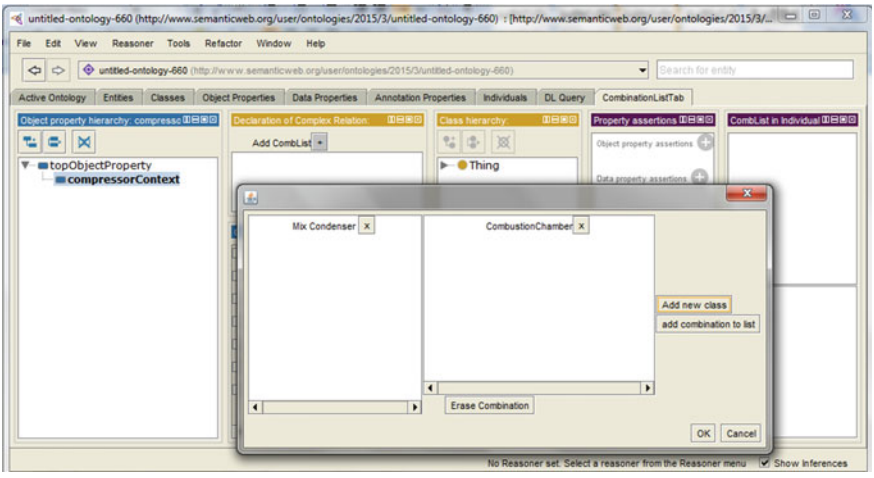


Fig. 9 Plugin AsmbList in Protégé

```
    <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/user/ontologies/2015/3/untitled-
ontology-659#compressorContext">
<rdf:range>
<owl:AssemblyRange>
<owl:oneOf>
<rdf:List>
<rdf:first rdf:parseType="Collection">
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-659#Mix>
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-
659#Condenser>
</rdf:first>
<rdf:rest>
<rdf:List>
<rdf:first rdf:parseType="Collection">
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-
659#Condenser>
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-
659#Thermostat>
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-
659#Evaporator>
</rdf:first>
<rdf:rest>
<rdf:List>
<rdf:first rdf:parseType="Collection">
<http://www.semanticweb.org/user/ontologies/2015/3/untitled-ontology-
659#CombustionChamber>
</rdf:first>
<rdf:rest rdf:resource="&rdf;nil"/>
</rdf:List>
</rdf:rest>
</rdf:List>
</rdf:rest>
</rdf:List>
</owl:oneOf>
</owl:AssemblyRange>
</rdf:range>
        <rdfs:domain
rdf:resource="http://www.semanticweb.org/user/ontologies/2015/3/untitled-
ontology-659#Compressor"/>
    </owl:ObjectProperty>
```

Fig. 10 List of assemblies representing possible contexts for the compressor component

Figure 10 shows the edited ontology in OWL using the AssemblyRange plug-in. A context relation, *compressorContext* is created in the depiction. The domain of this relation is the class *Compressor* and its range refers to a list of assemblies, which describe all possible contexts for this component. As defined by the diagram of Fig. 12 (refer also to the IBD of Fig. 1), the possible contexts for the *Compressor* are: (*Mix, Condenser*), (*Condenser, Thermostat, Evaporator*), and (*CombustionChamber*).

Figure 11 shows the individual syntax edited through the plug-in. In this example, we represented the *comp2* case, where the context for the *Compressor* component is set as (*mix, cond2*), which refers to the first assembly. Note that the individual *mix* is of class type *Mix* and *cond2* is of class type *Condenser*.

The resulting ontology, supporting the AssemblyRange construct, can be built as shown in Fig. 12.

We see that the use of an assembly list as the range for the relation *Compressor-Context*, facilitates the modeling of the three possible contexts for the *Compressor*

```
    <owl:NamedIndividual
rdf:about="http://www.semanticweb.org/user/ontologies/2015/3/untitle
d-ontology-659#comp2">
<compressorContext rdf:resource="&/untitled-ontology-65;mix"/>
<compressorContext rdf:resource="&/untitled-ontology-65;cond2"/>
        <rdf:type
rdf:resource="http://www.semanticweb.org/user/ontologies/2015/3/unti
tled-ontology-659#Compressor"/>
    </owl:NamedIndividual>
```

Fig. 11 The chosen assembly for the case comp2

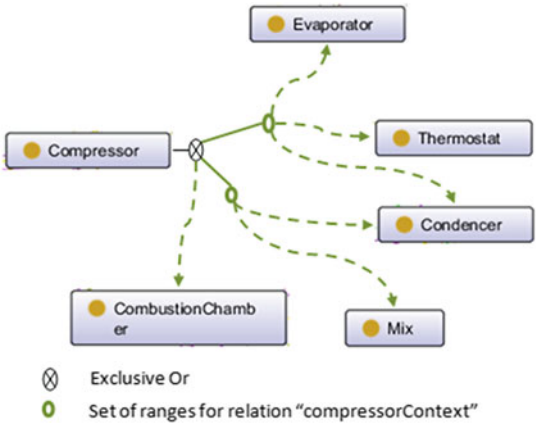


Fig. 12 Diagram of the created ontology using AssemblyRange

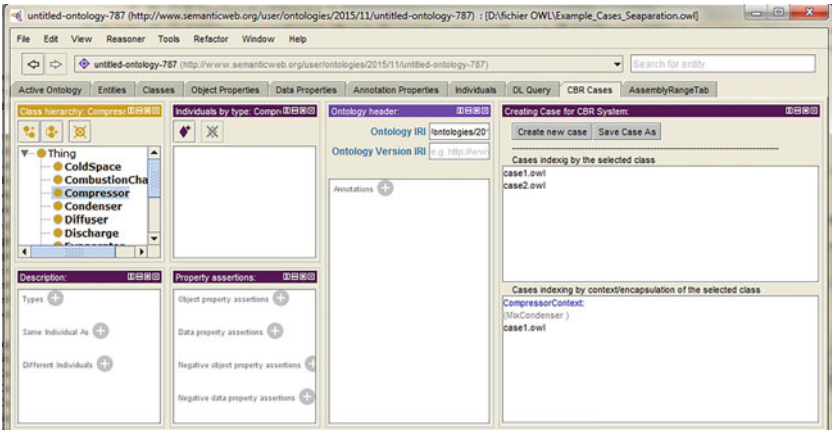


Fig. 13 Cases' indexing with different search criterions

component. The first context points to the components: *Condenser, Thermostat*, and *Evaporator*. The second context indexes the *Condenser* and *Mix* components. Finally, the third one points to the *CombustionChamber* component.

We can access and retrieve the cases that make use of these notions using the contexts and encapsulations for a given component. The indexing performed on cases, which is depicted in Fig. 13, enhances the search process with more criteria for retrieving similar cases. For instance, the figure shows the cases that include the *Compressor* component in the top level of the window and also shows cases that include the elements composing the first context of the *Compressor* component at the bottom of the interface.

7 Conclusion

Models need support, in a suitable form, along with knowledge embedded in designs to promote reuse. Ontologies comprise an appropriate representation model by way of the OWL language. However, this language is lacking in some capabilities for conceptual representation. We proposed, in this work, to enhance OWL's expressivity by extending the syntax and semantics associated with the ObjectProperty range. This extension is performed by associating a list of assemblies as a range for the ObjectProperty. Thus, to represent different contexts and encapsulations for a given concept, we have only to create an ObjectProperty for this concept, whose range expresses different contexts or encapsulation through a list of assemblies.

We have formally defined the syntax and semantics of the new construct and implemented the proposed changes in the plug-in through the Protégé editor. In order to benefit from this new reusable construct, case indexing needs to be enhanced with the context and encapsulation criteria. This allows for decreasing the complexity of search and provides for increased efficiency.

References

1. Gu, C.C., Hu, J., Peng, Y.H., Li, S.: FCBS model for functional knowledge representation in conceptual design. J. Eng. Des. **23**(8), 577–596 (2012)
2. Li, S., Hu, J., Peng, Y.H.: Representation of functional micro-knowledge cell (FMKC) for conceptual design. Eng. Appl. Artif. Intell. **23**(4), 569–585 (2009)
3. Ramirez, C., Valdes, B.: A general knowledge representation model of concepts. In: Ramirez , C., (ed.) Advances in Knowledge Representation (2012)
4. Gómez-Pérez, A.V., Benjamins, R.: Applications of ontologies and problem-solving methods. AI Mag. **20**(1), 119–122 (1999)
5. Bechhofer, S., Harmelen, F.V., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL web ontology language reference W3C recommendation, W3C, 10 February 2004
6. Connolly, D., Harmelen, F.V., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: DAML+OIL, reference description, W3C (2001)

7. Beckett, D., McBride, B.: RDF/XML syntax specification (Revised), W3C recommendation 10 February 2004
8. Brickley, D., Guha, R.V.: RDF schema 1.1, W3C recommendation 25 February 2014
9. Bouabana-Tebibel, T., Rubin, S.H., Chebba, A.: Context and encapsulation modeling for knowledge reuse. In: IRI 2015, pp. 98–105 (2015)
10. Richter, M.M., Weber, R.O.: Case-Based Reasoning, Springer (2013). ISBN: 978-3-642-40166-4 (Print) 978-3-642-40167-1
11. Uhlmann, J., Schulze, E.-E.: Evaluation of design knowledge: empirical studies and application of the results in product design education. In: ConnectED International Conference on Design Education 2007. Sydney (2007)
12. Visser, W.: The cognitive artifacts of designing. Lawerence Erlbaum Associates, Mahwah, New Jersey (2006)
13. Lawson, B.: What Designers Know. Elsevier Architectural Press, Oxford (2004)
14. Woelfel, C., Prescher, C.: A definition of design knowledge and its application to two empirical studies, In: Swiss Design Network (eds.) Focused—Current Design Research Projects and Methods, pp. 285–300. Swiss Design Network
15. Mohamed, R., Watada, J.: Evidence theory based knowledge representation. ACM (2011)
16. Quillian, M.R.: Semantic memory. In: Minsky, M. (ed.) Semantic Information Processing, pp. 227–270. MIT Press, Cambridge (1968)
17. Minsky, M.: A Framework for Representing Knowledge. Massachusetts Institute of Technology, Cambridge (1974)
18. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading (1984)
19. Eksioglu, K.M., Lachiver, G.: A cognitive model for context dependent fuzzy knowledge. In: 18th International Conference of the North American Fuzzy Information Processing Society, NAFIPS (1999)
20. W3C OWL Working Group: OWL 2 web ontology language: document overview (Second Edition). December 2012
21. Zainol, Z., Nakat, K.: Generic context ontology modelling: a review and framework. In: IEEE 2nd International Conference on Computer Technology and Development (ICCTD 2010), pp. 126–130. Piscataway, NJ (2010)
22. Coma, C., Cuppens-Boulahia, N., Cuppens, F., Cavalli, A.N.: Context ontology for secure interoperability. In: Third International Conference on Availability, Reliability and Security, pp. 821–827. IEEE Computer Security (2008)
23. Strimpakou, M.A., Roussaki, L.G., Anagnostou, M.E.: A context ontology for pervasive service provision. In: Proceedings of the 20th International Conference on Advanced Information Networking and Applications, pp. 775–779 (2006)
24. Bouquet, P., Giunchiglia, F., Harmelen, F.V., Serafini, L., Stuckenschmidt, H.: Contextualizing ontologies. Web Semant. Sci. Serv. Agents World Wide Web, **1**(4), 325–343 (2004)
25. Sarkheyli, A., Söffker, D.: Case indexing in Case-Based Reasoning by applying Situation Operator Model as knowledge representation model. IFAC-PapersOnLine, **48**(1), 81–86 (2015)
26. Mittal, A., Sharma, K.K., Dalal, S.: Applying clustering algorithm in case retrieval phase of the case-based reasoning. Int. J. Res. Aspects Eng. Manag. **1**(2), 14–16 (2014)
27. Naderpajouh, N., Afshar, A.: A case-based reasoning approach to application of value engineering methodology in the construction industry. Constr. Manag. Econ. **26**(4), 363–372 (2008)
28. Zarandi, F.M., Razaee, Z.S., Karbasian, M.: A fuzzy case based reasoning approach to value engineering. Expert Syst. Appl. **38**(8), 9334–9339 (2011)
29. OMG Systems Modeling Language: SysML V1.4 beta specification, March 2014
30. Protégé ontology editor developed by Stanford Medical Informatics, Stanford University School of Medicine. http://protege.stanford.edu/ Protégé 4.3, released (2013)
31. Horrocks, I., Patel-Schneider, P.F., Harmelen, F.V.: From SHIQ and RDF to OWL: the making of a web ontology language. J. Web Semant. **1**(1), 7–26 (2003)
32. Patel-Schneider, P.F., Horrocks, I.: OWL web ontology language semantics and abstract syntax section 2, Abstract Syntax W3C (2004)
33. Fellbaum, C.: WordNet: an electronic lexical database. MIT Press (1998)

Intelligent Decision Making for Customer Dynamics Management Based on Rule Mining and Contrast Set Mining

A Segmentation Analysis Perspective

Elham Akhond Zadeh Noughabi, Behrouz H. Far and Amir Albadvi

Abstract In real world situations, customer needs and preferences are changing over time and induce segment instability. The aim of this paper is to explore the patterns of customer segments' structural changes. This study examines how businesses can gain better insight and knowledge through using data mining techniques to support intelligent decision making in customer dynamics management. Up to now, no attempt was done to describe and explain segments' structural changes or to investigate the impact of customer dynamics on these changes. In this paper, a general method is presented based on rule mining and contrast set mining to describe and explain this issue. This method provides explanatory and predictive analytics to enlarge the opportunities for intelligent decision making in this area. The method is implemented on two different data sets for more generalizability. The results show that the method is capable in this domain. Based on the findings, a new concept is developed in the domain of customer dynamics as "structure breakers" that represents a group of customers whose dynamic behavior causes structural changes. The results provide knowledge through some if-then rules which would improve the decision making ability of marketing managers.

Keywords Customer segmentation · Structural changes · Data mining · Sequential rule mining · Contrast set mining · Intelligent decision making

E. Akhond Zadeh Noughabi (✉) · B.H. Far
Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada
e-mail: Elham.akhondzadehnou@ucalgary.ca

B.H. Far
e-mail: far@ucalgary.ca

A. Albadvi
Department of Industrial Engineering, Tarbiat Modares University, Tehran, Iran
e-mail: albadvi@modares.ac.ir

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_6

1 Introduction

With the advent and growth of information technology, intelligent decision making has reached greater significance and has proved to have a particularly large potential in many administrative, social, economic, business and medical issues. Intelligent Decision support systems (IDSS) are a specific class of computerized information systems that support business and organizational decision making activities. Intelligent expert systems, rule-based systems and business intelligence tools are some examples. These systems are developed with the goal of guiding users through some of the decision making phases and tasks [1–4].

On the other hand, data mining technology extends the possibilities for intelligent decision support by discovering and extracting hidden patterns and knowledge in data through an inductive approach. Data mining techniques are capable in analyzing a large amount of data to extract useful knowledge, and introduced as an important component in designing intelligent decision support systems (IDSS) [3, 5]. In this regard, this paper examines how data mining technology can support intelligent decision making and can enlarge the opportunities for intelligent decision support systems in customer relationship management (CRM).

The competitive business environment is continuously changing over time and forces companies to analyze and understand customer needs, preferences and behavior [6–8]. Organizations need to have a deeper understanding and more complete view of customer behavior in order to gain a competitive advantage. One of the most important issues that should be considered while analyzing customer behavior is "customer dynamics" [9–14]. Actually, in today's competitive business environment, customer behavior is often complex and uncertain. In fact, because of the influence of psychosocial and environmental factors, customer behavior and preferences are changing over time. Facing such a dynamic situation, it is necessary to understand the changes of customer behavior and consider this dynamism in different business-related activities to develop effective marketing strategies [15–18].

One of the most important and strategic marketing activities that should be conducted by considering the dynamic nature of customer behavior, is customer segmentation. In fact, in real world situations, customer needs and preferences are changing and induce segment instability [9–12, 14, 16]. Based on [14, 18], customer segments can change in several ways over time. New segments can appear, disappear, merge, move, shrink or grow. The focus of this paper is on the changes in the type and composition of segments. These changes are very important and introduced as segments' structural changes. For example, a segment may disappear over time; conversely, a new group might appear. Two segments can be merged into one group or a segment may split into two or more ones [19].

This paper focuses on analyzing dynamic customer behavior and investigating its impact on segments' structural changes. This study examines how businesses can gain better insight and knowledge through using data mining techniques to support intelligent decision making in customer dynamics management. This paper is a revised and extended version of our paper [20] published in IRI 2015. The main

aim is to explore the patterns of customer segments' structural changes. The main question is how and in which manner segments' structural changes occur. In other words, are there any patterns or trends that can describe the structural changes? We answered to these questions in our paper [20] published in IRI 2015. Our suggestion was to use the sequential rule mining technique to extract such patterns and trends.

These rules can provide us with a good insight about patterns of segments' structural changes. Furthermore, one of the main advantages and benefits of such rules is to support intelligent decision making which is discussed in the current paper. The intelligent agents and systems usually rely on a knowledge base containing a set of rules. Accordingly, the focus of this study is to present a general method to provide knowledge through some if-then rules to support intelligent decision making for customer dynamics analytics and management. This method provides explanatory and predictive analytics to enlarge the opportunities for intelligent decision making in this area. The method is developed based on data mining techniques including rule mining and contrast set mining.

Up to now, there has been no research on describing and explaining the manner of segments' structural changes. The researches have been conducted, extracted only the types of segments over time to see if there is any change or not. No attempt was done to describe and explain the structural changes or to investigate the impact of dynamic customer behavior on segments' structural changes. As the best of our knowledge, our research is the first to develop a general method based on rule mining techniques to approach this problem. A new concept is also developed in this study in the domain of customer dynamics as "structure breakers". We implement our method on two different data sets to validate this concept and the proposed method. This would allow for more generalizability of the new concept and method.

The structure of the paper is as follows: In Sect. 2, the related literature is briefly reviewed. Section 3 explains the proposed method. The analysis of results is presented for two different data sets in Sect. 4. Section 5 discusses the potential usefulness of results in intelligent decision making. Finally, Sect. 6 deals with the conclusion.

2 Literature Review

2.1 Customer Relationship Management

Customer relationship management (CRM) has been identified as an important business concept. All of the existing definitions consider it as a comprehensive process of acquiring and retaining customers integrating with business intelligence concepts in order to maximize the customer value [6, 18, 21].

CRM cycle includes four dimensions: customer identification, customer attraction, customer retention, and customer development. This cycle begins with customer identification that has two elements: "customer segmentation" and "target customer analysis" [10, 14]. Accordingly, customer segmentation has critical importance as the first phase of CRM process [6, 22].

2.2 Customer Segmentation

Customer segmentation is introduced as a strategic marketing and CRM activity [6, 23]. It is defined as the process of dividing customers into distinct and meaningful groups of homogeneous customers. It helps companies to build differentiated and adopted strategies according to each group's characteristics and to identify the most profitable group of customers [6, 24].

One important issue in customer segmentation is the criteria and attributes on which segmentation is performed. The RFM (Recency, Frequency and Monetary) model is a common and well-known method for customer segmentation. RFM values are defined as follows:

- R stands for "Recency" and relates to the time of the last purchase. This attribute shows the time interval between the last purchase and the target time of analysis;
- F represents "Frequency", indicating the number of purchases in a particular period; and
- M stands for "Monetary", showing the consumption of money during a certain period

A segment of customers with higher values of "Frequency" and "Monetary" and lower value of "Recency" is considered as the most valuable group [25].

Reference [26] proposed TFM model for customer segmentation in telecommunication industry based on RFM model. Time and frequency respectively, show the "average time" and "frequency" of using application services. Monetary indicates the total amount generated for using different application services. As telecommunication applications users may subscribe to applications every few minutes, the Recency variable is not meaningful in these cases. It has proved that heavy and valuable customers in telecommunication industry are the users who accumulate a greater volume of service time (T), purchase services frequently (F) and amass large billing amounts per month (M) [26].

2.3 Segment Instability

In real world situations, customer needs and preferences are changing over time and induce segment instability [9–12, 14, 16]. Tracking the changes of customer segments is very important for companies to develop effective marketing strategies [27].

Reference [19] can be considered as the major study in this field that addressed comprehensively the issue of dynamic segments in a review work. The basic related theories are conceptually explored and a comprehensive review of literature is performed [18]. It is notable that the focus of this study is on segment instability in business-to-business markets. The authors defined segment instability as below:

"Segment instability refers to a state of change in customers' needs and what they value within identified market segments, as well as changes in segment membership, as triggered by internal to the customer and external to the customer change drivers, and reflected by changes in segment contents and segment structure".

One important issue emphasized in this paper is investigating the relation between customer value change and segment instability. Developing tools that are capable of forecasting the direction of segment changes integrating the theories of customer value was recommended by these researchers.

Reference [10] also discussed the changing customer needs from the view point of product development in a review research. They discussed different related subjects including segment instability that was extensively documented. Some other researchers, who have empirically investigated segments' changes, are [9, 14, 27]. They have used the frequent item sets and association rule mining to mine the changes of segments [18].

Investigating the published papers show that most of them have considered only the content changes of segments. However, in real world situations, segments can change in several ways: new groups can appear, disappear, merge, move, shrink or grow over time [14, 18]. Accordingly, one of the challenging research areas can be modeling the complex nature of structural changes of segments. The researches have been conducted in this domain, extracted only the types of segments over time to see if there is any change or not. No attempt was done to describe and explain the structural changes.

2.4 Intelligent Decision Support Systems

Intelligent Decision support systems (IDSS) are a specific class of computerized information systems that support business and organizational decision making activities. These systems are developed with the goal of guiding users in some of the decision making phases and tasks. Intelligent expert systems, rule-based systems and business intelligence tools are some examples [1–4]. The use of intelligent decision support systems allows to decrease the time for decision making and to improve the quality and efficiency of decisions [28].

An IDSS has a data base, knowledge base, and model base. This paper is related to the knowledge base as a key component in these systems. The knowledge base holds problem knowledge, such as guidance for selecting decision alternatives or advice in interpreting possible outcomes. A large number of applications of IDSSs are based on the knowledge bases which have capabilities to maintain information and knowledge in the form of rules (i.e. if-then rules, decision trees etc.). If information about different scenarios is stored in these systems, it can provide a basis for taking any suitable action in many unexpected circumstances [1, 28, 29].

2.5 Clustering

Clustering is an unsupervised data mining method that divides data into groups such that the objects in each cluster are very homogenous but dissimilar to the objects in other clusters [30]. We use the K-means algorithm in this study which is the most common clustering method for customer segmentation.

This algorithm firstly selects K objects randomly as the initial centers of clusters. Next, the remaining objects are assigned to the closest clusters based on the distance between the object and the center of cluster. Then, the update and assign steps are run repeatedly until the criterion function converges [30].

We use the Davies–Bouldin index for clustering validation in this study. It is one of the most popular indexes and considers both the cohesion and separation concepts [19]. The value of K that minimizes this index is selected as the optimal number of clusters. This index is calculated as the following:

$$DB_{nc} = \frac{1}{n_c} \sum_{i=1}^{n_c} R_i \tag{1}$$

$$R_i = \max_{i=1,\ldots,n_n, i \neq j} R_{ij}, \tag{2}$$

$$\mathrm{R_{ij}} = (S_i + S_j)/d \tag{3}$$

R_{ij} is a similarity measure between cluster C_i and C_j. S_i and d_{ij} are two measures for the dispersion of a cluster and the dissimilarity between two clusters, respectively [31].

2.6 Sequential Rule Mining

Sequential pattern mining which extracts the frequent subsequences in a set of sequences was first proposed by Agrawal and Srikant (1995) [32–34]. This technique is not able to make predictions. To overcome this shortcoming, the sequential rule mining technique is developed that can address the prediction issues [33].

A sequential rule is an expression of the form X→Y with two measures including support and confidence. This rule indicates that if event X occurs, event Y is likely to occur following the occurrence of X with high confidence or probability. The support and confidence measures are used to quantify the significance of the rule. The support implies the probability that the rule may occur in the sequence database. The confidence indicates the support of the rule over the support of the antecedent (X). When sequential patterns are extracted, sequential rules can be generated by using a minimum confidence threshold [33, 35].

In this paper, we use the generalized sequential pattern (GSP) algorithm, which is a basic and well-known sequential pattern mining technique. It is the generalized form the algorithm proposed by [32]. The priority and advantage of this

algorithm is considering time constraints and sliding time window. This algorithm adopts an Apriori-like candidate set generation-and-test approach consisting of two main phases: candidate generation and support counting. In the first pass, all frequent single items (1-sequences) are extracted. The candidate 2-sequences are then formed based on these single frequent sequences. After calculating the support of these candidates and selecting the frequent ones, the candidate 3-sequences are generated, accordingly. This process is repeated until no more frequent sequences are found [18, 36–39].

2.7 Contrast Set Mining

A special data mining task which is developed to find differences between contrasting groups is contrast set mining [40]. Comparing groups to find differences between them can be very useful in many applications [39]. Contrast set mining can be performed by using different techniques; for example decision tree induction, rule learning and mining frequent item sets [40]. In this paper, we use distinguishing sequential rules and emerging patterns that are explained in the following two sub-sections.

Distinguishing sequential rules.

There are four types of distinguishing sequential patterns: site-characteristic, site-class-characteristic, class-characteristic and surprising. This paper is in the domain of the third type. In this field, a distinguishing sequential rule is defined as below based on [18]:

"Given two sequence sets A and B, a distinguishing sequential rule contrasting A from B, is a strong rule from A that does not match with the sequences of B with high strength. In other words, the sequences of B do not approve the occurrence of this sequential rule; in fact, this rule would be considered as a weak rule in B. The strength and interestingness of such rules change significantly from one group of customers to another."

These rules are helpful for mining useful contrast knowledge and also for prediction purposes [18, 34]. Reference [18] presented a framework for finding minimal distinguishing sequential rules as below:

"Given two groups of sequences pos (positive) and neg (negative), two support thresholds α_1 and α_2 ($\alpha_1 < \alpha_2$), and three confidence thresholds β_1, β_2 and β_3($\beta_1 < \beta_2 < \beta_3$), a sequential rule (r) is defined as a minimal distinguishing sequential rule if and only if the following conditions are satisfied:

- **High strength condition**: $r \in L$ where L is the set of top-k sequential rules of positive group that are extracted based on the following definition proposed by [41]. The min confidence is set equal to β_2.
  Reference [41] defined mining top-k sequential rules as "discovering a set L containing k rules such that for each rule $r_m \in L$, $conf(r_m) \geq$ min conf and there exists

no rule $r_n \notin L$ such that $sup(r_n) > sup(r_m)$ and $conf(r_n) \geq$ min conf. In fact, k rules are discovered that have the highest support such that their confidence is higher than the confidence threshold.

- **Low strength condition**: $conf_{neg}(r) < \beta_1$ or $supp_{neg}(r) < \alpha_2$
  Each of the above two options indicates low strength condition. In each case, we face a weak rule with low strength and poor reliability. In the first one, we consider only the confidence threshold, because the rules with low confidence are poor, even if their support is high. The rules with $supp_{neg}(r) < \alpha_2$ are considered as weak rules with one exception mentioned in Note 1.
- **Non-redundancy condition**: r is a Non-redundant rule.
  Assume r and r′ are two rules with the same support and confidence values that satisfy the above both conditions. Sequential rule r: $a \rightarrow b$ is redundant with respect to another rule r′: $a' \rightarrow b'$ if and only if it can be inferred by rule r′. In other words, rule r is redundant if and only if conf (r) = conf (r′) and supp(r) = supp(r′) and $a' \subseteq a$ and $b \subseteq b'$.
  The redundancy condition was derived from [41, 42] that discussed the non-redundant sequential rules.

Note1: The mentioned exception in low strength condition is the rule satisfying both $\alpha_1 \leq supp_{neg}(r) < \alpha_2$ and $Conf_{neg}(r) \geq \beta_3$ conditions, because the rules with low support and very high confidence are often interesting and provide new insights. Actually, these rules indicate small groups in the negative group that behave the same as the positive group members with a high confidence. Obviously, this affects the strength of distinguishing rules and may reduce the accuracy of decisions and predictions. The minimum threshold (α_1) for $supp_{neg}$ is considered to remove the rare items that may be noise or outliers.
Note 2: Considering that time constraints are essential to find more interesting rules and to efficiently aid decision making [37], we suggest these constraints for mining distinguishing sequential rules".

Emerging patterns.

Emerging patterns are defined as item sets whose support values change significantly from one data set to another. They can be used to find the emerging trends in time stamped databases or to extract useful and interesting contrasts between different classes and groups [9, 43]. Emerging patterns cover a wide range of techniques including decision tree, frequent item sets, association rules, classifiers and etc. The main idea is comparing two sets of patterns from two splits of data [14, 43].

Emerging patterns can be defined based on [43, 44] as below:

Rule r_j^{t+k} is called an emerging pattern with respect to rule r_j^t, if the following two conditions are met:

1. The antecedent and consequent parts of the rules r_j^{t+k} and r_j^t are the same.
2. Supports of two rules are significantly different.

3 The Proposed Method

The aim of this study is to present a general method to explore the patterns of customer segments' structural changes and to provide knowledge to support intelligent decision making for customer dynamics management in segmentation analysis.

The proposed method is developed based on clustering, distinguishing sequential rules and emerging patterns. This method includes 6 main steps. These steps are explained in details below:

Step 0: **Data collection and preprocessing**.

This step includes data collection and preprocessing which is discussed in details later.

Step 1: **Detecting customer segments in each period and identification of structural changes**.

This step includes three sub-steps as follows:

1. Clustering the customers based on TFM/RFM model in each period
   Firstly, customer segments are identified in each period by using the K-means algorithm. It is notable that the initial centers influence the results of the K-means algorithm. In this regard, the initial centers are selected randomly in this paper which is the common approach to choose the initial centroids. One technique that is commonly used to address the problems of choosing initial centroids by random is to perform multiple runs with a different set of randomly chosen initial centroids and selecting the set of clusters with the minimum SSE or Davies–Bouldin [30]. Accordingly, we performed the algorithm 70 runs with a different set of randomly chosen initial centroids for each K to select the optimum clusters. Normalizing the TFM/RFM attributes is performed by the Min-Max normalization method. The Davies–Bouldin index is used for the evaluation of the clustering results.
2. Analyzing and labeling the clusters
   Analyzing and labeling the clusters are performed based on the model proposed by [30] in the following manner:
   The average of T/R and F variables in each cluster are calculated and compared with their total average of all customers in the corresponding period. If the average of T/R (F) variable in a cluster is greater or less than the overall average T/R (F), the High (H) or Low (L) label is assigned to the corresponding variable, respectively. For the M variable, three labels are considered including High1 (H1), High2 (H2) and Low (L). The thresholds for the monetary categorization were chosen based on the first 20 and 30 % of customers with higher values of monetary [16]. We also assign the label "Inactive" to a customer when he/she does not have any transactions during a period.
3. Identification of structural changes
   The identification of structural changes is performed by comparing the types of segments obtained in different periods.

Step 2: **Extracting the transition sequences**.

In this step, an individual sequence is built for each customer indicating the history of his/her membership to different segments over time. We name this sequence as "transition sequence". In this step, the transition sequences of all customers are extracted.

Step 3: **Categorizing the transition sequences of dynamic customers into two groups**.

In this step, the corresponding transition sequences to dynamic customers are selected, firstly. Dynamic customers are those who do not remain stable or relatively stable in one specific segment over time. Then, these sequences are classified into two groups. The first group includes the transition sequences indicating the identified structural changes. The transition sequences that include at least one state referring to the structural changes are assigned to this group. The remaining sequences are considered as the second group.

Step 4: **Mining the distinguishing sequential rules of each group**.

In this step, the distinguishing sequential rules of each group are extracted. To mine the sequential rules, the sequential patterns are firstly extracted by using the GSP algorithm. The sequential patterns are the ones that satisfy the minimum support threshold and time constraints on the minimum and the maximum gap between two adjacent items. Then, sequential rules are extracted from the set of obtained sequential patterns by using a minimum confidence threshold. To find the distinguishing sequential rules, the framework of [18] is used.

In fact, there may be some identical and similar patterns relating to these two groups which cannot provide any additional and special information about them. In other words, this kind of rules cannot indicate the differences between the customers' behavior of two groups and are not able to classify them. In contrast to these rules, there may be some distinguishing rules that differentiate these two groups from each other. These sequential rules capture what makes one group different from another one and indicate their special characteristics. They can be also used for prediction purposes.

In this step, by analyzing the rules of the first group that relates to identified structural changes, we can understand how these changes occur. In other words, these rules are representing the patterns of structural changes. The distinguishing rules of the second group help us to understand the behavior of the customers of this group and the difference between these two groups better. In fact, we obtain three categories of rule: rules that distinguish group 1 from other customers, rules that differentiate the second group from the first one and identical rules between two groups. These three groups of rules firstly provide the marketing managers with a good insight about the behavior of dynamic customers. Second, these rules can be very helpful for prediction purposes and help the marketing managers to classify a new customer into these two groups and to predict his/her behavior.

Step 5: **Mining emerging patterns in each group**.

In this step, we try to find the contrast characteristics of these two groups by using the definition presented in Sect. 2.7 and mining the frequent item sets. At this stage, different characteristics of customers depend on the type of case and the aim of study can be used. In this paper, we use the demographic features which are available such as age, gender and etc. Based on the results of the previous step, a group of customers is identified as "structure breakers" whose behavior causes structural changes. The other group is named as "non-structure breakers". This group includes dynamic customers who switch between different segments over time but do not cause any structural changes. The aim of this stage is to find the characteristics that differentiate "structure breakers" from "non-structure breakers" or vice versa by mining the frequent item sets of these two groups and finding the contrast sets. If the difference between the support values of a frequent set in two groups is more than α, this frequent set is detected as a contrast set. The parameter α is set by the user. The obtained knowledge of this step can be very helpful for intelligent decision making.

4 Results

Here, we present the implementation of the proposed method and analysis of results. Firstly, we discuss the results of dataset 1 in details. Next, the results analysis of data set 2 is presented more briefly.

4.1 Results Analysis of Data Set 1

Data collection and preprocessing.

Among different businesses, the telecommunication industry and specially the mobile telecommunication are the best examples of a dynamic market; the customer behavior is very dynamic and complex in this industry. In this regard, we implemented our method on a data of a telecommunication service provider, firstly.

The data source has been the database of this company. It includes approximately 150 fields and 400 million records for three years. These fields provide us with detailed information about the usage of different services by customers; we know that each customer used which kinds of services, when and in what amount. The duration of using every service and obtained monetary value are available. The discount amount for using each service is also identified.

The target population was selected randomly composed of 165,800 customers. In this step, the data preprocessing was also performed including handling noise and missing values, deleting duplicate data, and feature selection.

Detecting customer segments in each period and identification of structural changes.

The data includes 18 two-month periods. The length of time window to analyze the users' usage in this company was set to two months. Customer segments were identified in each period by using the TFM model and K-means algorithm. The Davies–Bouldin index was used to find the optimum number of clusters; the K values of 2–12 were tested in each period.

It is notable that finding the best moment to perform a new analysis is very important and challenging. Selecting the time intervals and the length of time window depend on the type of business and the specific case study. Some businesses are very dynamic and changing very rapidly over time while some are not. The experts' opinion and domain knowledge can be also very helpful to find the best moment. As mentioned above, the time window is set to two months in this paper; because this company analyzes the users' usage every two months. This relates to the characteristics of business and the dynamic nature of customers' behavior in this area. Furthermore, these two-month periods are meaningful based on cultural issues and special events in the related country that affect customer changing behavior.

Based on the results, six different groups of customers were obtained during 18 periods including HHH1, HHH2, HLH2, LHH2, HLL and LLL. For example, label LHH2 implies a group of customers with a lower average of T and greater average of F comparing to the total average. The average of M variable for these customers has been between the average of this variable for the first 20 and 30 % of customers. Or segment HLH2 indicates a group of customers with a greater average of T and lower average of F comparing to the total average. The value of M variable for these customers is the same as segment LHH2 which explained above.

The clustering results of the second period are shown as sample in Table 1. Cluster_0, cluster_1 and cluster_2 are described as LLL, HHH2 and HHH1, respectively.

Table 2 shows the structure of segments over 18 periods. Value 1 indicates that the segment appeared in the corresponding period, while a zero value shows that it did not appear.

As shown, segments HHH1, HHH2 and LLL appeared in all periods. Segment LHH2 was present only three times. Segment HLH2 and HLL were created and stabled after the seventh and twelfth periods, respectively. Accordingly, the main structural changes occurred during the 18 periods include creating segments HLH2 and HLL.

Table 1 The centers of clusters (T_2)

ID	Cluster	T_Mean	F_Mean	M_Mean	Number of customers
1	cluster_2	93.90	2715.29	1114845.55	13,270
2	cluster_1	78.12	1053.77	460211.51	45,620
3	cluster_0	70.86	289.28	145947.09	99,088

Table 2 The structure of segments over 18 periods

Period	Segment					
	HHH1	HHH2	LLL	HLH2	LHH2	HLL
T_1	1	1	1	1	1	1
T_2	1	1	1	0	0	0
T_3	1	1	1	0	0	0
T_4	1	1	1	0	0	0
T_5	1	1	1	0	0	0
T_6	1	1	1	0	0	0
T_7	1	1	1	1	1	1
T_8	1	1	1	1	0	0
T_9	1	1	1	1	0	0
T_{10}	1	1	1	1	0	0
T_{11}	1	1	1	1	0	0
T_{12}	1	1	1	1	0	1
T_{13}	1	1	1	1	1	1
T_{14}	1	1	1	1	0	0
T_{15}	1	1	1	1	0	1
T_{16}	1	1	1	1	0	1
T_{17}	1	1	1	1	0	1
T_{18}	1	1	1	1	0	1

It is notable that the structure of segments in periods 1, 7 and 13 are the same and different from their neighbor periods. This is because of the changes in customers' behavior in these three periods relating to a specific holiday. Based on the marketing experts' opinions of this company, most of customers use services for their businesses; so, their behavior changes in these periods. Hereafter, we do not consider these three periods in our analysis; because the corresponding structural changes relate to a special reason.

Extracting the transition sequences.

In this step, the transition sequences of all customers were extracted. Six samples are shown in Fig. 1.

Categorizing the transition sequences of dynamic customers into two groups.

In this case, we defined the "static customer" and "dynamic customer" as below.

A customer is a "static customer" if he/she remained in one segment at least 13 times over 15 periods; otherwise, the customer is dynamic.

Accordingly, 44,462 customers were identified as "dynamic customers". The number of the transition sequences indicating the identified structural changes was equal to 22,037. The remaining sequences composed of 22,425 records were considered as the second group.

LLL→ LLL → Inactive → LLL→ LLL→ LLL→ LLL→ LLL→ LLL → LLL→ LLL→ LLL→ LLL→ LLL→ LLL
HHH2→ HHH1→ HHH2→ HHH2→ HHH2→ HHH2→ HHH2→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL
HHH2→ HHH1→ HHH2→ HHH2→ HHH2→ LHH2→ HHH2→ HHH2→ HLH2→ HLH2→ HLH2→ HLL→ HLL→ HLL→ HLL
HHH2→ HHH2→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL→ LLL
HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1→ HHH1
HHH2→ LLL→ LLL→ HHH2→ LLL→ LLL→ LLL→ LLL→ LLL → HLH2→ LLL → HLH2→ HLH2→ HLL→ HLL

Fig. 1 Samples of transition sequences

Mining the distinguishing sequential rules of each group.

To find the distinguishing rules, the proposed framework in Sect. 3 step 4 was used by the following parameters:

$\alpha_1 = 1\,\%$, $\alpha_2 = 4\,\%$, $\beta_1 = 50\,\%$, $\beta_2 = 70\,\%$ and $\beta_3 = 90\,\%$.

There are 92 rules that distinguish these two groups from each other. 48 rules belong to the first group and 52 rules relate to the second group. 27 rules are identified as "not-distinguishing" rules. Seven samples are shown in Table 3. The last two rows show 2 samples of not-distinguishing rules. In these two cases, the support and confidence values for the both groups are shown.

For example, the second rule implies a group of customers who were in segment HHH2 and then migrated to HLH2. In fact, the frequency variable of these customers decreased over time and led to create a new behavioral pattern with H, L and H2 values for T, F and M variables, respectively. This new behavioral pattern formed segment HLH2. In other words, the changes in the behavior of these customers caused to form a new segment named in HLH2.

In this section, we analyzed all the obtained distinguishing and not-distinguishing rules. Accordingly, two groups of customers are identified as "structure breakers" and "non-structure breakers" as the following:

"**Structure breakers**": There is a group of customers whose behavior and the dynamism of their behavior caused to form segments HLH2 or HLL. We name these customers as "structure breakers".

The analysis of obtained rules indicates two groups of customers whose changing behavior causes to form segment HLH2: The customers with a high relative stability in segment HHH2 and a group of customers with a high relative stability in segment LLL. The first group includes the customers whose behavior also causes creating segment HLL. In fact, when the frequency variable of these customers decreases, the monetary variable will fall after some periods. The second group contains the customers whose average time of using services and monetary values increased over time.

Table 3 Distinguishing and not-distinguishing rules

ID	Group	Antecedent	Consequent	Rule support (%)	Confidence (%)
1	Group 1	HHH2 then HHH2 then HHH2 then HHH2 then HLH2 then HLH2 then HLH2 then HLH2 then HLH2	HLL	15.62	80.95
2	Group 1	HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HLH2	HLH2	9.08	95.81
3	Group 1	HHH1 then HHH1 then HHH1 then HHH2 then HHH2 then HHH2 then HHH2	HHH2	5.48	85.92
4	Group 2	HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then LLL then LLL then LLL	LLL	36.96	71.68
5	Group 2	LLL then LLL then LLL then Inactive then LLL then LLL then Inactive then Inactive then Inactive	Inactive	15.05	90.45
6	Not-distinguishing	HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2 then HHH2	HHH2	31.28	89.14
				39.72	91.27
7	Not-distinguishing	LLL then LLL then LLL then LLL then LLL then LLL then LLL	LLL	16.31	70.04
				22.47	81.23

Distinguishing rules of the structure breakers show that all the customers with a high relative stability in segment HHH2 are not structure breakers. But, the customers who were firstly in segment HHH1 and then migrated to HHH2 are structure breakers with a high confidence. These customers will migrate to segment HLH2 and then HLL over time. The customers, who shift between segments HHH1 and HHH2 with a higher relative stability in HHH2, also follow this behavior. These rules help us to detect the structure breakers more accurately. If a customer behavior is similar to these patterns, he/she will be a structure breaker with high probability. It is notable that every customer who belongs to segment LLL with a high relative stability, may be a structure breaker or not.

"**Non-structure breakers**": The obtained rules from the second group imply another group of dynamic customers who switch between different segments over time but do not cause any structural changes. We name this group as "non-structure breakers". Analyzing the distinguishing sequential rules of these customers indicate three sub-groups: a group of customers who were relatively stable in segment LLL and sometimes migrated to the inactive status; these customers finally churned. The second group includes the customers who switched between segments HHH2 and LLL with a higher relative stability in segment HHH2, but finally migrated to LLL. The third group of customers was in segment HHH2 and then migrated to LLL over time.

Mining emerging patterns in each group.

In this step, we tried to find the contrast characteristics of "structure breakers" and "non-structure breakers" by using the method presented in Sect. 3 step 5. We had access to four attributes of customers including age, gender, job and geographical region. For the age variable, we defined five categories based on the experts' opinions of this company. The job attribute is also classified into three categories: "jobless", "private jobs" and "governmental jobs". The minimum support to find the frequent item sets was considered equal to 5 %. The parameter α was set to 20 %. The extracted contrast frequent sets are shown in Table 4.

Based on the experts' opinions of this company, the first contrast set of "structure breakers" which is shown in the first row of Table 4, implies the students who partially churned to one of competitors because of special promotions that were launched

Table 4 Contrast frequent item sets of "structure breakers" and "non-structure breakers"

ID	Group	Frequent item set	$Support_{structurebreakers}$ (%)	$Support_{non\text{-}structurebreakers}$ (%)
1	"Structure breakers"	18 <= Age < 25 and Job = jobless	54.37	29.41
2	"Structure breakers"	45 <= Age < 60 and Job = "private jobs" and Gender = "male"	8.43	–
3	"Non-structure breakers"	Age >= 60	1.52	23.76

by that company. This frequent item set is probably related to the customers who migrated from HHH2 to HLH2 and HLL. The experts of this company believe that the second frequent item set refers to the men who use the services for their job; they are the customers who migrated from HHH1 to HHH2. It is notable that the difference between the support values of this frequent set is less that 20 %. As the support of this frequent set for "non-structure breakers" is too low, we considered this item set as a contrast one. We run the algorithm by a minimum support of 1 %; accordingly, the support of this frequent item set in "non-structure breakers" group is surely less than 1 %.

The third contrast set implies that 23.76 % of "non-structure breakers" are older than 60 years old. Based on the experts' opinions, these customers use these types of services infrequently, not only the services of this company. In other words, the demand for using these services decreased gradually over time in this group. Based on the obtained rules of "non-structure breakers", these customers migrated from HHH2 to LLL or from LLL to inactive status that admits this idea. The obtained contrast sets can be used to improve marketing decisions; they can be also implemented for prediction purposes to predict the behavior of a customer based on his/her characteristics.

4.2 Results Analysis of Data Set 2

We applied the proposed method on the customer data of a private bank. The related data contains detailed information about customers' transactions during two years including these attributes: customer ID, date, type of product/account, amount, channel and branch. 20000 customers were selected randomly.

We used the RFM model to detect customer segments over 8 three-month periods. Based on the results, four different groups of customers were obtained during 8 periods including HLL, LLL, LHH1 and LHH2. Segments HLL, LLL and LHH1 appeared in all periods. Segment LHH2 was present only two times at periods 6 and 7. Accordingly, the structural changes in this case include creating segments LHH2.

Similar to the previous case which was discussed in the previous sub-section, steps 3 and 4 of the proposed method were implemented on the both categories of transition sequences including 3893 and 5673 records, respectively. It is notable that most of the customers in this case follow a static behavior. In this case, we defined a customer as "static customer", if he/she remained in one segment at least 6 times over 8 periods.

Based on the results, two groups of customers are identified:

"**Structure breakers**": the customers who were in segment LHH1 in the first periods and then migrated to segment LHH2. In fact, the behavior of this group and the changes in the behavior of these customers cause the formation of segment LHH2 in periods 6 and 7.

"**Non-structure breakers**": this group includes the customer who switched between segments LLL and HLL with a higher relative stability in segment HLL and finally migrated to HLL.

In this case, we did not access to the demographics or any other features of the customers to implement step 5.

5 The Potential Usefulness of Results in Intelligent Decision Making

The focus of this study is to present a general method to provide knowledge through some if-then rules to support intelligent decision making for customer dynamics analytics and management. In this section, we discuss how the obtained results and rules can enlarge the opportunities for intelligent decision making. The obtained results provide a good insight about customers' changing behavior and the patterns of structural changes. This would help the marketing managers to improve marketing decisions.

For example, as a result of the telecommunication case study, a group of customers were detected whose changing behavior causes to create segments HLH2 and HLL over time. As the obtained rules imply, customers partially churn gradually, not instantaneously and the movement from segment HHH2 to HLL happens gradually. In fact, firstly, the frequency variable of these customers decreases and then the monetary variable falls after some periods. This implies that changing customer behavior in this manner is a warning signal indicating a fall in the monetary value. By using the obtained rules and the knowledge achieved about the distinguishing and contrast characteristics of this group, the marketing experts can identify the potential structure breakers at the first steps. Here, there is a good opportunity for the marketing managers to avoid the fall in the monetary values of these customers by using appropriate marketing strategies.

As a main result, three categories of rules are obtained by using the proposed method: "rules that distinguish structure breakers from non-structure breakers", "rules that differentiate non-structure breakers from structure breakers" and "identical rules between two groups". These rules can provide the knowledge required to design the knowledge base of an intelligent decision support system. To support intelligent decision making, the intelligent agents and systems must maintain a knowledge base. This knowledge base usually relies on a set of rules. In this regard, these rules can be very effective and provide good capabilities.

These rules and the obtained contrast characteristics can be very helpful for prediction purposes. This can help the marketing managers to classify a new customer into the mentioned two groups. The obtained rules are also helpful to predict his/her behavior. If a target customer's membership history is similar to the conditional part of a rule, then his/her switching behavior is deemed the consequent part of the rule.

Accordingly, an intelligent expert system can be designed based on the obtained rules and the contrast characteristics. This system can support intelligent decision

making and suggest the type of customer and the future behavior based on the trail of his/her previous switches between different segment and his/her profile.

6 Conclusion

This study proposed a general method to investigate the impact of customer dynamics on segments' structural changes and to explore the patterns of these changes for the first time. This method provides explanatory and predictive analytics through some if-then rules to provide knowledge for intelligent decision making in customer dynamics management. The proposed method was successfully implemented on the customer data of a telecommunication service provider and also a private bank.

According to the results, we defined a new concept in the domain of customer dynamics as "structure breakers". Structure breakers are the customers whose changing behavior causes structural changes. Identifying these customers is very important; because they cause segment instability which is a difficult challenge in customer segmentation analysis. Another group of dynamic customers is also identified as "non-structure breakers" who switch between different segments over time but do not cause any changes in the structure of segments. The distinguishing rules and contrast sets of these two groups provide accurate knowledge about the behavior and characteristics of these two types of customers. The findings provide a good insight about customers' dynamic behavior that helps the marketing managers to improve marketing decisions and strategies.

One of the main advantages of the results is to enlarge the opportunities for intelligent decision making for dynamic customer management. Three groups of if-then rules are extracted: "distinguishing rules of structure breakers", "distinguishing rules of non-structure breakers" and "not-distinguishing and identical rules between these two groups". These rules and the emerging patterns can be used for prediction purposes. We can identify the group that a new customer may belong to; we can also predict the behavior of a customer based on these rules. The obtained rules are relatively easy to interpret and use and so strengthen the practicality of results.

Acknowledgments This research is partially supported by Alberta Innovates Technology Futures (AITF) and University of Calgary's Eyes High program.

References

1. Fasihfar, Z., Mehrabifard, M., Rivandi, J., Al-Hosseini, M.S.: Design of a fuzzy expert system as an intelligent assistant for thesis supervisor's educational counseling. Cum. Sci. J. **36**, 1372–1376 (2015)
2. Phillips-Wren, G., Mora, M., Forgionne, G.A., Gupta, J.N.: An integrative evaluation framework for intelligent decision support systems. Eur. J. Oper. Res. **195**, 642–652 (2009)
3. Kaklauskas, A.: Biometric and Intelligent Decision Making Support. Springer International Publishing, Switzerland (2015)

4. Khademolqorani, S., Hamadani, A.Z.: An adjusted decision support system through data mining and multiple criteria decision making. Procedia Soc. Behav. Sci. **73**, 388–395 (2013)
5. Yang, Y., Tan, W., Li, T., Ruan, D.: Consensus clustering based on constrained self-organizing map and improved Cop-Kmeans ensemble in intelligent decision support systems. Knowl. Based Syst. **32**, 101–115 (2012)
6. Ngai, E.W.T., Xiu, L.I., Chau, D.C.K.: Application of data mining techniques in customer relationship management: a literature review and classification. Expert Syst. Appl. **36**, 2592–2602 (2009)
7. Liu, D.R., Shih, Y.Y.: Integrating AHP and data mining for product recommendation based on customer lifetime value. Inf. Manage. **42**, 387–400 (2005)
8. Lee, ShL: Commodity recommendations of retail business based on decision tree induction. Expert Syst. Appl. **37**, 3685–3694 (2010)
9. Chen, M.C., Chiu, A.L., Chang, H.H.: Mining changes in customer behavior in retail marketing. Expert Syst. Appl. **28**, 773–781 (2005)
10. Chong, Y.T., Chen, C.H.: Customer needs as moving targets of product development: a review. Int. J. Adv. Manuf. Technol. **48**, 395–406 (2010)
11. Kim, J.K., Song, H.S., Kim, T.S., Kim, H.K.: Detecting the change of customer behavior based on decision tree analysis. Expert Syst. **22**, 193–205 (2005)
12. Song, H.S., Kim, J.K., Kim, S.: Mining the change of customer behavior in an internet shopping mall. Expert Syst. Appl. **21**, 157–168 (2001)
13. Liu, D.R., Lai, ChH, Lee, W.J.: A hybrid of sequential rules and collaborative filtering for product recommendation. Inf. Sci. **179**, 3505–3519 (2009)
14. Böttcher, M., Spott, M., Nauck, D., Kruse, R.: Mining changing customer segments in dynamic markets. Expert Syst. Appl. **36**, 155–164 (2009)
15. Ha, S.H., Bae, S.M.: Keeping track of customer life cycle to build customer relationship. In: Li, X., Zaïane, O.R., Li, Z.H. (eds.) Advanced Data Mining and Applications, vol. 4093, pp. 372–379. Springer, Heidelberg (2006)
16. Tan, H., Xu, J., Zhao, B.: Research on index system of dynamic customer segmentation: based on the case study of China telecom. In: 9th IEEE International Conference on Information Management and Engineering, pp. 441–445. IEEE Press, Kuala Lumpur (2009)
17. Ha, S.H.: Applying knowledge engineering techniques to customer analysis in the service industry. Adv. Eng. Inf. **21**, 293–301 (2007)
18. Akhondzadeh-Noughabi, E., Albadvi, A.: Mining the dominant patterns of customer shifts between segments by using Top-K and distinguishing sequential rules. Manag. Decis. **53**, 1976–2003 (2015)
19. Blocker, C.P., Flint, D.J.: Customer segments as moving targets: integrating customer value dynamism into segment instability logic. Ind. Mark. Manage. **36**, 810–822 (2007)
20. Akhond Zadeh Noughabi, E., Albadvi, A., Far, B.H.: How can we explore patterns of customer segments' structural changes? a sequential rule mining approach. In: IEEE International Conference on Information Reuse and Integration, pp. 273–280. IEEE Press, San Francisco (2015)
21. Becker, J.U., Greve, G., Albers, S.: The impact of technological and organizational implementation of CRM on customer acquisition, maintenance, and retention. Int. J. Res. Mark. **26**, 207–215 (2009)
22. Parvatiyar, A., Sheth, J.N.: Customer relationship management: emerging practice, process, and discipline. J. Econ. Soc. Res. **3**, 1–34 (2001)
23. Dibb, S.: Market segmentation: strategies for success. Mark. Intell. Plann. **16**, 394–406 (1998)
24. Tsiptsis, K., Chorianopoulos, A.: Data Mining Techniques in CRM: Inside Customer Segmentation. Wiley, UK (2009)
25. Cheng, C.H., Chen, Y.S.: Classifying the segmentation of customer value via RFM model and RS theory. Expert Syst. Appl. **36**, 4176–4184 (2009)
26. Cheng, L.C., Sun, L.M.: Exploring consumer adoption of new services by analyzing the behavior of 3G subscribers: an empirical case study. Electron. Commer. Res. Appl. **11**, 89–100 (2012)

27. Wang, Z., Lei, X.: Study on customer retention under dynamic markets. In: 2th IEEE International Conference on Networks Security Wireless Communications and Trusted Computing, pp. 514–517. IEEE Press, Wuhan (2010)
28. Wriggers, P., Kultsova, M., Kapysh, A., Kultsov, A., Zhukova, I.: Intelligent Decision Support System for River Floodplain Management. In: Kravets, A., Shcherbakov, M., Kultsova, M., Iijima, T. (eds.) Knowledge-Based Software Engineering. CCIS, vol. 466, pp. 195–213. Springer International Publishing (2014)
29. Ur-Rahman, N.: Textual data mining for next generation intelligent decision making in industrial environment: a survey. Eur. Sci. J. **11**, 346–377 (2015)
30. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Education Inc., USA (2006)
31. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. **1**, 224–227 (1979)
32. Agrawal, R., Srikant, R.: Mining sequential patterns. In: 11th IEEE International Conference on Data Engineering, pp. 3–14. IEEE Press, Taipei (1995)
33. Fournier-Viger, P., Faghihi, U., Nkambou, R., Nguifo, E.M.: CMRules: mining sequential rules common to several sequences. Knowl. Syst. **25**, 63–76 (2012)
34. Dong, G., Pei, J.: Sequence data mining. In: Ahmed, A.K., Elmagarmid, K. (eds.) Advances in Database Systems Advances in Database Systems, vol. 33, pp. 1–148. Springer, New York (2007)
35. Tang, H., Liao, S.S., Sun, S.X.: A prediction framework based on contextual data to support mobile personalized marketing. Decis. Support Syst. **56**, 234–246 (2013)
36. Pei, J., Han, J., Wang, W.: Constraint-based sequential pattern mining: the pattern-growth methods. J. Int. Inf. Syst. **28**, 133–160 (2007)
37. Masseglia, F., Poncelet, P., Teisseire, M.: Efficient mining of sequential patterns with time constraints: reducing the combinations. Expert Syst. Appl. **36**, 2677–2690 (2009)
38. Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds.) Extending Database Technology: Advances in Database Technology, vol. 1057, pp. 25–29. Springer, Heidelberg (1996)
39. Deng, K., Zaiane, O.R.: Contrasting sequence groups by emerging sequences. In: Li, X., Zaïane, O.R., Li, Z.H.H. (eds.) Discovery Science, vol. 5808, pp. 377–384. Springer, Heidelberg (2009)
40. Kralj, P., Lavrač, N., Gamberger, D., Krstačić, A.: Contrast set mining for distinguishing between similar diseases. In: Bellazzi, R., Abu-Hanna, A., Hunter, J. (eds.) Artificial Intelligence in Medicine. LNCS, vol. 4594, pp. 109–118. Springer, Heidelberg (2007)
41. Fournier-Viger, P., Tseng, V.S.: TNS: mining Top-k non-redundant sequential rules. In: 28th Annual ACM Symposium on Applied Computing, pp. 164-166. Coimbra (2013)
42. Lo, D., Khoo, S.C., Wong, L.: Non-redundant sequential rules—theory and algorithm. Inf. Syst. **34**, 438–453 (2009)
43. Song, H.S., kyeong Kim, J., Kim, S.H.: Mining the change of customer behavior in an internet shopping mall. Expert Syst. Appl. **21**, 157–168 (2001)
44. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 43–52. ACM, New York (1999)

Is Data Sampling Required When Using Random Forest for Classification on Imbalanced Bioinformatics Data?

David J. Dittman, Taghi M. Khoshgoftaar and Amri Napolitano

Abstract Random Forest is a robust and powerful ensemble classifier that is known to perform well on bioinformatics data. However, the Random Forest algorithm does not take into account the level of class imbalance that is a common problem within this domain and imposes such complications as bias towards the majority class and decreased classification performance. In this study, we seek to determine if the inclusion of data sampling will improve the performance of the Random Forest classifier. We test the effect of data sampling using three data sampling techniques coupled with two post-sampling class distribution ratios. Additionally, we built inductive models with Random Forest when no data sampling technique was applied, so we can observe the true effect of the data sampling. Lastly, we utilize three feature selection techniques, four feature subset sizes, and fifteen imbalanced bioinformatics datasets. Our results show that, in general, data sampling does improve the classification performance of Random Forest. However, statistical analysis shows that the increase in performance is not statistically significant. Thus, we can state that while data sampling does improve the classification performance of Random Forest, it is not a necessary step as the classifier is fairly robust to imbalanced data on its own. Note, this work is an extension of our previous work "The Effect of Data Sampling When Using Random Forest on Imbalanced Bioinformatics Data" [13] with more experimental results.

D.J. Dittman · T.M. Khoshgoftaar (✉) · A. Napolitano
Florida Atlantic University, Boca Raton, FL 33431, USA
e-mail: khoshgof@fau.edu

D.J. Dittman
e-mail: ddittman@fau.edu

A. Napolitano
e-mail: amrifau@gmail.com

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_7

1 Introduction

Ensemble learning seeks to improve performance through combining the results of multiple models into a single final decision. This process has shown to be a powerful tool for bioinformatics research [10]. Among ensemble learners, Random Forest has shown to be an effective option for achieving quality performance from bioinformatics data.

Random Forest, as the name suggests, consists of a series of unpruned decision trees. Each tree model will make a decision on a new instance and the final decision is a majority vote of the collected trees. Random Forest obtains its ensemble diversity by training each tree on a bootstrapped (sampling with replacement) dataset generated from the training dataset and considering only a random subset of features for each node of each tree. There are a number of advantages to using Random Forest over many other learners, such as robustness to outliers and noise and simplicity of use [4, 20].

However, Random Forest does not take into account the possible class imbalance of the dataset. This can be an issue as class imbalance is known for increasing the bias towards the majority class (which is further compounded by the fact the frequently the class of interest is the minority class) and reducing classification performance. One option to correct the balance issue is to utilize data sampling.

Data sampling is a process which is designed to balance the dataset through the addition or removal of instances until the desired balance level is achieved. The alteration of the dataset can be performed in a number of ways, including the random removal of instances from the majority class, random duplication of instances from the minority class, or the creation and addition of synthetic data instances based on the instances that populate the minority class.

In this study, (which is an extension of our previous work "The Effect of Data Sampling When Using Random Forest on Imbalanced Bioinformatics Data" [13]) we seek to answer the question, does data sampling improve the classification performance of Random Forest? We combine Random Forest with three data sampling techniques: one from the original work (Random Undersampling) and two other techniques (Random Oversampling, and Synthetic Minority Oversampling TEchnique or SMOTE) to further extend our results. All three techniques use two post-sampling class distribution ratios, 35:65 and 50:50. As with the original work, we use Random Forest without data sampling applied to observe the effect of the inclusion of data sampling. In order to test the effect of data sampling, we use a collection of fifteen imbalanced bioinformatics datasets, three feature rankers, and four feature subset sizes. It should be noted that the purpose of this work is not to determine which data sampling technique is best suited toward bioinformatics data, but to determine the affect of data sampling on the classification performance of Random Forest.

Our results show that data sampling does consistently improve the classification performance of Random Forest. For two of the data sampling techniques (Random Undersampling and Random Oversampling) at no point is the option of no data sampling the top performing decision. When using SMOTE, there is only one scenario

in which no data sampling is the best decision. Further statistical analysis shows that data sampling improves the performance of Random Forest. However, this increase in performance is not statistically significant. Thus, we can say that Random Forest is relatively robust with imbalanced bioinformatics datasets and that the inclusion of data sampling is not required, though a small improvement may be possible when using it. To our knowledge, this is the first work which specifically seeks to observe the effect of data sampling on the Random Forest classifier in the domain of bioinformatics.

The remainder of this paper is organized as follows: Sect. 2 outlines previous research that is relevant to our present work. In Sect. 3, the details of our data sampling process and the Random Forest classifier are given. Section 4 presents methods used to perform our empirical study, including datasets, feature ranking techniques along with feature subset sizes, cross-validation process, and performance metric. In Sect. 5, we present our results with discussions of our findings. Finally, in Sect. 6, we present our conclusions and potential avenues of future work.

2 Related Works

Random Forest is a robust and powerful classifier and has been applied to the domain of bioinformatics in the past. One example is a study performed by Diaz-Uriarte et al. [6] in which the Random Forest classifier was applied to a series of ten DNA microarray datasets focusing on different areas of the body. Another is a 2011 study performed by Dittman et al. [8] which used Random Forest on a pair of DNA microarray datasets with the goal of predicting a patient's response to a drug treatment. Both studies used Random Forest to great effect on bioinformatics datasets and the classifier was reliably the top performing classifier in both studies.

However, Random Forest is not just powerful, but relatively easy to utilize. Dittman et al. [11] sought to discover if it was possible to simplify the machine learning process into a framework that is easy to implement but produces consistently good results. They used a series of twenty-five DNA microarray datasets, twenty-four feature selection techniques, and six classifiers including Random Forest. The results found that Random Forest is not only frequently the top performing classifier, but produces good results despite the choice of feature selection technique as long as feature selection occurs.

In a study by Wald et al. [29], the authors focused on the reliability of classification models on bioinformatics data. They tested a series of six classifiers including Random Forest. The results show that Random Forest is clearly the top performing classifier, not only achieving the best classification performance, but having the smallest range between their best and worst performances of all the classifiers. Lastly,

the results show that using a feature subset size of more than 200 features does not produce statistically significantly better results, so the recommendation was to use a maximum of 200 features.

However, all of these works do not take into account class imbalance. The problem is that many classification algorithms assume that the classes will have an equal number of instances in the dataset [17]. This assumption can lead to some serious problems, including increased bias towards the majority class (whereas the class of interest is frequently the minority class) and an increased number of misclassifications [2]. One recommendation for combating some of these issues is applying data sampling methods [12].

Our research group has done research on the effect of data sampling on bioinformatics data. In Khoshgoftaar et al. [19] we focused on the approach of combining data sampling with feature selection. There are three options, depending on if data sampling or feature selection is performed first and, if data sampling is the first step, whether to build the model from the sampled data or use the original training instances (i.e. the data sampling only affects which features are selected). Our results show that performing data sampling first followed by feature selection and using the unsampled data to build the model is the best approach. Another work by Dittman et al. [12] looked at which data sampling technique to use. We compared the results of Random Undersampling, Random Oversampling, and Synthetic Minority Over Sampling TEchnique (SMOTE). Our results show that there is no statistical difference between the three techniques and that Random Undersampling is preferred as it was the most frequent top performing technique.

3 Data Sampling and Random Forest

As the focus of this work is to determine if data sampling will improve the classification performance of Random Forest, we will discuss the particulars of these two topics in this section.

3.1 Data Sampling

Data sampling is a data preprocessing technique that can be used to combat class imbalance. The process seeks to modify the dataset so as to have a more balanced class distribution. This is achieved by either the removal of instances from the majority class or the addition of instances to the minority class. Additionally, the modification process can be conducted randomly and the addition of instances can use duplicates of existing instances or synthetically created instances based on the existing ones.

In this work, we use three different data sampling techniques: random undersampling, random oversampling, and Synthetic Minority Oversampling TEchnique, or SMOTE [1, 12]. Random undersampling (RUS) seeks to create balance between the two classes by reducing the size of the majority class. This is accomplished by randomly removing instances from the majority class until the desired class ratio has been achieved. Alternatively, random oversampling (ROS) seeks to improve the class balance by increasing the size of the minority class. The increase is performed by randomly duplicating instances from the minority class until the desired class ratio is achieved.

SMOTE is another form of oversampling which seeks to improve the balance between the two classes by increasing the size of the minority class. However, unlike random oversampling, SMOTE does not duplicate instances. Instead SMOTE creates new minority instances using the original ones as a basis. It starts with an instance from the minority class and looks at a collection of its nearest neighbors (we use 5 neighbors in this work) and selects one at random. Once the neighbor has been selected, the differences between the two instances in terms of each feature is calculated. Finally a new instance is created by adding the product of the differences calculated and a random number between 0 and 1 to the original instance.

We use two different post-sampling class distribution ratios: 35:65 and 50:50. The two ratios were chosen because 50:50 is a perfectly balanced dataset and 35:65 performs as well as 50:50 but is a less aggressive class distribution ratio which when applied to RUS will result in reduced data loss compared to 50:50 [7, 19]. Additionally, we performed data sampling prior to feature selection (see Sect. 4.2) and use the selected features along with the unaltered training dataset to train the classifier. We chose this specific process due to previous research indicating that they are the top performing options when it comes to data sampling [12, 19].

3.2 Random Forest

Random Forest (see Fig. 1) was developed in 2001 [4] as an ensemble learning approach based on the principles of sampling with replacement. The Random Forest learner constructs a set of unpruned decision trees, each built from a dataset created by performing sampling with replacement on the training dataset until the number of instances matches that of the training dataset. The Random Forest algorithm selects a random subset of the features for each node. In this study, we use Random Forest with 100 trees because earlier research shows that it is the optimum number of trees [21]. We used the Weka [34] implementation of the Random Forest algorithm where the number of trees used to build the forest is determined by the *numTrees* parameter.

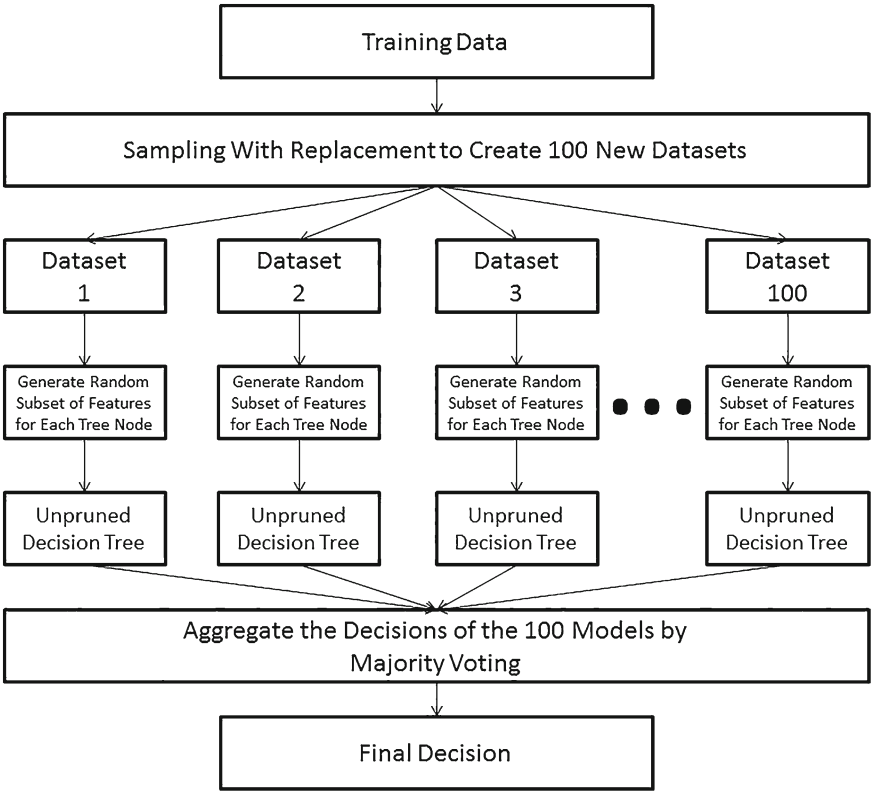


Fig. 1 Random forest with 100 trees

4 Methodology

In this section, we present multiple aspects of our experiments, including datasets, feature selection techniques and feature subset sizes, learners, cross-validation, and performance metric (Area Under the Receiver Operating Characteristic Curve or AUC).

4.1 Datasets

Table 1 contains the list of datasets used in this study along with their characteristics. All 15 datasets are DNA microarray datasets publicly available from a number of different bioinformatics and medical projects (See provided citations in Table 1 for more information on these datasets). For each dataset, we present the name, total number of minority-class instances, total number of instances, minority class

Table 1 Details of the datasets

Name	# Minority instances	Total # of instances	% Minority instances (%)	# of attributes	Average AUC
Brain Tumor [28]	23	90	25.56	27680	0.7210
ECML Pancreas [27]	8	90	8.89	27680	0.6723
GSE1456 [24]	40	159	25.16	12066	0.6108
GSE20271 [26]	26	178	14.61	22284	0.5867
GSE25055 [16]	57	306	18.63	22284	0.6674
GSE25065 [16]	42	182	23.08	22284	0.6384
GSE3494-GPL96-ER [23]	34	247	13.77	22284	0.7688
GSE3494-GPL96-Grade [23]	54	249	21.69	22284	0.8176
GSE3494-GPL97-ER [23]	34	247	13.77	22646	0.7674
GSE3494-GPL97-Grade [23]	54	249	21.69	22646	0.7722
Lung 50k [9]	70	400	17.50	54614	0.8150
Ovarian MAT [5]	16	66	24.24	6001	0.7896
Raponi 2007 No SD [25]	10	54	18.52	22284	0.4420
Raponi 2007 R+SD [25]	14	58	24.14	22284	0.4739
Watanabe 2006 [32]	11	46	23.91	12626	0.4487

percentage, the number of gene probes (features), and the average Area Under the Receiver Operating Characteristic Curve (denoted as AUC and further described in Sect. 4.3) values for all datasets. We chose these datasets because they exhibit large levels of class imbalance (the largest minority class percentage is 25.56%).

The last column, Average AUC, represents the level of dataset difficulty. This average AUC value is based on classification models built on raw data using no pre-processing technique such as feature selection and/or data sampling. To create these AUC scores, five-fold cross-validation was employed and the average performance from six classification learners was used: Naïve Bayes, Multilayer Perceptron, 5 Nearest Neighbors, Support Vector Machines, and two versions of C4.5 decision trees (one using default parameter values in the Weka data mining toolset, one using Laplace smoothing and no pruning [33]). For more information on these classifiers please refer to Khoshgoftaar et al. [19]. These average AUC values are used to show that the datasets being used do not represent trivial classification tasks, but have no further bearing in this work.

4.2 Feature Selection Technique and Feature Subset Size

In this experimental study, we use three different types of filter-based feature selection techniques: Information Gain (IG) [15], Area Under the ROC Curve (ROC) [30],

and Signal-to-Noise (S2N) [18]. We use feature ranking techniques because filter- and wrapper-based subset selection techniques can be computationally prohibitive with the high-dimensional datasets commonly found in bioinformatics domain. We provide a brief description of each feature ranker and refer interested readers to the references for more information.

IG [15] is one of the simplest and fastest feature ranking techniques, and is thus popular in bioinformatics where high dimensionality makes some of the more complex techniques infeasible. IG determines the significance of a feature based on the amount by which the entropy of the class decreases and information increases when considering that feature.

ROC [30] is a "Threshold-Based Feature Selection" (TBFS) technique used in conjunction with the performance metric of Area Under the Receiver Operating Characteristic Curve (AUC). TBFS treats feature values as ersatz posterior probabilities and classifies instances based on these probabilities, allowing us to use performance metrics as filter-based feature selection techniques. The TBFS technique which uses ROC aims to measure and optimize the balance between True Positive Rate (TPR) and False Positive Rate (FPR) across all decision thresholds. The larger the area, the more relevant the feature is.

The S2N ratio represents how well a feature distinguishes instances of two classes. Its equation is as follows:

$$\text{S2N} = (\mu_P - \mu_N)/(\sigma_P + \sigma_N)$$

where μ_P and μ_N represent the respective means of the positive and negative class, and σ_P and σ_N are the corresponding standard deviations [31]. The more relevant features have the larger S2N ratios.

As all of these techniques are feature rankers, we must decide on how many features to use. In this work we chose four feature subset sizes: 25, 50, 100, and 200. These sizes were chosen after being determined appropriate by previous research [29].

4.3 Cross-Validation and Performance Metric

Cross-validation [22] is the process of splitting the original dataset into N approximately equal-size partitions (folds), building the model using $(N-1)$ of these folds, then testing the built model using the Nth fold. This process is repeated N times so that each fold is used $(N-1)$ times to build the models and used only once to test the built model. The advantage of N-fold cross-validation over random sub-sampling is that all instances are used for both training and testing, and each instance is used only once per run for evaluating purposes. In this study, we used four runs of five-fold cross-validation to reduce any bias due to randomness. It should be noted that the data sampling and feature selection steps are performed on each training dataset generated by the cross-validation process.

We use the Area Under the Receiver Operating Characteristic Curve (AUC) [14] to assess the performance of all classification models. The curve plots the True Positive Rate (TPR) versus the False Positive Rate (FPR) across all decision boundaries. The area under the curve represents the quality of the model. It should be noted that the AUC described here is different from the feature selection technique mentioned in Sect. 4.2. To prevent any confusion, we use the notation AUC for the classification metric and ROC for the feature ranking technique.

5 Results

Tables 2, 3, and 4 contain the results of our experiments. Each table contains the results from one of the three data sampling techniques and each value in the tables are the average AUC value across the four runs of five-fold cross-validation for all fifteen datasets where the choice of post-sampling class distribution ratio (this includes no data sampling), feature ranker, and subset size is kept static. Additionally, we put the best performing post for every combination of feature ranker and feature subset size in **boldface** and the worst performing option is in *italics*.

When looking at the results for RUS, we see that for all but one combination of feature ranker and feature subset size (ROC with 200 features, which prefers RUS with the 35:65 post-sampling class distribution ratio) that using RUS with a post-sampling class distribution ratio of 50:50 produces the best results. Additionally, in

Table 2 Classification results: random forest using RUS

# of features	IG			ROC			S2N		
	None	35:65	50:50	None	35:65	50:50	None	35:65	50:50
25	*0.74849*	0.75977	**0.76176**	0.76384	*0.76288*	**0.76654**	*0.74102*	0.74968	**0.75621**
50	*0.75654*	0.76070	**0.76590**	*0.75895*	0.76425	**0.77242**	*0.74456*	0.75438	**0.75822**
100	0.76569	*0.76426*	**0.77492**	*0.76107*	0.76331	**0.77375**	*0.75503*	0.75936	**0.76635**
200	0.76941	*0.76793*	**0.77502**	*0.76392*	**0.77175**	0.77161	*0.75622*	0.75788	**0.76870**

Table 3 Classification results: random forest using ROS

# of features	IG			ROC			S2N		
	None	35:65	50:50	None	35:65	50:50	None	35:65	50:50
25	0.74849	**0.75495**	*0.74676*	0.76384	**0.76762**	*0.76369*	*0.74102*	**0.74866**	0.74633
50	*0.75654*	0.75957	**0.76586**	*0.75895*	**0.77035**	0.76780	*0.74456*	0.74897	**0.74938**
100	*0.76569*	0.76889	**0.77020**	*0.76107*	**0.76885**	0.76858	*0.75503*	0.75558	**0.75806**
200	0.76941	**0.77192**	*0.76464*	*0.76392*	0.76542	**0.76766**	*0.75622*	0.75791	**0.76199**

Table 4 Classification results: random forest using SMOTE

# of features	IG			ROC			S2N		
	None	35:65	50:50	None	35:65	50:50	None	35:65	50:50
25	*0.74849*	0.76312	**0.76352**	**0.76384**	0.75836	*0.75348*	*0.74102*	0.74264	**0.74613**
50	*0.75654*	**0.77073**	0.76878	*0.75895*	**0.76456**	0.76182	*0.74456*	0.75092	**0.75341**
100	*0.76569*	**0.77841**	0.77643	*0.76107*	**0.77332**	0.76637	*0.75503*	**0.76127**	0.75600
200	*0.76941*	**0.77493**	0.77167	0.76392	*0.76323*	**0.76834**	*0.75622*	0.75819	**0.76296**

the one exception the RUS with a post-sampling class distribution ratio of 50:50 follows 35:65 by only 0.00014 AUC. In terms of the worst performing data sampling option, we see that no data sampling is the worst option, with three exceptions (IG with 100 and 200 features and ROC with 25 features).

In terms of the results for ROS, we see that the post-sampling class distribution ratios of 35:65 and 50:50, are tied for the most frequent top performing result with six scenarios each. Additionally, we see that no data sampling is the most frequently the worst performing data sampling selection with nine scenarios. The post-sampling class distribution ratio of 50:50 is next with three scenarios.

Lastly, in terms of SMOTE, we see that the post-sampling class distribution ratio of 35:65 is the most frequently top performing choice with six scenarios followed by the post-sampling class distribution ratio of 50:50 with five scenarios. Additionally, we see that SMOTE is the only data sampling technique which has a scenario where using no data sampling is the top performing option (using ROC with 200 features). Looking at the worst performing results, we see that using no data sampling is the most frequently lowest performing option for data sampling with 10 scenarios followed by the post-sampling class distribution ratios of 35:65 and 50:50 with one scenario each. This leads us to state that the use of data sampling is in general beneficial to the classification performance when using Random Forest.

However, we must determine if the improvement generated by including data sampling is a statistically significant improvement. To this end, we performed an ANalysis Of VAriance (ANOVA) [3] where the factor is the choice of data sampling option. Table 5 represents the results of the ANOVA tests. We chose a significance level of 5 % for this ANOVA analysis; thus a "Prob>F" score of less than 0.05 is considered to be statistically significant. The results show that the differences between a post-sampling class distribution ratio of 50:50, 35:65, and no data sampling, is not significantly different for any of the data sampling techniques. However, for the purposes of better visualization, we performed a multiple comparison test with Tukey's Honestly Significant Difference (HSD) [3] criterion to find out how the three data sampling options compare to each other.

Figures 2, 3 and 4 illustrate the multiple comparison for all the three data sampling options. The figures illustrate each group mean by a symbol (∘) and 95 % confidence interval as a line around the symbol. While none of the results are not significantly

Table 5 ANOVA results

Data sampling	Source	Sum sq.	d.f.	Mean sq.	F	Prob > F
RUS	Option	0.203	2	0.10145	2.05	0.1288
	Error	534.406	10797	0.0495		
	Total	534.609	10799			
ROS	Order	0.043	2	0.02129	0.42	0.6541
	Error	541.312	10797	0.05014		
	Total	541.355	10799			
SMOTE	Order	0.082	2	0.04105	0.83	0.4367
	Error	534.839	10797	0.04954		
	Total	534.921	10799			

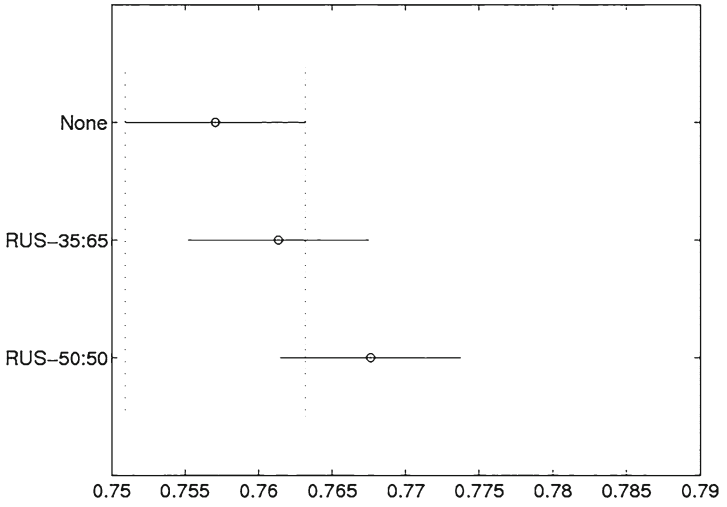


Fig. 2 Tukey's HSD results: data sampling—RUS

different from each other, we do see that there is improvement when performing data sampling compared to when no data sampling occurs, though there is very little difference between the two post-sampling class distribution ratios. Thus, we can state that though data sampling does improve classification performance of the Random Forest classifier, it is not a significant increase which indicates that Random Forest is relatively robust towards imbalanced data before the inclusion of data sampling and as a result makes the inclusion of data sampling not a requirement when using Random Forest.

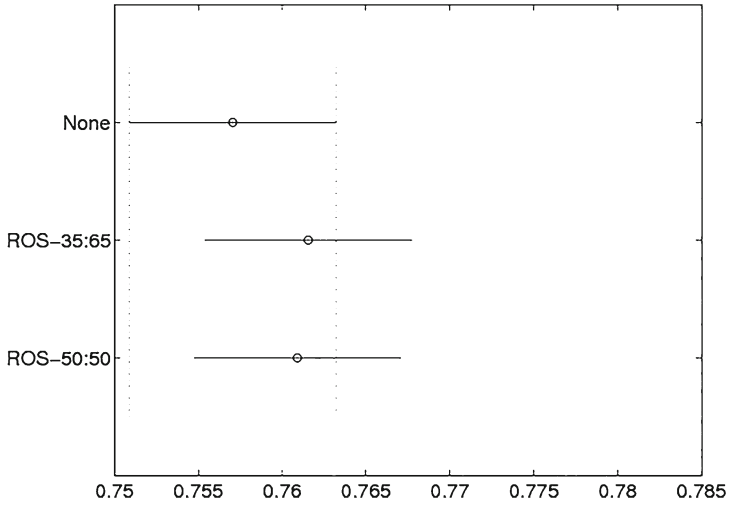


Fig. 3 Tukey's HSD results: data sampling—ROS

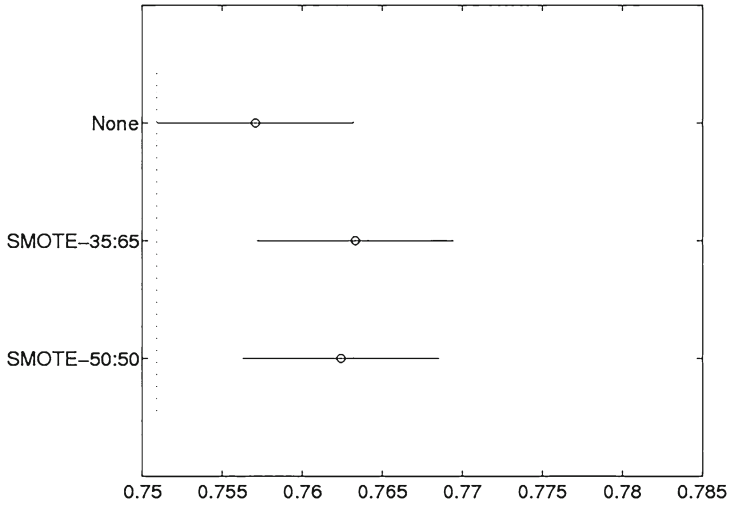


Fig. 4 Tukey's HSD results: data sampling—SMOTE

6 Conclusions

Random Forest is a powerful and effective classifier for bioinformatics datasets. However, Random Forest does not account for class imbalance. In this study, we sought to determine if the inclusion of data sampling would improve the classification performance of the Random Forest classifier. To investigate this we use three data

sampling techniques, three different post sampling class distribution ratios (50:50 and 35:65) along with no data sampling, and fifteen imbalanced bioinformatics datasets.

Our results indicate that the inclusion of data sampling does, in general, improve the classification performance for almost all scenarios. However, statistical analysis shows that the inclusion of data sampling does not significantly improve the classification performance when using Random Forest. Therefore, we can conclude that while data sampling is beneficial, it is not necessary to include it when using Random Forest because the classifier is relatively robust in terms of handling imbalanced data. Future work will consist of using datasets with different objectives within bioinformatics (patient re-admittance prediction, surgical recovery rate, etc.).

Acknowledgments The authors gratefully acknowledge partial support by the National Science Foundation, under grant number CNS-1427536. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Abu Shanab, A., Khoshgoftaar, T.M., Wald, R., Napolitano, A.: Impact of noise and data sampling on stability of feature ranking techniques for biological datasets. In: 2012 IEEE International Conference on Information Reuse and Integration (IRI), pp. 415–422, Aug 2012
2. Al-Shahib, A., Breitling, R., Gilbert, D.: Feature selection and the class imbalance problem in predicting protein function from sequence. Appl. Bioinform. **4**(3), 195–203 (2005). http://www.ingentaconnect.com/content/adis/abi/2005/00000004/00000003/art00004
3. Berenson, M.L., Goldstein, M., Levine, D.: Intermediate Statistical Methods and Applications: A Computer Package Approach, 2nd edn. Prentice Hall (1983)
4. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
5. Chen, X., Wasikowski, M.: Fast: a ROC-based feature selection metric for small samples and imbalanced data classification problems. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08), pp. 124–132. ACM, New York, NY (2008)
6. Diaz-Uriarte, R., Alvarez de Andres, S.: Gene selection and classification of microarray data using random forest. BMC Bioinform. **7**, 1–13 (2006)
7. Dittman, D.J., Khoshgoftaar, T.M., Napolitano, A.: Selecting the appropriate data sampling approach for imbalanced and high-dimensional bioinformatics datasets. In: 2014 14th IEEE International Conference on Bioinformatics and Bioengineering (BIBE), pp. 304–310 (2014)
8. Dittman, D.J., Khoshgoftaar, T.M., Wald, R., Napolitano, A.: Random forest: a reliable tool for patient response prediction. In: Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM) Workshops, pp. 289–296. BIBM (2011)
9. Dittman, D.J., Khoshgoftaar, T.M., Wald, R., Van Hulse, J.: Comparative analysis of dna microarray data through the use of feature selection techniques. In: Proceedings of the Ninth IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 147–152. ICMLA (2010)
10. Dittman, D.J., Khoshgoftaar, T.M., Napolitano, A.: Selecting the appropriate ensemble learning approach for balanced bioinformatics data. In: Florida Artificial Intelligence Research Society Conference, pp. 329–334 (2015)
11. Dittman, D.J., Khoshgoftaar, T.M., Wald, R., Napolitano, A.: Simplifying the utilization of machine learning techniques for bioinformatics. In: 2013 12th International Conference on Machine Learning and Applications (ICMLA), pp. 396–403 (2013)

12. Dittman, D.J., Khoshgoftaar, T.M., Wald, R., Napolitano, A.: Comparison of data sampling approaches for imbalanced bioinformatics data. In: 27th International Conference on Florida Artificial Intelligence Society (FLAIRS), pp. 268–271 (2014)
13. Dittman, D.J., Khoshgoftaar, T.M., Napolitano, A.: The effect of data sampling when using random forest on imbalanced bioinformatics data. In: 2015 IEEE International Conference on Information Reuse and Integration (IRI), pp. 457–463, Aug 2015
14. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006). http://www.sciencedirect.com/science/article/pii/S016786550500303X
15. Hall, M.A., Holmes, G.: Benchmarking attribute selection techniques for discrete class data mining. IEEE Trans. Knowl. Data Eng. **15**(6), 392–398 (2003)
16. Hatzis, C., Pusztai, L., Valero, V., et al.: A genomic predictor of response and survival following taxane-anthracycline chemotherapy for invasive breast cancer. JAMA **305**(18), 1873–1881 (2011). http://dx.doi.org/10.1001/jama.2011.593
17. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
18. Khoshgoftaar, T.M., Dittman, D.J., Wald, R., Fazelpour, A.: First order statistics based feature selection: a diverse and powerful family of feature selection techniques. In: Proceedings of the Eleventh International Conference on Machine Learning and Applications (ICMLA): Health Informatics Workshop, pp. 151–157. ICMLA (2012)
19. Khoshgoftaar, T.M., Wald, R., Dittman, D.J., Napolitano, A.: Classification performance of three approaches for combining data sampling and gene selection on bioinformatics data. In: 2014 14th IEEE International Conference on Information Reuse and Integration (IRI), pp. 315–321 (2014)
20. Khoshgoftaar, T.M., Dittman, D.J., Wald, R., Awada, W.: A review of ensemble classification for dna microarrays data. In: 2013 IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 381–389. IEEE (2013)
21. Khoshgoftaar, T.M., Golawala, M., Van Hulse, J.: An empirical study of learning from imbalanced data using random forest. In: IEEE International Conference on Tools with Artificial Intelligence, pp. 310–317 (2007)
22. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI **14**, 1137–1145 (1995)
23. Miller, L.D., Smeds, J., George, J., Vega, V.B., Vergara, L., Ploner, A., Pawitan, Y., Hall, P., Klaar, S., Liu, E.T., Bergh, J.: An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival. In: Proceedings of the National Academy of Sciences of the United States of America **102**(38), 13550–13555 (2005). http://www.pnas.org/content/102/38/13550.abstract
24. Pawitan, Y., Bjohle, J., Amler, L., Borg, A.L., Egyhazi, S., Hall, P., Han, X., Holmberg, L., Huang, F., Klaar, S., Liu, E., Miller, L., Nordgren, H., Ploner, A., Sandelin, K., Shaw, P., Smeds, J., Skoog, L., Wedren, S., Bergh, J.: Gene expression profiling spares early breast cancer patients from adjuvant therapy: derived and validated in two population-based cohorts. Breast Cancer Res. **7**(6), R953–R964 (2005). http://breast-cancer-research.com/content/7/6/R953
25. Raponi, M., Harousseau, J.L., Lancet, J.E., Lwenberg, B., Stone, R., Zhang, Y., Rackoff, W., Wang, Y., Atkins, D.: Identification of molecular predictors of response in a study of tipifarnib treatment in relapsed and refractory acute myelogenous leukemia. Clin. Cancer Res. **13**(7), 2254–2260 (2007). http://clincancerres.aacrjournals.org/content/13/7/2254.abstract
26. Tabchy, A., Valero, V., Vidaurre, T., Lluch, A., Gomez, H., Martin, M., Qi, Y., Barajas-Figueroa, L.J., Souchon, E., Coutant, C., Doimi, F.D., Ibrahim, N.K., Gong, Y., Hortobagyi, G.N., Hess, K.R., Symmans, W.F., Pusztai, L.: Evaluation of a 30-gene paclitaxel, fluorouracil, doxorubicin, and cyclophosphamide chemotherapy response predictor in a multicenter randomized trial in breast cancer. Clin. Cancer Res. **16**(21), 5351–5361 (2010). http://clincancerres.aacrjournals.org/content/16/21/5351.abstract
27. Van Hulse, J., Khoshgoftaar, T.M., Napolitano, A., Wald, R.: Feature selection with high-dimensional imbalanced data. In: 2009 IEEE International Conference on Data Mining Workshops, ICDMW'09, pp. 507–514, Dec 2009

28. Van Hulse, J., Khoshgoftaar, T.M., Napolitano, A., Wald, R.: A comparative evaluation of feature ranking methods for high dimensional bioinformatics data. In: Proceedings of the IEEE International Conference on Information Reuse and Integration—IRI'11, pp. 315–320 (2011)
29. Wald, R., Khoshgoftaar, T.M., Dittman, D.J., Napolitano, A.: Random forest with 200 selected features: an optimal model for bioinformatics research. In: 2013 12th International Conference on Machine Learning and Applications (ICMLA), vol. 1, pp. 154–160, Dec 2013
30. Wang, H., Khoshgoftaar, T.M., Van Hulse, J.: A comparative study of threshold-based feature selection techniques. In: 2010 IEEE International Conference on Granular Computing (GrC), pp. 499–504 (2010)
31. Wasikowski, M., wen Chen, X.: Combating the small sample class imbalance problem using feature selection. IEEE Trans. Knowl. Data Eng. **22**, 1388–1400 (2010)
32. Watanabe, T., Komuro, Y., Kiyomatsu, T., Kanazawa, T., Kazama, Y., Tanaka, J., Tanaka, T., Yamamoto, Y., Shirane, M., Muto, T., Nagawa, H.: Prediction of sensitivity of rectal cancer cells in response to preoperative radiotherapy by DNA microarray analysis of gene expression profiles. Cancer Res. **66**(7), 3370–3374 (2006). http://cancerres.aacrjournals.org/content/66/7/3370.abstract
33. Weiss, G.M., Provost, F.J.: Learning when training data are costly: the effect of class distribution on tree induction. J. Artif. Intell. Res. (JAIR) **19**, 315–354 (2003)
34. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd edn. Morgan Kaufmann (2011)

Concurrent Alignment of Multiple Anonymized Social Networks with Generic Stable Matching

Jiawei Zhang, Qianyi Zhan and Philip S. Yu

Abstract Users nowadays are normally involved in multiple (usually more than two) online social networks simultaneously to enjoy more social network services. Some of the networks that users are involved in can share common structures either due to the analogous network construction purposes or because of the similar social network characteristics. However, the social network datasets available in research are usually pre-anonymized and accounts of the shared users in different networks are mostly isolated without any known connections. In this paper, we want to identify such connections between the shared users' accounts in multiple social networks (which are called the anchor links), and the problem is formally defined as the M-NASA (Multiple Anonymized Social Networks Alignment) problem. M-NASA is very challenging to address due to (1) the lack of known anchor links to build models, (2) the studied networks are anonymized, where no users' personal profile or attribute information is available, and (3) the "*transitivity law*" and the "*one-to-one property*" based constraints on anchor links. To resolve these challenges, a novel two-phase network alignment framework UMA (Unsupervised Multi-network Alignment) is proposed in this paper. Extensive experiments conducted on multiple real-world partially aligned social networks demonstrate that UMA can perform very well in solving the M-NASA problem.

This paper is an extended version of PNA: Partial Network Alignment with Generic Stable Matching accepted by IEEE IRI 2015 [32].

J. Zhang (✉) · P.S. Yu
University of Illinois at Chicago, Chicago, IL, USA
e-mail: jzhan9@uic.edu

Q. Zhan
National Laboratory for Novel Software Technology,
Nanjing University, Nanjing, China
e-mail: zhanqianyi@gmail.com

P.S. Yu
Institute for Data Science, Tsinghua University, Beijing, China
e-mail: psyu@cs.uic.edu

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_8

Keywords Partial network alignment · Multiple heterogeneous social networks · Data mining

1 Introduction

As proposed in [13], people nowadays are normally involved in multiple (usually *more than two*) social networks simultaneously to enjoy more social network services. Many of these networks can share common structure information (e.g., friendship connections) due to either the analogous network establishing purposes or because of similar network characteristics. Meanwhile, social network data available for research is usually anonymized for privacy concerns [2], where users' personal profile and attribute information (e.g., names, hometown, gender and age) is either removed or replaced with meaningless unique identifiers, and the accounts of the shared users in these anonymized social networks are mostly isolated without any correspondence relationships. In this paper, we want to study the "*Multiple Anonymized Social Networks Alignment*" (M-NASA) problem to identify such correspondence relationships between the shared users' accounts across multiple anonymized social networks.

By following terminology definitions used in existing aligned networks studies [13, 37], social networks sharing common users are defined as "*partially aligned networks*", where the shared users are named as "*anchor users*" [37] and the correspondence relationships between anchor users' accounts in different networks are called "*anchor links*" [13]. The M-NASA problem studied in this paper aims at identifying the anchor links among multiple anonymized social networks. To help illustrate the M-NASA problem more clearly, we also give an example in Fig. 1, which involves 3 different social networks (i.e., networks I, II and III). Users in these 3 networks are all anonymized and their names are replaced with randomly

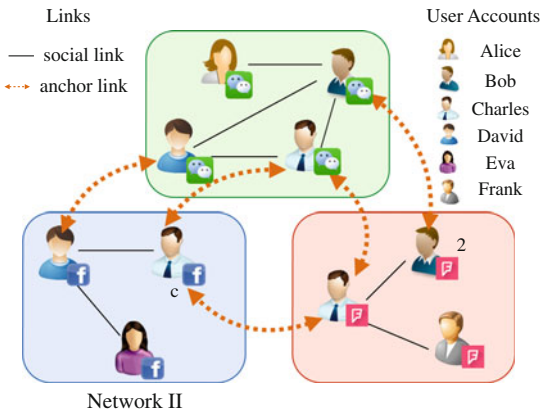


Fig. 1 An example of multiple anonymized partially aligned social networks

generated identifiers. Each pair of these 3 anonymized networks can actually share some common users, e.g., "David" participates in both networks I and II simultaneously, "Bob" is using networks I and III concurrently, and "Charles" is involved in all these 3 networks at the same time. Besides these shared anchor users, in these 3 partially aligned networks, some users are involved in one single network only (i.e., the non-anchor users [37]), e.g., "Alice" in network I, "Eva" in network II and "Frank" in network III. The M-NASA problem studied in this paper aims at discovering the anchor links (i.e., the dashed bi-directional orange lines) connecting anchor users across these 3 social networks.

The M-NASA problem is of great importance for online social networks, as it can be the prerequisite for various cross-site social network services, e.g., cross-network link transfer [37], inter-network community detection [34], and viral marketing across networks [31]. With the information transferred from developed social networks, link prediction models proposed in [37] can overcome the *cold-start problem* effectively; constrained by the anchor links, community detection across aligned networks can refine the community structures of each social network mutually [10, 34]; via the anchor users, information can diffuse not only within but also across networks which will lead to broader impact and activate more users in viral marketing [31].

Besides its importance, the M-NASA problem is a novel problem and totally different from existing works, e.g., (1) *supervised anchor link inference across social networks* [13], which focuses on inferring the anchor links between *two* social networks with a supervised learning model; (2) *network matching* [12, 18], which explores various heuristics to match *two* networks based the known existence probabilities of potential correspondence relationships; (3) *entity resolution* [4], which aims at discovering multiple references to the same entity in *one single database* with a relational clustering algorithm; and (4) *cross-media user identification* [30], which matches users between *two* networks based on various node attribute information generated by users' social activities.

M-NASA differs from all these related works in various aspects: (1) M-NASA is a general multi-network alignment problem and can be applied to align either two [13] or more than two social networks; (2) M-NASA is an *unsupervised* network alignment problem and requires no known anchor links (which are also extremely expensive to obtain in the real world); (3) no extra heuristics will be needed and used in the M-NASA problem; and (4) no information about the potential anchor links nor their existence probabilities is required; and (5) social networks studied in M-NASA are anonymized and involve structure information only but no attribute information.

Besides these easily distinguishable distinctions mentioned above, another significant difference of M-NASA from existing *two* network alignment problems is due to the "*transitivity law*" that anchor links follow. In traditional set theory [15], a relation $\mathcal{R}$ is defined to be a *transitive relation* in domain $\mathcal{X}$ iff $\forall a, b, c \in \mathcal{X}, (a, b) \in \mathcal{R} \wedge (b, c) \in \mathcal{R} \rightarrow (a, c) \in \mathcal{R}$. If we treat the union of user account sets of all these social networks as the target domain $\mathcal{X}$ and treat anchor links as the relation $\mathcal{R}$, then anchor links depict a "*transitive relation*" among users across networks. We can take the networks shown in Fig. 1 as an example. Let u be a

user involved in networks I, II and III simultaneously, whose accounts in these networks are u^I, u^{II} and u^{III} respectively. If anchor links (u^I, u^{II}) and (u^{II}, u^{III}) are identified in aligning networks (I, II) and networks (II, III) respectively (i.e., u^I, u^{II} and u^{III} are discovered to be the same user), then anchor link (u^I, u^{III}) should also exist in the alignment result of networks (I, III) as well. In the M-NASA problem, we need to guarantee the inferred anchor links can meet the *transitivity law*.

In addition to its importance and novelty, the M-NASA problem is very difficult to solve due to the following challenges:

- *unsupervised network alignment*: No existing anchor links are available between pairs of social networks in the M-NASA problem and inferring anchor links between social networks in an unsupervised manner is very challenging.
- *anonymized network alignment*: Networks studied in this paper are all pre-anonymized, where no attribute information indicating users' personal characteristics exists. It makes the M-NASA problem much tougher to address.
- *transitivity law preservation and utilization*: Anchor links among social networks follow the "transitivity law". How to (1) preserve such a property of anchor links, and (2) utilize such a property to improve the multiple networks partial alignment is still an open problem in this context so far.
- *one-to-one constraint on anchor links*: Anchor links have an inherent *one-to-one* constraint [13], i.e., each user can have at most one account in each social network, which will pose extra challenges on solving the M-NASA problem. (The case that users have multiple accounts in one network can be resolved with method introduced in [27], where these duplicated accounts can be aggregated in advance to form one unique virtual account and the constraint on anchor links connecting these virtual accounts will still be "one-to-one".)

To solve the M-NASA problem, a novel network alignment framework UMA (Unsupervised Multi-network Alignment) is proposed in this paper. UMA addresses the M-NASA problem with two steps: (1) unsupervised transitive anchor link inference across multi-networks, and (2) transitive multi-network matching to maintain the constraints on anchor links. In step (1), UMA infers sets of potential anchor links with unsupervised learning techniques by minimizing the *friendship inconsistency* and preserving the *alignment transitivity* property across networks. In step (2), UMA keeps the one-to-one constraint on anchor links by selecting those with high confidence scores but no blocking pairs, while maintaining the *matching transitivity* property at the same time. The above mentioned new concepts will be introduced in Sect. 3.

The rest of this paper is organized as follows. In Sect. 2, we define some important concepts and the M-NASA problem. Method UMA will be introduced in Sect. 3 and evaluated in Sect. 4. Finally, we introduce the related works in Sect. 5 and conclude this paper in Sect. 6.

2 Problem Formulation

In this section, we will follow the definitions of "*aligned networks*" and "*anchor links*" proposed in [37], which are introduced as follows.

Definition 1 (*Anonymized Social Network*) An anonymized social network can be represented as graph $G = (\mathcal{U}, \mathcal{E})$, where $\mathcal{U}$ denotes the set of users in the network and $\mathcal{E}$ represents the *social links* among users. Users' profile and attribute information in G has all been deleted to protect individuals' privacy.

Definition 2 (*Multiple Aligned Social Networks*) Multiple aligned social networks can be represented as $\mathcal{G} = ((G^{(1)}, G^{(2)}, \ldots, G^{(n)}), (\mathcal{A}^{(1,2)}, \mathcal{A}^{(1,3)}, \ldots, \mathcal{A}^{(n-1,n)}))$, where $G^{(i)}, i \in \{1, 2, \ldots, n\}$ represents an anonymized social network and $\mathcal{A}^{(i,j)}, i, j \in \{1, 2, \ldots, n\}$ denotes the set of undirected *anchor links* between networks $G^{(i)}$ and $G^{(j)}$.

Definition 3 (*Anchor Links*) Given two social networks $G^{(i)}$ and $G^{(j)}$, link $(u^{(i)}, v^{(j)})$ is an *anchor link* between $G^{(i)}$ and $G^{(j)}$ iff $(u^{(i)} \in \mathcal{U}^{(i)}) \wedge (v^{(j)} \in \mathcal{U}^{(j)}) \wedge (u^{(i)}$ and $v^{(j)}$ are accounts of the same user), where $\mathcal{U}^{(i)}$ and $\mathcal{U}^{(j)}$ are the user sets of $G^{(i)}$ and $G^{(j)}$ respectively.

Social networks studied in this paper are all partially aligned [37] and the formal definitions of the concepts like "anchor users", "non-anchor users", "full alignment", "partial alignment" are available in [37].

Based on the above definitions, the M-NASA problem can be formulated as follows:

The M-NASA Problem: Given the n isolated anonymized social networks $\{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}$, the M-NASA problem aims at discovering the anchor links among these n networks, i.e., the anchor link sets $\mathcal{A}^{(1,2)}, \mathcal{A}^{(1,3)}, \ldots, \mathcal{A}^{(n-1,n)}$. Networks $G^{(1)}, G^{(2)}, \ldots, G^{(n)}$ are partially aligned and the constraint on anchor links in $\mathcal{A}^{(1,2)}, \mathcal{A}^{(1,3)}, \ldots, \mathcal{A}^{(n-1,n)}$ is *one-to-one*, which also need to follow the *transitivity law*.

3 Proposed Method

Based on observation about the "transitivity property" of anchor links, in this section, we will introduce the framework UMA to address the M-NASA problem: in Sect. 3.1, we formulate the unsupervised pairwise network alignment based on friendship connection information as an optimization problem; integrated multi-network alignment will be introduced in Sect. 3.2, where an extra constraint called *alignment transitivity* penalty is added to the objective function; the joint optimization function will be solved in Sect. 3.3 by relaxing its constraints, and the redundant non-existing anchor links introduced by such relaxation will be pruned with *transitive network matching* in Sect. 3.4.

3.1 Unsupervised Pairwise Network Alignment

Anchor links between any two given networks $G^{(i)}$ and $G^{(j)}$ actually define an *one-to-one* mapping (of users and social links) between $G^{(i)}$ and $G^{(j)}$. To evaluate the quality of different inferred mapping (i.e., the inferred anchor links), we introduce the concepts of cross-network *Friendship Consistency/Inconsistency* in this paper. The optimal inferred anchor links are those which can maximize the *Friendship Consistency* (or minimize the *Friendship Inconsistency*) across networks.

For any anonymized social network $G = (\mathcal{U}, \mathcal{E})$, the social connections among users in it can be represented with the *social adjacency matrix*.

Definition 4 (*Social Adjacency Matrix*) Given network $G = (\mathcal{U}, \mathcal{E})$, its *social adjacency matrix* can be represented with binary matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ and entry $\mathbf{S}(l, m) = 1$ iff the corresponding social link $(u_l, u_m) \in \mathcal{E}$, where u_l and u_m are users in G.

Based on the above definition, given two partially aligned social networks $G^{(i)} = (\mathcal{U}^{(i)}, \mathcal{E}^{(i)})$ and $G^{(j)} = (\mathcal{U}^{(j)}, \mathcal{E}^{(j)})$, we can represent their corresponding *social adjacency* matrices to be $\mathbf{S}^{(i)} \in \mathbb{R}^{|\mathcal{U}^{(i)}|\times|\mathcal{U}^{(i)}|}$ and $\mathbf{S}^{(j)} \in \mathbb{R}^{|\mathcal{U}^{(j)}|\times|\mathcal{U}^{(j)}|}$ respectively.

Meanwhile, let $\mathcal{A}^{(i,j)}$ be the set of undirected anchor links to be inferred connecting networks $G^{(i)}$ and $G^{(j)}$, based on which, we can construct the corresponding *binary transitional matrix* $\mathbf{T}^{(i,j)}$ between networks $G^{(i)}$ and $G^{(j)}$, where users corresponding to rows and columns of $\mathbf{T}^{(i,j)}$ are of the same order as those of $\mathbf{S}^{(i)}$ and $\mathbf{S}^{(j)}$ respectively.

Definition 5 (*Binary Transitional Matrix*) Given anchor link set $\mathcal{A}^{(i,j)} \subset \mathcal{U}^{(i)} \times \mathcal{U}^{(j)}$ between networks $G^{(i)}$ and $G^{(j)}$, the *binary transitional matrix* from $G^{(i)}$ to $G^{(j)}$ can be represented as $\mathbf{T}^{(i,j)} \in \{0, 1\}^{|\mathcal{U}^{(i)}|\times|\mathcal{U}^{(j)}|}$, where $\mathbf{T}^{(i,j)}(l, m) = 1$ iff link $(u_l^{(i)}, u_m^{(j)}) \in \mathcal{A}^{(i,j)}$, $u_l^{(i)} \in \mathcal{U}^{(i)}$, $u_m^{(j)} \in \mathcal{U}^{(j)}$.

The *binary transitional matrix* from $G^{(j)}$ to $G^{(i)}$ can be defined in a similar way, which can be represented as $\mathbf{T}^{(j,i)} \in \{0, 1\}^{|\mathcal{U}^{(j)}|\times|\mathcal{U}^{(i)}|}$, where $(\mathbf{T}^{(i,j)})^\top = \mathbf{T}^{(j,i)}$ as the anchor links between $G^{(i)}$ and $G^{(j)}$ are undirected. Considering that anchor links have an inherent *one-to-one* constraint, each row and each column of the *binary transitional matrices* $\mathbf{T}^{(i,j)}$ and $\mathbf{T}^{(j,i)}$ should have at most one entry filled with 1, which will constrain the inference space of potential *binary transitional matrices* $\mathbf{T}^{(i,j)}$ and $\mathbf{T}^{(j,i)}$ greatly.

Binary transitional matrix $\mathbf{T}^{(i,j)}$ defines a mapping of users from network $G^{(i)}$ to $G^{(j)}$, i.e., $\mathbf{T}^{(i,j)} : \mathcal{U}^{(i)} \to \mathcal{U}^{(j)}$. Besides the user nodes, the social links in network $G^{(i)}$ can also be projected to network $G^{(j)}$ via the binary transitional matrices $\mathbf{T}^{(i,j)}$ and $\mathbf{T}^{(j,i)}$: the *social adjacency matrix* $\mathbf{S}^{(i)}$ being mapped from $G^{(i)}$ to $G^{(j)}$ can be represented as $\mathbf{T}^{(j,i)}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}$ (i.e., $(\mathbf{T}^{(i,j)})^\top\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}$). Furthermore, considering social networks $G^{(i)}$ and $G^{(j)}$ share significant community structure overlaps, the friendship connections mapped from $G^{(i)}$ to $G^{(j)}$ (i.e., $(\mathbf{T}^{(i,j)})^\top\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}$) should be consistent with those in $G^{(j)}$ (i.e., $\mathbf{S}^{(j)}$), which can be quantified as the following cross-network *friendship consistency* formally [14].

Definition 6 (*Friendship Consistency/Inconsistency*) The *friendship consistency* between network $G^{(i)}$ and $G^{(j)}$ introduced by the cross-network mapping $\mathbf{T}^{(i,j)}$ is defined as number of shared social links between those mapped from $G^{(i)}$ and the social links in $G^{(j)}$ originally.

Meanwhile, we can define the *friendship inconsistency* as the number of non-shared social links between those mapped from $G^{(i)}$ and those in $G^{(j)}$. Based on the inferred *anchor transitional matrix* $\mathbf{T}^{(i,j)}$, the introduced *friendship inconsistency* between matrices $(\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)}$ and $\mathbf{S}^{(j)}$ can be represented as:

$$\left\| (\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)} - \mathbf{S}^{(j)} \right\|_F^2,$$

where $\|\cdot\|_F$ denotes the Frobenius norm. And the optimal *binary transitional matrix* $\bar{\mathbf{T}}^{(i,j)}$, which can lead to the minimum *friendship inconsistency* can be represented as

$$\begin{aligned} \bar{\mathbf{T}}^{(i,j)} &= \arg\min_{\mathbf{T}^{(i,j)}} \left\| (\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)} - \mathbf{S}^{(j)} \right\|_F^2 \\ s.t.\quad & \mathbf{T}^{(i,j)} \in \{0,1\}^{|\mathcal{U}^{(i)}| \times |\mathcal{U}^{(j)}|}, \\ & \mathbf{T}^{(i,j)} \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1}, \\ & (\mathbf{T}^{(i,j)})^\top \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1}, \end{aligned}$$

where the last two equations are added to maintain the *one-to-one* constraint on anchor links and $\mathbf{X} \preccurlyeq \mathbf{Y}$ iff $\mathbf{X}$ is of the same dimensions as $\mathbf{Y}$ and every entry in $\mathbf{X}$ is no greater than the corresponding entry in $\mathbf{Y}$.

3.2 Transitive Integrate Network Alignment

Isolated network alignment can work well in addressing the alignment problem of two social networks. However, in the M-NASA problem studied in this paper, multiple (more than two) social networks are to be aligned simultaneously. Besides minimizing the *friendship inconsistency* between each pair of networks, the *transitivity* property of anchor links also needs to be preserved in the transitional matrices inference.

The *transitivity* property should holds for the alignment of any n networks, where the minimum of n is 3. To help illustrate the *transitivity property* more clearly and simplify the descriptions of the model, we will use 3 network alignment as an example to introduce the M-NASA problem, which can be easily generalized to the case of n networks alignment. Let $G^{(i)}$, $G^{(j)}$ and $G^{(k)}$ be 3 social networks to be aligned concurrently. To accommodate the alignment results and preserve the *transitivity* property, we introduce the following *alignment transitivity penalty*:

Definition 7 (*Alignment Transitivity Penalty*) Let $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$ and $\mathbf{T}^{(i,k)}$ be the inferred binary transitional matrices from $G^{(i)}$ to $G^{(j)}$, from $G^{(j)}$ to $G^{(k)}$ and from $G^{(i)}$ to $G^{(k)}$ respectively among these 3 networks. The *alignment transitivity penalty* $C(\{G^{(i)}, G^{(j)}, G^{(k)}\})$ introduced by the inferred transitional matrices can be quantified as the number of inconsistent social links being mapped from $G^{(i)}$ to $G^{(k)}$ via two different alignment paths $G^{(i)} \rightarrow G^{(j)} \rightarrow G^{(k)}$ and $G^{(i)} \rightarrow G^{(k)}$, i.e.,

$$C(\{G^{(i)}, G^{(j)}, G^{(k)}\}) = \left\| (\mathbf{T}^{(j,k)})^\top (\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)} \mathbf{T}^{(j,k)} - (\mathbf{T}^{(i,k)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,k)} \right\|_F^2 .$$

Alignment transitivity penalty is a general penalty concept and can be applied to n networks $\{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}$, $n \geq 3$ as well, which can be defined as the summation of penalty introduced by any three networks in the set, i.e.,

$$C(\{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}) = \sum_{\forall \{G^{(i)}, G^{(j)}, G^{(k)}\} \subset \{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}} C(\{G^{(i)}, G^{(j)}, G^{(k)}\}).$$

The optimal *binary transitional matrices* $\bar{\mathbf{T}}^{(i,j)}$, $\bar{\mathbf{T}}^{(j,k)}$ and $\bar{\mathbf{T}}^{(k,i)}$ which can minimize friendship inconsistency and the *alignment transitivity penalty* at the same time can be represented to be

$$\begin{aligned}
\bar{\mathbf{T}}^{(i,j)}, \bar{\mathbf{T}}^{(j,k)}, \bar{\mathbf{T}}^{(k,i)} = & \arg\min_{\mathbf{T}^{(i,j)}, \mathbf{T}^{(j,k)}, \mathbf{T}^{(k,i)}} \left\| (\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)} - \mathbf{S}^{(j)} \right\|_F^2 \\
& + \left\| (\mathbf{T}^{(j,k)})^\top \mathbf{S}^{(j)} \mathbf{T}^{(j,k)} - \mathbf{S}^{(k)} \right\|_F^2 + \left\| (\mathbf{T}^{(k,i)})^\top \mathbf{S}^{(k)} \mathbf{T}^{(k,i)} - \mathbf{S}^{(i)} \right\|_F^2 \\
& + \alpha \cdot \left\| (\mathbf{T}^{(j,k)})^\top (\mathbf{T}^{(i,j)})^\top \mathbf{S}^{(i)} \mathbf{T}^{(i,j)} \mathbf{T}^{(j,k)} - \mathbf{T}^{(k,i)} \mathbf{S}^{(i)} (\mathbf{T}^{(k,i)})^\top \right\|_F^2 \\
& s.t.\ \mathbf{T}^{(i,j)} \in \{0,1\}^{|\mathcal{U}^{(i)}| \times |\mathcal{U}^{(j)}|}, \mathbf{T}^{(j,k)} \in \{0,1\}^{|\mathcal{U}^{(j)}| \times |\mathcal{U}^{(k)}|} \\
& \quad \mathbf{T}^{(k,i)} \in \{0,1\}^{|\mathcal{U}^{(k)}| \times |\mathcal{U}^{(i)}|} \\
& \quad \mathbf{T}^{(i,j)} \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1}, (\mathbf{T}^{(i,j)})^\top \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1}, \\
& \quad \mathbf{T}^{(j,k)} \mathbf{1}^{|\mathcal{U}^{(k)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1}, (\mathbf{T}^{(j,k)})^\top \mathbf{1}^{|\mathcal{U}^{(j)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(k)}| \times 1}, \\
& \quad \mathbf{T}^{(k,i)} \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(k)}| \times 1}, (\mathbf{T}^{(k,i)})^\top \mathbf{1}^{|\mathcal{U}^{(k)}| \times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(i)}| \times 1},
\end{aligned}$$

where parameter α denotes the weight of the alignment transitivity penalty term, which is set as 1 by default in this paper.

3.3 Relaxation of the Optimization Problem

The above objective function aims at obtaining the *hard* mappings among users across different networks and entries in all these *transitional matrices* are binary, which can lead to a fatal drawback: *hard assignment* can be neither possible nor

realistic for networks with star structures as proposed in [14] and the hard subgraph isomorphism [16] is NP-hard.

To overcome such a problem, we propose to relax the binary constraint of entries in transitional matrices to allow them to be real values within range [0, 1]. Each entry in the transitional matrix represents a probability, denoting the confidence of certain user-user mapping across networks. Such a relaxation can make the *one-to-one* constraint no longer hold (which will be addressed with transitive network matching in the next subsection) as multiple entries in rows/columns of the transitional matrix can have non-zero values. To limit the existence of non-zero entries in the transitional matrices, we replace the one-to-one constraint, e.g.,

$$\mathbf{T}^{(k,i)}\mathbf{1}^{|\mathcal{U}^{(i)}|\times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(k)}|\times 1}, (\mathbf{T}^{(k,i)})^{\top}\mathbf{1}^{|\mathcal{U}^{(k)}|\times 1} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(i)}|\times 1}$$

with *sparsity constraints*

$$\left\|\mathbf{T}^{(k,i)}\right\|_0 \leq t$$

instead, where term $\|\mathbf{T}\|_0$ denotes the L_0 norm of matrix $\mathbf{T}$, i.e., the number of non-zero entries in $\mathbf{T}$, and t is a small positive number to limit the non-zero entries in the matrix (i.e., the sparsity). Furthermore, in this paper, we propose to add term $\|\mathbf{T}\|_0$ to the minimization objective function, as it can be hard to determine the value of t in the constraint.

Based on the above relaxations, we can obtain the new objective function (available in the Appendix), which involves 3 variables $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$ and $\mathbf{T}^{(k,i)}$ simultaneously, obtaining the joint optimal solution for which at the same time is very hard and time consuming. We propose to address the above objective function by fixing two variables and updating the other variable alternatively with gradient descent method [1]. As proposed in [14], if during the alternating updating steps, the entries of the transitional matrices become invalid (i.e., values less than 0 or greater than 1), we apply the projection technique introduced in [14] to project (1) negative entries to 0, and (2) entries greater than 1 to 1 instead. With these processes, the updating equations of matrices $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$, $\mathbf{T}^{(k,i)}$ at step $t+1$ are given as follows

$$\mathbf{T}^{(i,j)}(t+1) = \mathbf{T}^{(i,j)}(t) - \eta^{(i,j)}\frac{\partial\mathcal{L}\left(\mathbf{T}^{(i,j)}(t), \mathbf{T}^{(j,k)}(t), \mathbf{T}^{(k,i)}(t), \beta, \gamma, \theta\right)}{\partial\mathbf{T}^{(i,j)}},$$

$$\mathbf{T}^{(j,k)}(t+1) = \mathbf{T}^{(j,k)}(t) - \eta^{(j,k)}\frac{\partial\mathcal{L}\left(\mathbf{T}^{(i,j)}(t+1), \mathbf{T}^{(j,k)}(t), \mathbf{T}^{(k,i)}(t), \beta, \gamma, \theta\right)}{\partial\mathbf{T}^{(j,k)}},$$

$$\mathbf{T}^{(k,i)}(t+1) = \mathbf{T}^{(k,i)}(t) - \eta^{(k,i)}\frac{\partial\mathcal{L}\left(\mathbf{T}^{(i,j)}(t+1), \mathbf{T}^{(j,k)}(t+1), \mathbf{T}^{(k,i)}(t), \beta, \gamma, \theta\right)}{\partial\mathbf{T}^{(k,i)}}.$$

Such an iteratively updating process will stop when all *transitional matrices* converge. In the updating equations, $\eta^{(i,j)}$, $\eta^{(j,k)}$ and $\eta^{(k,i)}$ are the gradient descent steps in updating $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$ and $\mathbf{T}^{(k,i)}$ respectively. The Lagrangian function of the objective function is available in the Appendix.

Meanwhile, considering that $\|\cdot\|_0$ is not differentiable because of its discrete values [29], we will replace the $\|\cdot\|_0$ with the $\|\cdot\|_1$ instead (i.e., the sum of absolute values of all entries). Furthermore, as all the negative entries will be projected to 0, the L_1 norm of transitional matrix $\mathbf{T}$ can be represented as $\left\|\mathbf{T}^{(k,i)}\right\|_1 = \mathbf{1}^\top \mathbf{T}^{(k,i)} \mathbf{1}$ (i.e., the sum of all entries in the matrix). In addition, the Frobenius norm $\|\mathbf{X}\|_F^2$ can be represented with trace $\mathrm{Tr}(\mathbf{X}\mathbf{X}^\top)$. The partial derivatives of function $\mathcal{L}$ with regard to $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$, and $\mathbf{T}^{(k,i)}$ are given in the Appendix.

3.4 Transitive Generic Stable Matching

Based on the transitive integrated network alignment introduced in the previous sections, we can obtain the confidence scores among users across networks, which can be used to construct user's partner preference list across networks. For instance, if the score of link $(u^{(i)}, v^{(j)})$ is greater than that of link $(u^{(i)}, w^{(j)})$ between networks $G^{(i)}$ and $G^{(j)}$, then we can user $u^{(i)}$ prefers $v^{(j)}$ to $w^{(j)}$.

However, due to the constraint relaxation, the *one-to-one* constraint on the inferred anchor links can no longer hold. In this section, we propose to apply the *transitive network matching* algorithm to help prune the redundant non-existing anchor links introduced by the constraint relaxation.

In this section, we will first briefly talk about the traditional stable matching for two networks, then we will introduce the generic stable matching for two networks. Finally, we will introduce transitive generic stable matching for multiple networks.

3.4.1 Traditional Stable Matching

Meanwhile, as proposed in [13], the *one-to-one* constraint of anchor links across *fully aligned social networks* can be met by pruning extra potential *anchor link candidates* with *traditional stable matching*. In this subsection, we will introduce the concept of traditional *stable matching* briefly.

We first use a toy example in Fig. 2 to illustrate the main idea of our solution. Suppose in Fig. 2a we are given the ranking scores from the transitive integrated network alignment. We can see in Fig. 2b that link prediction methods with a fixed threshold may not be able to predict well, because the predicted links do not satisfy the constraint of one-to-one relationship. Thus one user account in the source network can be linked with multiple accounts in the target network. In Fig. 2c, *weighted maximum matching* methods can find a set of links with maximum sum of weights. However, it is worth noting that the input scores are uncalibrated, so maximum

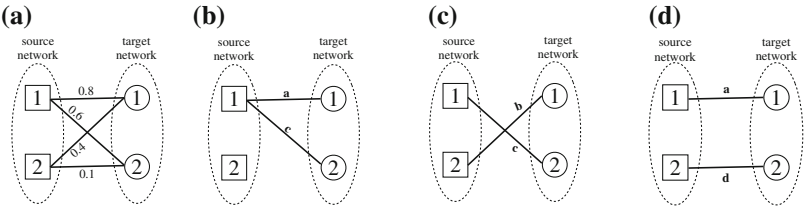


Fig. 2 An example of anchor link inference by different methods. **a** is the input, ranking scores. **b–d** are the results of different methods for anchor link inference. **a** Input scores. **b** Link prediction. **c** Max weight(1:1). **d** UMA(1:1)

weight matching may not be a good solution for anchor link prediction problems. The input scores only indicate the ranking of different user pairs, i.e., the preference relationship among different user pairs.

Here we say 'node x prefers node y over node z', if the score of pair (x, y) is larger than the score of pair (x, z). For example, in Fig. 2c, the weight of pair a, i.e., Score $(a) = 0.8$, is larger than Score $(c) = 0.6$. It shows that user u_i (the first user in the source network) *prefers* v_i over v_j. The problem with the prediction result in Fig. 2c is that, the pair (u_i, v_i) should be more likely to be an anchor link due to the following reasons: (1) u_i prefers v_i over v_j; (2) v_i also prefers u_i over u_j.

Given the user sets $\mathcal{U}^{(1)}$ and $\mathcal{U}^{(2)}$ of two *partially aligned social networks* $G^{(1)}$ and $G^{(2)}$, each user in $\mathcal{U}^{(1)}$(or $\mathcal{U}^{(2)}$) has his preference over users in $\mathcal{U}^{(2)}$(or $\mathcal{U}^{(1)}$). Term $v_j P^{(1)}_{u_i} v_k$ is used to denote that $u_i \in \mathcal{U}^{(1)}$ prefers v_j to v_k for simplicity, where $v_j, v_k \in \mathcal{U}^{(2)}$ and $P^{(1)}_{u_i}$ is the preference operator of $u_i \in \mathcal{U}^{(1)}$. Similarly, we can use term $u_i P^{(2)}_{v_j} u_k$ to denote that $v_j \in \mathcal{U}^{(2)}$ prefers u_i to u_k in $\mathcal{U}^{(1)}$ as well.

Definition 8 (*Matching*) Mapping $\mu : \mathcal{U}^{(1)} \cup \mathcal{U}^{(2)} \to \mathcal{U}^{(1)} \cup \mathcal{U}^{(2)}$ is defined to be a *matching* iff (1) $|\mu(u_i)| = 1, \forall u_i \in \mathcal{U}^{(1)}$ and $\mu(u_i) \in \mathcal{U}^{(2)}$; (2) $|\mu(v_j)| = 1, \forall v_j \in \mathcal{U}^{(2)}$ and $\mu(v_j) \in \mathcal{U}^{(1)}$; (3) $\mu(u_i) = v_j$ iff $\mu(v_j) = u_i$.

Definition 9 (*Blocking Pair*) A pair (u_i, v_j) is a a *blocking pair* of matching μ if u_i and v_j prefers each other to their mapped partner, i.e., $(\mu(u_i) \neq v_j) \wedge (\mu(v_j) \neq u_i)$ and $(v_j P^{(1)}_{u_i} \mu(u_i)) \wedge (u_i P^{(2)}_{v_j} \mu(v_j))$.

Definition 10 (*Stable Matching*) Given a matching μ, μ is *stable* if there is no *blocking pair* in the matching results [8].

We propose to formulate the anchor link prediction problem as a stable matching problem between user accounts in source network and accounts in target network. Assume that we have two sets of unlabeled user accounts, i.e., $\mathcal{U}^{(1)} = \{u_1, u_2, \ldots, u_{|\mathcal{U}^{(1)}|}\}$ in source network and $\mathcal{U}^{(2)} = \{v_1, v_2, \ldots, v_{|\mathcal{U}^{(2)}|}\}$ in target network. Each u_i has a ranking list or preference list $P(u_i)$ over all the user accounts in target network ($v_i \in \mathcal{U}^{(2)}$) based upon the input scores of different pairs. For example, in Fig. 2a, the preference list of node u_i is $P(u_i) = (v_i, v_j)$, indicating that node v_i is preferred by u_i over v_j. The preference list of node u_j is also $P(u_j) = (v_i, v_j)$.

Similarly, we also build a preference list for each user account in the target network. In Fig. 2a, $P(v_i) = P(v_j) = (u_i, u_j)$.

3.4.2 Generic Stable Matching

Stable matching based method proposed in [13] can only work well in *fully aligned social networks*. However, in the real world, few social networks are fully aligned and lots of users in social networks are involved in one network only, i.e., *non-anchor users*, and they should not be connected by any anchor links. However, traditional *stable matching* method cannot identify these *non-anchor users* and remove the predicted *potential anchor links* connected with them. To overcome such a problem, we will introduce the *generic stable matching* to identify the *non-anchor users* and prune the anchor link results to meet the *one-to-one* constraint.

In UMA, we introduce a novel concept, *self matching*, which allows users to be mapped to themselves if they are discovered to be *non-anchor users*. In other words, we will identify the *non-anchor users* as those who are mapped to themselves in the final matching results.

Definition 11 (*Self Matching*) For the given two partially aligned networks $G^{(1)}$ and $G^{(2)}$, user $u_i \in \mathcal{U}^{(1)}$, can have his preference $P^{(1)}_{u_i}$ over users in $\mathcal{U}^{(2)} \cup \{u_i\}$ and u_i preferring u_i himself denotes that u_i is an *non-anchor user* and prefers to stay unconnected, which is formally defined as *self matching*.

Users in one social network will be matched with either partners in other social networks or themselves according to their preference lists (i.e., from high preference scores to low preference scores). Only partners that users prefer over themselves will be *accepted* finally, otherwise users will be matched with themselves instead.

Definition 12 (*Acceptable Partner*) For a given *matching* $\mu : \mathcal{U}^{(1)} \cup \mathcal{U}^{(2)} \to \mathcal{U}^{(1)} \cup \mathcal{U}^{(2)}$, the mapped partner of users $u_i \in \mathcal{U}^{(1)}$, i.e., $\mu(u_i)$, is *acceptable* to u_i iff $\mu(u_i) P^{(1)}_{u_i} u_i$.

To cut off the partners with very low *preference scores*, we propose the *partial matching strategy* to obtain the promising partners, who will participate in the matching finally.

Definition 13 (*Partial Matching Strategy*) The *partial matching strategy* of user $u_i \in \mathcal{U}^{(1)}$, i.e., $Q^{(1)}_{u_i}$, consists of the first K the *acceptable partners* in u_i's preference list $P^{(1)}_{u_i}$, which are in the same order as those in $P^{(1)}_{u_i}$, and u_i in the $(K+1)$th entry of $Q^{(1)}_{u_i}$. Parameter K is called the *partial matching rate* in this paper.

An example is given at the last plot of Fig. 3, where to get the top 2 promising partners for the user, we place the user himself at the 3rd cell in the preference list. All the remaining potential partners will be cut off and only the top 3 users will participate in the final matching.

Based on the concepts of *self matching* and *partial matching strategy*, we define the concepts of *partial stable matching* and *generic stable matching* as follow.

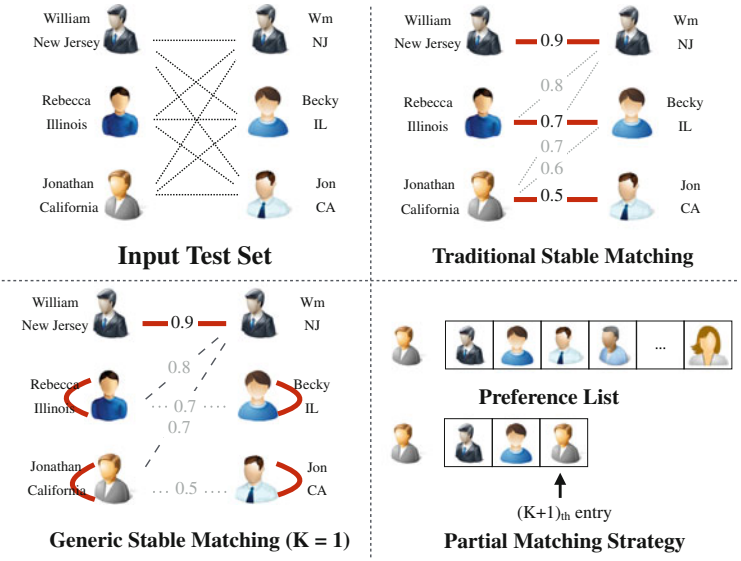


Fig. 3 Partial network alignment with pruning

Definition 14 (*Partial Stable Matching*) For a given *matching* μ, μ is (1) *rational* if $\mu(u_i)Q_{u_i}^{(1)}u_i, \forall u_i \in \mathcal{U}^{(1)}$ and $\mu(v_j)Q_{v_j}^{(2)}v_j, \forall v_j \in \mathcal{U}^{(2)}$, (2) *pairwise stable* if there exist no *blocking pairs* in the matching results, and (3) *stable* if it is both *rational* and *pairwise stable*.

Definition 15 (*Generic Stable Matching*) For a given *matching* μ, μ is a *generic stable matching* iff μ is a *self matching* or μ is a *partial stable matching*.

As example of *generic stable matching* is shown in the bottom two plots of Fig. 3. *Traditional stable matching* can prune most non-existing anchor links and make sure the results can meet *one-to-one* constraint. However, it preserves the anchor links (Rebecca, Becky) and (Jonathan, Jon), which are connecting *non-anchor users*. In *generic stable matching* with parameter $K = 1$, users will be either connected with their most preferred partner or stay *unconnected*. Users "William" and "Wm" are matched as link (William, Wm) has the highest score. "Rebecca" and "Jonathan" will prefer to stay *unconnected* as their most preferred partner "Wm" is connected with "William" already. Furthermore, "Becky" and "Jon" will stay *unconnected* as their most preferred partner "Rebecca" and "Jonathan" prefer to stay *unconnected*. In this way, *generic stable matching* can further prune the non-existing anchor links (Rebecca, Becky) and (Jonathan, Jon).

The *truncated generic stable matching* results can be achieved with the *Generic Gale-Shapley* algorithm as given in Algorithm 1.

Algorithm 1 Generalized Gale-Shapley Algorithm

Input: user sets of aligned networks: $\mathcal{U}^{(1)}$ and $\mathcal{U}^{(2)}$.
classification results of potential anchor links in $\mathcal{L}$
known anchor links in $\mathcal{A}^{(1,2)}$
truncation rate K
Output: a set of inferred anchor links $\mathcal{L}'$
Initialize the preference lists of users in $\mathcal{U}^{(1)}$ and $\mathcal{U}^{(2)}$ with predicted existence probabilities of links in $\mathcal{L}$ and known anchor links in $\mathcal{A}^{(1,2)}$, whose existence probabilities are 1.0
construct the truncated strategies from the preference lists
Initialize all users in $\mathcal{U}^{(1)}$ and $\mathcal{U}^{(2)}$ as *free*
$\mathcal{L}' = \emptyset$
while $\exists$ *free* $u_i^{(1)}$ in $\mathcal{U}^{(1)}$ and $u_i^{(1)}$'s truncated strategy is non-empty **do**
  Remove the top-ranked account $u_j^{(2)}$ from $u_i^{(1)}$'s truncated strategy
  if $u_j^{(2)}$==$u_i^{(1)}$ **then**
    $\mathcal{L}' = \mathcal{L}' \cup \{(u_i^{(1)}, u_i^{(1)})\}$
    Set $u_i^{(1)}$ as *stay unconnected*
  else
    if $u_j^{(2)}$ is *free* **then**
      $\mathcal{L}' = \mathcal{L}' \cup \{(u_i^{(1)}, u_j^{(2)})\}$
      Set $u_i^{(1)}$ and $u_j^{(2)}$ as *occupied*
    else
      $\exists u_p^{(1)}$ that $u_j^{(2)}$ is occupied with.
      if $u_j^{(2)}$ prefers $u_i^{(1)}$ to $u_p^{(1)}$ **then**
        $\mathcal{L}' = (\mathcal{L}' - \{(u_p^{(1)}, u_j^{(2)})\}) \cup \{(u_i^{(1)}, u_j^{(2)})\}$
        Set $u_p^{(1)}$ as *free* and $u_i^{(1)}$ as *occupied*
      end if
    end if
  end if
end while

3.4.3 Transitive Generic Stable Matching

To ensure the network matching results can meet the "*transitivity law*", in matching networks $(G^{(i)}, G^{(j)})$, $(G^{(j)}, G^{(k)})$ and $(G^{(k)}, G^{(i)})$, we need to consider the results globally. For instance,when matching these 3 networks, we can match networks $(G^{(j)}, G^{(k)})$ with Algorithm 1, which is identical to the regular pairwise network matching problem. Next, we can match networks $(G^{(i)}, G^{(j)})$. If we identify $(u^{(i)}, v^{(j)})$ and $(v^{(j)}, w^{(k)})$ should be matched between networks $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$ respectively, we will follow the following strategy to either pre-add $(w^{(k)}, u^{(i)})$ to the alignment result between networks $(G^{(k)}, G^{(i)})$ or separate pair $(u^{(i)}, v^{(j)})$ and set $u^{(i)}$ and $v^{(j)}$ as self-occupied:

- *case 1*: Given that $(v^{(j)}, w^{(k)})$ is matched between networks $(G^{(j)}, G^{(k)})$, if users $(u^{(i)}, v^{(j)})$ is paired together between networks $(G^{(i)}, G^{(j)})$, and $u^{(i)}$ and $w^{(k)}$ are either free or self-occupied, then we will add $(w^{(k)}, u^{(i)})$ to the result between networks $(G^{(k)}, G^{(i)})$.
- *case 2*: Given that $(v^{(j)}, w^{(k)})$ is matched between networks $(G^{(j)}, G^{(k)})$, if users $(u^{(i)}, v^{(j)})$ is paired together between networks $(G^{(i)}, G^{(j)})$, but either $u^{(i)}$ or $w^{(k)}$ has been matched with other users when matching networks $(G^{(k)}, G^{(i)})$, then we

will set users $u^{(i)}$ and $v^{(j)}$ to be self-occupied in the results between networks $(G^{(i)}, G^{(j)})$.

Next, we can match networks $G^{(k)}$, $G^{(i)}$ by following very similar strategies. For each user pair $(w^{(k)}, u^{(i)})$ to be matched (excluding the pre-added ones), we check the matching statuses of users $w^{(k)}$ and $u^{(i)}$ in the matching of $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$:

- *case 1*: if $w^{(k)}$ and $u^{(i)}$ are both paired with other users in matching $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$, and their partners are the same user actually, then we will add $(w^{(k)}, u^{(i)})$ into the alignment result of networks $(G^{(k)}, G^{(i)})$;
- *case 2*: if $w^{(k)}$ and $u^{(i)}$ are both paired with other users in matching $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$, but their partners are different users, then we will set $w^{(k)}$ and $u^{(i)}$ as free/self-occupied and continue the matching process of networks $(G^{(k)}, G^{(i)})$;
- *case 3*: if one user (e.g., $w^{(k)}$) is matched with one user (e.g., $v^{(j)}$) but the other one (i.e., $u^{(i)}$) is set as self-occupied in matching $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$, then we check the status of $v^{(j)}$ in matching $(G^{(j)}, G^{(k)})$. If $v^{(j)}$ is paired with another user, then we will set $w^{(k)}$ and $u^{(i)}$ as free/self-occupied and continue the matching process of networks $(G^{(k)}, G^{(i)})$;
- *case 4*: if $v^{(j)}$ is also set as self-occupied in matching networks $(G^{(j)}, G^{(k)})$, then we will add pair $(v^{(j)}, w^{(k)})$ into the matching result of networks $(G^{(j)}, G^{(k)})$ and add pair $(w^{(k)}, u^{(i)})$ into the alignment result of networks $(G^{(k)}, G^{(i)})$.

Finally, we can achieve the matching results among networks $G^{(i)}$, $G^{(j)}$ and $G^{(k)}$ respectively.

4 Experiments

To examine the effectiveness of UMA in addressing the M-NASA problem, extensive experiments on real-world multiple partially aligned social networks will be done in this section. Next, we will introduce the dataset used in the experiments in Sect. 4.1 and give brief descriptions about the experiment settings in Sect. 4.2. Experiment results and detailed analysis will be given in Sects. 4.3 and 4.4.

4.1 Dataset Description

Nowadays, Question-and-Answer (Q&A) websites are becoming a new platform for people to share knowledge, where individuals can conveniently post their questions online and get first-hand replies very quickly. A large number of Q&A sites have

sprung out overnight, e.g., Stack Overflow,[1] Super User,[2] Programmers,[3] Quora.[4] Stack Overflow, Super User and Programmers are all Q&A sites constructed for exchanging knowledge about computer science and share large number of common users, which are used as the partially aligned networks $G^{(i)}$, $G^{(j)}$ and $G^{(k)}$ respectively in the experiments.

We crawled the multiple partially aligned Q&A networks during November 2014–January 2015 and the complete information of 10,000 users in Stack Overflow, Super User and Programmers Q&A sites respectively. The anchor links (i.e., the ground truth) between pairs of these Q&A networks are obtained by crawling their homepages in these sites respectively, where users' IDs in all these networks they participate in are listed. For example, at site,[5] we can have access to all the Q&A sites IDs that Jon Skeet owns, which can be used to extract the ground truth anchor links across networks. Among these 3 networks, the number of shared anchor users (1) between Stack Overflow and Super User is 3,677, (2) between Stack Overflow and Programmers is 2,626, (3) between Super User and Programmers is 1,953. Users in Q&A sites can answer questions which are of their interests. Considering that users don't have social links in these Q&A sites, we will create social connections among users if they have every answered the same question in the past. Answering common questions in Q&A sites denotes that they may share common interests as well as common expertise in certain areas.

4.2 Experiment Settings

In the experiments, anchor links between users across networks are used for validation only and are not involved in building models. Considering that the network alignment method introduced in this paper is based on the social link information only, isolated users with no social connections in each network are sampled and removed. Based on the social links among users, we infer the optimal transitional matrices between pairs of networks by minimizing the *friendship inconsistency* as well as the alignment transitivity penalty. Alternative updating method is used to solve the joint objective function, where the transitional matrices are initialized with method introduced in [14]. All users in each network are partitioned into 10 bins according to their social degrees, where initial anchor links are assumed to exist between users belonging to the corresponding bins between pairs of networks, e.g., users in bin 1 of Stack Overflow and those in bin 1 of Programmers. The initial values of entries corresponding to these anchor links in transitional matrices are calculated with the *relative degree distance*

[1] http://stackoverflow.com.

[2] http://superuser.com.

[3] http://programmers.stackexchange.com.

[4] http://www.quora.com.

[5] http://stackexchange.com/users/11683/jon-skeet?tab=accounts.

based on their social degrees, e.g., $rdd(u_l^{(i)}, u_m^{(j)}) = \left(1 + \frac{|deg(u_l^{(i)}) - deg(u_m^{(j)})|}{(deg(u_l^{(i)}) + deg(u_m^{(j)}))/2}\right)^{-1}$, where $deg(u)$ denotes the social degree of user u in the networks. Based on the inferred transitional matrices, anchor links with the highest scores but can meet the *one-to-one* constraint and *transitivity law* are selected with the method introduced in Sect. 3.4, which can output both the confidence scores and their inferred labels.

Comparison Methods: Considering that social networks studied in this paper (1) contain only social link information, and (2) no known anchor links exist between networks, therefore, neither inter-network user resolution method MOBIUS [30] built with various user attribute information nor supervised network alignment method MNA [13] can be applied to address the M-NASA problem. To show the advantages of UMA, we compare UMA with many other baseline methods, including both state-of-art network alignment methods as well as extended traditional methods, which are all unsupervised network alignment methods based on the link information only. All the comparison methods used in the experiments are listed as follows.

- *Unsupervised Multi-network Alignment*: Method UMA introduced in this paper can align multiple partially networks concurrently, which include two steps: (1) transitive network alignment, and (2) transitive network matching. Anchor links inferred by UMA can maintain both *one-to-one* constraint and *transitivity property*.
- *Integrated Network Alignment* (INA): To show that transitive network matching can improve the alignment results, we introduce another method named INA, which is identical to the first step of UMA but without the matching step. Anchor links inferred by INA cannot maintain the *one-to-one* constraint nor *transitivity law* property.
- *Pairwise Network Alignment*: BIG-ALIGN is a state-of-art unsupervised network alignment method proposed in [14] for aligning pairwise networks. When applied to the multiple-network case, BIG-ALIGN can only align networks pair by pair. What's more, the output of BIG-ALIGN cannot maintain the *one-to-one* constraint nor *transitivity property* of anchor links. We also use BIG-ALIGN as a baseline method to show the advantages of the multiple-network alignment framework UMA introduced in this paper.
- *Pairwise Alignment + Pairwise Matching*: We also extend BIG-ALIGN [14] and introduce another baseline method BIG-ALIGN-PM, which can further prune the redundant non-existing anchor links with pairwise network stable matching proposed in [13] to guarantee the inferred anchor links can meet the *one-to-one* constraint.
- *Relative Degree Distance (RDD) based Alignment*: The transitional matrix initialization method RDD [14] is compared as another baseline methods, which calculate the confidence scores of potential anchor links with the degree information of users.
- *Relative PageRank based Alignment*: Traditional PageRank method is mainly proposed for calculating the correlation rank scores of a webpage to the given query. In addition, we also extend the traditional PageRank method and propose a new

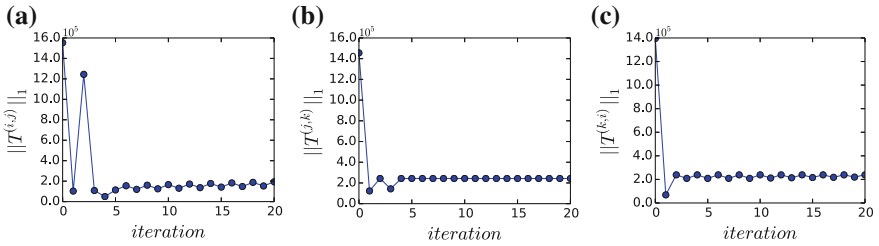


Fig. 4 L_1 norm of transitional matrices at each iteration. **a** Matrix $\mathbf{T}^{(i,j)}$. **b** Matrix $\mathbf{T}^{(j,k)}$. **c** Matrix $\mathbf{T}^{(k,i)}$

method RPR to infer potential anchor links. For a potential anchor link $(u_l^{(i)}, u_m^{(j)})$, RPR calculates the reciprocal of the relative pagerank scores between $u_l^{(i)}, u_m^{(j)}$ as its existence confidence, i.e., $|pagerank(u_l^{(i)}) - pagerank(u_m^{(j)})|^{-1}$.

Evaluation Metrics:
To evaluate the performance of different comparison methods, various commonly used evaluation metrics are applied. All these comparison methods (in INA, the selected anchor links are assigned with scores 1, while those not selected are assigned with scores 0) can output confidence scores of potential anchor links, which are evaluated by metrics AUC and Precision@100.

4.3 Convergence Analysis

To solve the objective function in Sect. 3.3, alternative updating method is applied to infer the optimal transitional matrices across networks. To demonstrate that the matrix updating equation can converge within a limited iterations, we calculate the L_1 norms (i.e., the sum of all entries' absolute value) of transitional matrices $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$ and $\mathbf{T}^{(k,i)}$ at each iteration, which are available in Fig. 4. As shown in the plots, after a few iterations (about 5 iterations), the L_1 norm of these transitional matrices will converge quickly with minor fluctuations around certain values, which demonstrates that the derived equation updating can converge very well in updating the transitional matrices.

4.4 Experiment Results

The experiment results of all these comparison methods are available in Fig. 5, where performance of all these comparison methods in Fig. 5 are evaluated by AUC and Precision@100 respectively.

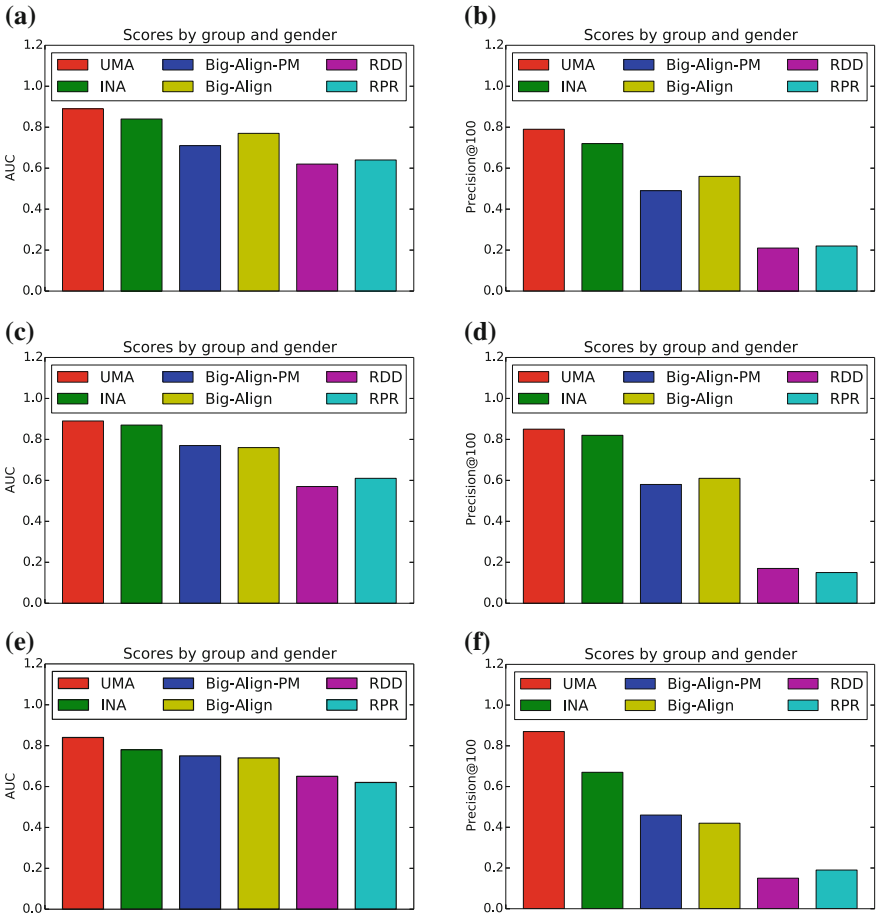


Fig. 5 Performance comparison of different methods evaluated by AUC and Precision@100. **a** AUC ($G^{(i)}$, $G^{(J)}$). **b** Precision @ 100 ($G^{(i)}$, $G^{(J)}$). **c** AUC ($G^{(j)}$, $G^{(k)}$). **d** Precision @ 100 ($G^{(j)}$, $G^{(k)}$). **e** AUC ($G^{(k)}$, $G^{(i)}$). **f** Precision @ 100 ($G^{(k)}$, $G^{(i)}$)

In Fig. 5, we show the alignment results achieved by all the 6 comparison methods between network pairs ($G^{(i)}$, $G^{(j)}$), ($G^{(j)}$, $G^{(k)}$) and ($G^{(k)}$, $G^{(i)}$). As shown in the plots, UMA performs much better than all the other comparison methods with great advantages in predicting the anchor links between all these networks pairs. For instance, in Fig. 5a, the AUC obtained by UMA is 0.89, which is about 4 % larger than INA and over 13 % larger than the other comparison methods; in Fig. 5f, the Precision@100 achieved by UMA is 0.87, which is over 25 % higher than that of INA, almost the double of that gained by BIG-ALIGN and BIG-ALIGN-PM, and even 4–5 times of that obtained by RDD and RPR.

By comparing UMA and INA, method UMA consisting of transitive integrated network alignment and transitive network matching performs better, which demonstrates the effectiveness of the transitive network matching step in pruning redundant non-existing anchor links.

Compared with the isolated pairwise network alignment method BIG-ALIGN, the fact that INA achieves better performance justifies that aligning multiple networks simultaneously by incorporating the alignment transitivity penalty into the objective function can identify better anchor links than pairwise isolated network alignment.

By comparing BIG-ALIGN-PM and BIG-ALIGN, the pairwise network matching step can help improve the prediction results of anchor links between networks $(G^{(k)}, G^{(i)})$ but has no positive effects (even has negative effects) on the anchor links between other network pairs, e.g., network pairs $(G^{(i)}, G^{(j)})$ and $(G^{(j)}, G^{(k)})$. However, the effective of the transitive network matching method applied in UMA has been proved in the comparison of UMA and INA. It may show that transitive network matching exploiting the transitivity law performs much better than the pairwise network matching method.

For completeness, we also compare UMA with extensions of traditional methods RDD and RPR and the advantages of UMA over these methods are very obvious.

5 Related Works

Graph alignment is an important research problem in graph studies [6] and dozens of papers have been published on this topic in the past decades. Depending on specific disciplines, the studied graphs can be social networks in data mining [13] protein-protein interaction (PPI) networks and gene regulatory networks in bioinformatics [11, 17, 23, 24], chemical compound in chemistry [26], data schemas in data warehouse [19], ontology in web semantics [7], graph matching in combinatorial mathematics [18], as well as graphs in computer vision and pattern recognition [3, 5].

In bioinformatics, the network alignment problem aims at predicting the best mapping between two biological networks based on the similarity of the molecules and their interaction patterns. By studying the cross-species variations of biological networks, network alignment problem can be applied to predict conserved functional modules [21] and infer the functions of proteins [20]. Graemlin [9] conducts pairwise network alignment by maximizing an objective function based on a set of learned parameters. Some works have been done on aligning multiple network in bioinformatics. IsoRank proposed in [25] can align multiple networks greedily based on the pairwise node similarity scores calculated with spectral graph theory. IsoRankN [17] further extends IsoRank by exploiting a spectral clustering scheme.

In recent years, with rapid development of online social networks, researchers' attention starts to shift to the alignment of social networks. A comprehensive survey about recent works on heterogeneous social networks, including the recent network alignment works, is available in [22]. Enlightened by the homogeneous network

alignment method in [28], Koutra et al. [14] propose to align two bipartite graphs with a fast alignment algorithm. Zafarani et al. [30] propose to match users across social networks based on various node attributes, e.g., username, typing patterns and language patterns etc. Kong et al. formulate the heterogeneous social network alignment problem as an anchor link prediction problem. A two-step supervised method MNA is proposed in [13] to infer potential anchor links across networks with heterogeneous information in the networks. However, social networks in the real world are mostly partially aligned actually and lots of users are not anchor users. Zhang et al. have proposed the partial network alignment methods based on supervised learning setting and PU learning setting in [32, 33] respectively. Existing social network alignment paper mostly focus on aligning two social networks, Zhang et al. [35] introduce a multiple network concurrent alignment framework to align multiple social networks simultaneously. Besides the common users shared by different social networks, many other categories of information entities, e.g., movies, geo-locations, and products, can also be shared by different movie-related networks, location based social networks, and e-commerce sites respectively. Zhang et al. are the first to introduce the partial co-alignment of social network, and propose a sophisticated network co-alignment framework in [36].

6 Conclusion

In this paper, we have studied the *multiple anonymized social network alignment* (M-NASA) problem to infer the anchor links across multiple anonymized online social networks simultaneously. An effective two-step multiple network alignment framework UMA has been proposed to address the M-NASA problem. The anchor links to be inferred follow both *transitivity law* and *one-to-one* property, under the constraint of which, UMA matches multiple anonymized networks by minimizing the *friendship inconsistency* and selects anchor links which can lead to the maximum confidence scores across multiple anonymized social networks based on the generic stable matching method. In this paper, we take 3 Q&A networks as an example to introduce both the method and conduct the experiments. In our future works, we will generalize the proposed model to multiple networks of diverse categories.

Acknowledgments This work is supported in part by NSF through grants III-1526499, CNS-1115234, and OISE-1129076, Google Research Award, and the Pinnacle Lab at Singapore Management University.

Appendix: New Objective Function

Based on the above relaxations used in Sect. 3.3, the new objective function can be represented as

$$
\begin{aligned}
&\bar{\mathbf{T}}^{(i,j)}, \bar{\mathbf{T}}^{(j,k)}, \bar{\mathbf{T}}^{(k,i)} \\
&= \arg\min_{\mathbf{T}^{(i,j)},\mathbf{T}^{(j,k)},\mathbf{T}^{(k,i)}} \left\| (\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)} - \mathbf{S}^{(j)} \right\|_F^2 \\
&+ \left\| (\mathbf{T}^{(j,k)})^{\top}\mathbf{S}^{(j)}\mathbf{T}^{(j,k)} - \mathbf{S}^{(k)} \right\|_F^2 + \left\| (\mathbf{T}^{(k,i)})^{\top}\mathbf{S}^{(k)}\mathbf{T}^{(k,i)} - \mathbf{S}^{(i)} \right\|_F^2 \\
&+ \alpha \cdot \left\| (\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)} - \mathbf{T}^{(k,i)}\mathbf{S}^{(i)}(\mathbf{T}^{(k,i)})^{\top} \right\|_F^2 \\
&+ \beta \cdot \left\| \mathbf{T}^{(i,j)} \right\|_0 + \gamma \cdot \left\| \mathbf{T}^{(j,k)} \right\|_0 + \theta \cdot \left\| \mathbf{T}^{(k,i)} \right\|_0 \\
&s.t.\ \mathbf{0}^{|\mathcal{U}^{(i)}|\times|\mathcal{U}^{(j)}|} \preccurlyeq \mathbf{T}^{(i,j)} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(i)}|\times|\mathcal{U}^{(j)}|}, \\
&\quad \mathbf{0}^{|\mathcal{U}^{(j)}|\times|\mathcal{U}^{(k)}|} \preccurlyeq \mathbf{T}^{(j,k)} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(j)}|\times|\mathcal{U}^{(k)}|}, \\
&\quad \mathbf{0}^{|\mathcal{U}^{(k)}|\times|\mathcal{U}^{(i)}|} \preccurlyeq \mathbf{T}^{(k,i)} \preccurlyeq \mathbf{1}^{|\mathcal{U}^{(k)}|\times|\mathcal{U}^{(i)}|}.
\end{aligned}
$$

The Lagrangian function of the objective function can be represented as

$$
\begin{aligned}
&\mathcal{L}(\mathbf{T}^{(i,j)}, \mathbf{T}^{(j,k)}, \mathbf{T}^{(k,i)}, \beta, \gamma, \theta) = \left\| (\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)} - \mathbf{S}^{(j)} \right\|_F^2 \\
&+ \left\| (\mathbf{T}^{(j,k)})^{\top}\mathbf{S}^{(j)}\mathbf{T}^{(j,k)} - \mathbf{S}^{(k)} \right\|_F^2 + \left\| (\mathbf{T}^{(k,i)})^{\top}\mathbf{S}^{(k)}\mathbf{T}^{(k,i)} - \mathbf{S}^{(i)} \right\|_F^2 \\
&+ \alpha \cdot \left\| (\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)} - \mathbf{T}^{(k,i)}\mathbf{S}^{(i)}(\mathbf{T}^{(k,i)})^{\top} \right\|_F^2 \\
&+ \beta \cdot \left\| \mathbf{T}^{(i,j)} \right\|_0 + \gamma \cdot \left\| \mathbf{T}^{(j,k)} \right\|_0 + \theta \cdot \left\| \mathbf{T}^{(k,i)} \right\|_0 .
\end{aligned}
$$

The partial derivatives of function $\mathcal{L}$ with regard to $\mathbf{T}^{(i,j)}$, $\mathbf{T}^{(j,k)}$, and $\mathbf{T}^{(k,i)}$ will be:

$$
\begin{aligned}
(1)&\frac{\partial \mathcal{L}\left(\mathbf{T}^{(i,j)}, \mathbf{T}^{(j,k)}, \mathbf{T}^{(k,i)}, \beta, \gamma, \theta\right)}{\partial \mathbf{T}^{(i,j)}} \\
&= 2 \cdot \mathbf{S}^{(i)}\mathbf{T}^{(i,j)}(\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)} \\
&+ 2 \cdot (\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)} \\
&+ 2\alpha \cdot \mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top} \\
&+ 2\alpha \cdot (\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top} \\
&- 2 \cdot \mathbf{S}^{(i)}\mathbf{T}^{(i,j)}(\mathbf{S}^{(j)})^{\top} - 2 \cdot (\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{S}^{(j)} \\
&- 2\alpha \cdot (\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}\mathbf{S}^{(i)}(\mathbf{T}^{(k,i)})^{\top}(\mathbf{T}^{(j,k)})^{\top} \\
&- 2\alpha \cdot \mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top}(\mathbf{T}^{(k,i)})^{\top}(\mathbf{T}^{(j,k)})^{\top} - \beta \cdot \mathbf{1}\mathbf{1}^{\top}. \\
(2)&\frac{\partial \mathcal{L}\left(\mathbf{T}^{(i,j)}, \mathbf{T}^{(j,k)}, \mathbf{T}^{(k,i)}, \beta, \gamma, \theta\right)}{\partial \mathbf{T}^{(j,k)}} \\
&= 2 \cdot \mathbf{S}^{(j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}(\mathbf{S}^{(j)})^{\top}\mathbf{T}^{(j,k)} \\
&+ 2 \cdot (\mathbf{S}^{(j)})^{\top}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}\mathbf{S}^{(j)}\mathbf{T}^{(j,k)} \\
&+ 2\alpha \cdot (\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)} \\
&+ 2\alpha \cdot (\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}(\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)} \\
&- 2 \cdot \mathbf{S}^{(j)}\mathbf{T}^{(j,k)}(\mathbf{S}^{(k)})^{\top} - 2 \cdot (\mathbf{S}^{(j)})^{\top}\mathbf{T}^{(j,k)}\mathbf{S}^{(k)}
\end{aligned}
$$

$$
\begin{aligned}
&- 2\alpha \cdot (\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}\mathbf{S}^{(i)}(\mathbf{T}^{(k,i)})^{\top} \\
&- 2\alpha \cdot (\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top}(\mathbf{T}^{(k,i)})^{\top} - \gamma \cdot \mathbf{1}\mathbf{1}^{\top}.
\end{aligned}
$$

$$
\begin{aligned}
(3)\ &\frac{\partial \mathcal{L}\left(\mathbf{T}^{(i,j)}, \mathbf{T}^{(j,k)}, \mathbf{T}^{(k,i)}, \beta, \gamma, \theta\right)}{\partial \mathbf{T}^{(k,i)}} \\
&= 2 \cdot \mathbf{S}^{(k)}\mathbf{T}^{(k,i)}(\mathbf{T}^{(k,i)})^{\top}(\mathbf{S}^{(k)})^{\top}\mathbf{T}^{(k,i)} \\
&+ 2 \cdot (\mathbf{S}^{(k)})^{\top}\mathbf{T}^{(k,i)}(\mathbf{T}^{(k,i)})^{\top}\mathbf{S}^{(k)}\mathbf{T}^{(k,i)} \\
&+ 2\alpha\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top}(\mathbf{T}^{(k,i)})^{\top}\mathbf{T}^{(k,i)}\mathbf{S}^{(i)} \\
&+ 2\alpha\mathbf{T}^{(k,i)}\mathbf{S}^{(i)}(\mathbf{T}^{(k,i)})^{\top}\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top} \\
&- 2 \cdot \mathbf{S}^{(k)}\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top} - 2 \cdot (\mathbf{S}^{(k)})^{\top}\mathbf{T}^{(k,i)}\mathbf{S}^{(i)} \\
&- 2\alpha \cdot (\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}(\mathbf{S}^{(i)})^{\top}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}\mathbf{S}^{(i)} \\
&- 2\alpha \cdot (\mathbf{T}^{(j,k)})^{\top}(\mathbf{T}^{(i,j)})^{\top}\mathbf{S}^{(i)}\mathbf{T}^{(i,j)}\mathbf{T}^{(j,k)}\mathbf{T}^{(k,i)}(\mathbf{S}^{(i)})^{\top} - \theta \cdot \mathbf{1}\mathbf{1}^{\top}.
\end{aligned}
$$

References

1. Avriel, M.: Nonlinear Programming: Analysis and Methods. Prentice-Hall, Englewood Cliffs (1976)
2. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In: WWW (2007)
3. Bayati, M., Gerritsen, M., Gleich, D., Saberi, A., Wang, Y.: Algorithms for large, sparse network alignment problems. In: ICDM (2009)
4. Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. TKDD (2007)
5. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. IJPRAI (2004)
6. Deo, N.: Graph Theory with Applications to Engineering and Computer Science. Prentice Hall Series in Automatic Computation. Prentice-Hall Inc. (1974)
7. Doan, A., Madhavan, J., Domingos, P., Halevy, A.: Ontology matching: a machine learning approach. In: Handbook on Ontologies (2004)
8. Dubins, L., Freedman, D.: Machiavelli and the gale-shapley algorithm. Am. Math. Mon. (1981)
9. Flannick, J., Novak, A., Srinivasan, B., McAdams, H., Batzoglou, S.: Graemlin: general and robust alignment of multiple large interaction networks. Genome Res. (2006)
10. Jin, S., Zhang, J., Yu, P., Yang, S., Li, A.: Synergistic partitioning in multiple large scale social networks. In: IEEE BigData (2014)
11. Kalaev, M., Bafna, V., Sharan, R.: Fast and accurate alignment of multiple protein networks. In: RECOMB (2008)
12. Khan, A., Gleich, D., Pothen, A., Halappanavar, M.: A multithreaded algorithm for network alignment via approximate matching. In: SC (2012)
13. Kong, X., Zhang, J., Yu, P.: Inferring anchor links across multiple heterogeneous social networks. In: CIKM (2013)
14. Koutra, D., Tong, H., Lubensky, D.: Big-align: fast bipartite graph alignment. In: ICDM (2013)
15. Kunen, K.: Set Theory. Elsevier Science Publishers (1980)
16. Lee, J., Han, W., Kasperovics, R., Lee, J.: An in-depth comparison of subgraph isomorphism algorithms in graph databases. VLDB (2012)

17. Liao, C., Lu, K., Baym, M., Singh, R., Berger, B.: Isorankn: spectral methods for global alignment of multiple protein networks. Bioinformatics (2009)
18. Manne, F., Halappanavar, M.: New effective multithreaded matching algorithms. In: IPDP (2014)
19. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In: ICDE (2002)
20. Park, D., Singh, R., Baym, M., Liao, C., Berger, B.: Isobase: a database of functionally related proteins across ppi networks. Nucleic Acids Res. (2011)
21. Sharan, R., Suthram, S., Kelley, R., Kuhn, T., McCuine, S., Uetz, P., Sittler, T., Karp, R., Ideker, T.: Conserved patterns of protein interaction in multiple species (2005)
22. Shi, C., Li, Y., Zhang, J., Sun, Y., Yu, P.: A survey of heterogeneous information network analysis. CoRR (2015). arXiv:1511.04854
23. Shih, Y., Parthasarathy, S.: Scalable global alignment for multiple biological networks. Bioinformatics (2012)
24. Singh, R., Xu, J., Berger, B.: Pairwise global alignment of protein interaction networks by matching neighborhood topology. In: RECOMB (2007)
25. Singh, R., Xu, J., Berger, B.: Global alignment of multiple protein interaction networks with application to functional orthology detection. In: Proceedings of the National Academy of Sciences (2008)
26. Smalter, A., Huan, J., Lushington, G.: Gpm: a graph pattern matching kernel with diffusion for chemical compound classification. In: IEEE BIBE (2008)
27. Tsikerdekis, M., Zeadally, S.: Multiple account identity deception detection in social media using nonverbal behavior. IEEE TIFS (2014)
28. Umeyama, S.: An eigendecomposition approach to weighted graph matching problems. IEEE TPAMI (1988)
29. Wipf, D., Rao, B.: L0-norm minimization for basis selection. In: NIPS (2005)
30. Zafarani, R., Liu, H.: Connecting users across social media sites: a behavioral-modeling approach. In: KDD (2013)
31. Zhan, Q., Zhang, J., Wang, S., Yu, P., Xie, J.: Influence maximization across partially aligned heterogenous social networks. In: PAKDD (2015)
32. Zhang, J., Shao, W., Wang, S., Kong, X., Yu, P.: Partial network alignment with anchor meta path and truncated generalized stable matching. In: IRI (2015)
33. Zhang, J., Yu, P.: Integrated anchor and social link predictions across social networks. In: IJCAI (2015)
34. Zhang, J., Yu, P.: Mcd: mutual clustering across multiple heterogeneous networks. In: IEEE BigData Congress (2015)
35. Zhang, J., Yu, P., Multiple anonymized social networks alignment. In: ICDM (2015)
36. Zhang, J., Yu, P., Pct: partial co-alignment of social networks. In: WWW (2016)
37. Zhang, J., Yu, P., Zhou, Z.: Meta-path based multi-network collective link prediction. In: KDD (2014)

An Accurate Multi-sensor Multi-target Localization Method for Cooperating Vehicles

Sepideh Afkhami Goli, Behrouz H. Far and Abraham O. Fapojuwo

Abstract Accurate and reliable vehicle localization is a key component of Intelligent Transportation System (ITS) applications. Personalized travel related services and recommendation systems like collision avoidance rely principally on the accurate and reliable knowledge of vehicles' positioning. In this paper we propose a cooperative multi-sensor multi-vehicle localization method with high accuracy for terrestrial consumer vehicles. Two streams of real-time data are assumed available. One in the form of GPS coordinates of nearby vehicles received from a vehicle-to-vehicle (V2V) network and the other in the form of inter-vehicle distance measurements from a range sensor. In real-world situations, these heterogeneous sources of information are noisy and could be unavailable during short intervals. To overcome the effect of noise, measurements from two sources are fused together to estimate the number and motion model parameters of the vehicles. The problem is formulated in the context of Bayesian framework and vehicle locations as well as their velocities are estimated via a Sequential Monte-Carlo Probability Hypothesis Density (SMC-PHD) filter. To test the effectiveness of the proposed approach, a simulated scenario based on a grid portion of downtown Calgary is designed. Traffic intensity values match real-world reported data for the selected test location. Results of the simulation indicate that the proposed method provides reliable estimation of motion model parameters that predict the future location of vehicles.

Keywords Data fusion · Intelligent Transportation System (ITS) · PHD filter · Random finite set · VANET · Vehicle localization

S.A. Goli (✉) · B.H. Far (✉) · A.O. Fapojuwo
Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada
e-mail: Sepideh.afkhamigoli@ucalgary.ca

B.H. Far
e-mail: Far@ucalgary.ca

A.O. Fapojuwo
e-mail: Fapojuwo@ucalgary.ca

T. Bouabana-Tebibel and S.H. Rubin (eds.), *Theoretical Information Reuse and Integration*, Advances in Intelligent Systems and Computing 446,
DOI 10.1007/978-3-319-31311-5_9

1 Introduction

Localization is at the core of personalized travel services. For example, collision avoidance, emergency braking assistance, lane navigation and location based recommendation systems rely heavily on accurate and reliable knowledge of vehicles' positions [1].

Global Positioning System (GPS) is being used in vehicle localization and navigation. This system provides absolute position of vehicles. However, GPS signals are subject to various sources of noise. The signals can be degraded or blocked when a vehicle is traveling through tunnels or in urban areas close to buildings or vegetation [2]. It is shown in [3, 4] that using a radio based ranging technique such as Received Signal Strength Indicator (RSSI) together with GPS can reduce the localization error and improve accuracy. Data fusion from GPS and non-GPS sources can potentially improve position estimation by exploiting different sources of information.

Recent introduction of Dedicated Short Range Communication (DSRC) devices, based on IEEE 802.11p standard for wireless access in vehicular environments [5], has evoked considerable interest within the research communities and automotive industries [6]. With vehicle to vehicle (V2V) communication through DSRC, cooperative architectures have become an attractive approach for solving the localization problem. The main goal of cooperative localization is to exploit different sources of information coming from different vehicles within a short range area, in order to enhance positioning system efficiency while keeping the computing cost at a reasonable level. In other words, vehicles share their location and environment information via V2V communication with the other close-by vehicles to improve their own global perception.

In this paper, the goal is to enable terrestrial vehicles to accurately find their own and other nearby vehicles' locations. Each vehicle incorporates its own GPS data, other vehicles GPS data received via V2V communication and inter-vehicle distance measurements in the localization process. Due to availability of multiple sources of real time data for multiple vehicle localization, the problem is defined as a multi-sensor multi-target information fusion and tracking.

Most of the prominent approaches for cooperative localization are based on Extended Kalman filtering [1, 4], Bayesian methods [7] and maximum likelihood estimation (MLE) [8]. These methods improve the location information, but they are mostly based on single-target tracking algorithms which do not inherently support multi-target environments. More specifically, the multi-target problem is broken down to a set of single-target problems and then solved via a data association algorithm that adds to the complexity and unreliability of the method. This problem worsens in the presence of noisy and missing data.

The problem of true multi-source multi-vehicle localization has been studied in [9, 10]. The authors have merged the data from proprioceptive and exteroceptive sensors into one stream of data and practically considered only one sensor as data

types were the same. However, fusion of heterogeneous data sources has not been addressed in the literature, to the authors' best knowledge.

The main contribution of this paper is the proposal and implementation of a novel cooperative multi-vehicle localization method incorporating multiple sources of heterogeneous data with a potential to achieve high tracking accuracy. In the proposed method, the objective is to improve location estimation and prediction of all nearby vehicles, in a cooperative manner. The method is decentralized and is run in each vehicle separately. At every time step, coordinates of each vehicle (obtained from the GPS receiver) and those of its neighbors (obtained through V2V communication), as well as the distance to each vehicle in range is provided to the method. These streams of real-time data could be noisy, mixed with clutter and missed for short periods of time.

Here, the number of vehicles and also the number of observations are unknown and time variant. Therefore, the challenge is tracking the joint detection and estimation of the unknown and time-varying number of vehicles and their dynamic state given multiple sequences of observation sets. Probability Hypothesis Density (PHD) filter for tracking unknown, time-variant multiple targets in the presence of false alarms, miss-detections, and clutter based on the formal theory of Random Finite Sets (RFSs) has already been proposed [11, 12]. Compared to more traditional approaches, such as the Kalman Filtering based methods, the main advantage of using set-based PHD Filters is the ability to skip the data association step, as this alleviates some of the computational burden of assigning estimations to targets.

There are two general implementations for the PHD filter in the literature [13, 14]. Gaussian Mixture PHD filter (GM-PHD) assumes linear and Gaussian model for sensors and objects. On the other hand, Sequential Monte Carlo PHD (SMC-PHD) [13] is another implementation in which particles are distributed to represent the density functions by sampling which follows more relaxed assumptions. Considering the highly dynamic nature of urban traffic, SMC-PHD was chosen as it is a better fit to this problem.

In our method the state of individual vehicles is treated as a set-valued state, and the collection of individual sensor measurements is treated as a set-valued observation and modeled as RFS. Based on the Multi-source Multi-target Information Fusion Framework, a recursive Bayesian based filter is derived to jointly estimate target states and their number. The SMC-PHD filter [13] is implemented to estimate vehicle states. Simulation results show that the method better estimates the location of vehicles than raw GPS data.

The rest of this paper is as follows: Sect. 2 provides the system modeling assumptions and problem formulation. Section 3 describes in detail the proposed localization method, followed by presentation of simulation results and discussion in Sect. 4. Finally, Sect. 5 concludes the paper.

2 System Model

2.1 Assumptions

1. Vehicles are equipped with GPS receivers. Absolute position of a vehicle is provided by its own GPS receiver.
2. The distance between a vehicle and some other nearby vehicles is provided by a range-based sensor such as a radio-based ranging sensor. In our experiments, the detection range for range sensor is set to 100 m which is the typical range with modern sensors [15].
3. Sensor data can be noisy and temporarily unavailable. The noise is assumed to be additive white Gaussian noise (AWGN).
4. Vehicles are equipped with DSRC devices and able to share their absolute position via V2V broadcast communication. A group of vehicles in communication range of each other can be considered as a cluster.
5. Each vehicle's observation is transmitted and shared over the network, hence data association (i.e. which observation belongs to which vehicle) is not modeled explicitly.

2.2 Problem Definition and Formulation

Cooperative multi sensor vehicle localization can be defined as follows. Consider a cluster of N vehicles labeled from $1, 2 \ldots N$ at unknown locations at time t. The goal is to have each vehicle find its own precise location and also the locations of the other $N-1$ vehicles in its communication and sensing range at time t. Due to the dynamic state of road conditions (i.e. road layout and pattern, number of lanes, dual/single carriage way, etc.), the number of vehicles in range of the considered vehicle (i.e. number of targets of the considered vehicle) may change. For the same reason the number of received observations via V2V communication and range sensor may also change. In order to formulate the problem, vehicle dynamics, state and measurements are defined as follows.

2.2.1 Vehicle Dynamics Model

For simplicity and fast convergence of the proposed method, a constant velocity motion model is assumed to describe vehicles' dynamics. This assumption seems to be unrealistic in urban transportation because it implies zero acceleration but is somewhat reasonable for a highway traffic scenario. However, it is sufficient for our purpose because only the velocity range of vehicles is considered.

According to the constant velocity motion equations, the change in the position of a vehicle at each direction can be calculated using the following formula:

$$\Delta \mathrm{p} = \mathrm{v}\Delta t \tag{1}$$

where p is the coordinate vector in the form of [x y] and Δp is the traveled distance during time period Δt with constant velocity v in the form of $[\mathrm{v_x}\ \mathrm{v}_y]$; v and p are collectively referred to as the kinematic model parameters.

The kinematic model parameters can be assumed to be partially known as interval values, that is $\mathrm{v} \in \left[\bar{\mathrm{v}}, \underline{\mathrm{v}}\right]$ and $p \in \left[\bar{p}, \underline{p}\right]$, where $\bar{\mathrm{v}}$ and $\underline{\mathrm{v}}$ are the minimum and maximum velocity respectively, and $\bar{p}$ and $\underline{p}$ are respectively the minimum and maximum Cartesian coordinates for a vehicle. These intervals can be determined based on environmental variables and the application requirements.

2.2.2 State Model

The motion Eq. (1) is given in continuous time. For the purpose of computer implementation, we require a discrete-time approximation of this model. The state vector for each target area is defined as

$$x = \left[\mathrm{x\ y\ v_x\ v_y}\right]^T \tag{2}$$

where T denotes the matrix transpose. It includes x and y coordinates and also the velocity components $\mathrm{v_x}$ and $\mathrm{v_y}$ at each direction.

For analytic tractability, the state is assumed to follow a Markovian process on state space $\mathcal{X} \in \mathbb{R}^4$. Let N_k be the number of vehicles detected at time index k where $k = t_k/\tau$ is the discrete time index for small discretization interval $\tau > 0$. The targets in a multi-target scenario at time k are represented as a Random Finite Set (RFS) [16] of vectors:

$$X_k = \left\{x_{k,1}, \ldots, x_{k,N_k}\right\} \in \mathcal{F}(\mathcal{X}) \tag{3}$$

where $\mathcal{F}(\mathcal{X})$ denotes the set of all finite subsets of $\mathcal{X}$.

2.2.3 Measurement Model

At a specific time step we assume:

$$z_k = l_k + \omega_k \tag{4}$$

where z_k is a single sensor measurement. In Eq. (4) l_k can be the position (x, y) of a vehicle or the Euclidean distance $(d, 0)$ to a vehicle in the observation field,

based on the measuring sensor. The noise term ω_k is added to simulate random nature of measurement (sensor measurement noise). From assumption 3, ω_k is a white Gaussian noise.

Similarly, the observations are represented as an RFS. Consider each sensor r with overlapping coverage providing measurements of the targets in $\mathcal{X}$. By taking values in the observation space $\mathcal{Z} \in \mathbb{R}^2$ we can write:

$$Z_k^r = \left\{ z_{k,1}^r, \ldots, z_{k,M_k}^r \right\} \in \mathcal{F}(\mathcal{Z}) \tag{5}$$

where M_k is the total number of observations at time index k and $\mathcal{F}(\mathcal{Z})$ is the collection of all finite subsets of $\mathcal{Z}$ ($Z_k = \bigcup_r Z_k^r$). In this study, the collection of the received coordinates is considered as Z_k^1 and the set of the distances is considered as Z_k^2.

Modelling set-valued states and set-valued observations as RFSs allows the problem of dynamically estimating multiple targets in the presence of clutter [12, 16]. The problem of vehicle localization is then formulated as a multi-target tracking problem, recast as a filtering problem with multi-target state space $\mathcal{F}(\mathcal{X})$ and observation space $\mathcal{F}(\mathcal{Z})$.

3 Proposed Localization Method

The proposed localization method estimates precise location and velocity of multiple vehicles moving in the vicinity of each other. Each vehicle's future states can be predicted by projecting the estimated velocity and position in time. The proposed method therefore can potentially serve as the nucleus of an ITS solution such as a Lane Navigation or a Collision Avoidance system Fig. 1 represents the input, output and application of the proposed technique. The details about the input data are given in Sect. 2.1.

The proposed method recursively predicts the position of each vehicle using the kinematic motion model and then updates the corresponding states using distance and GPS measurements. The novelty of this approach is the ability to use multiple sources of heterogeneous and noisy observations to jointly estimate the number and states of vehicles based on a true multi-source multi-target mathematical foundation. This is in contrast with previous methods like [1, 7] where a single filter has to be run for each vehicle separately. Another benefit of the proposed approach in comparison to others (such as the Kalman based methods) is that sharing additional information like covariance matrices is not needed. GPS data is the only shared piece of information among vehicles.

The number of vehicles and also the number of input observations are unknown. Therefore, the collection of motion model parameters of individual vehicles is treated as a set-valued state, and the collection of individual sensor measurements is treated as a set-valued observation and modeled as RFSs (Eqs. (3) and (5)). Based on the Multi-source Multi-target Information Fusion Framework [10, 11], a recursive Bayesian

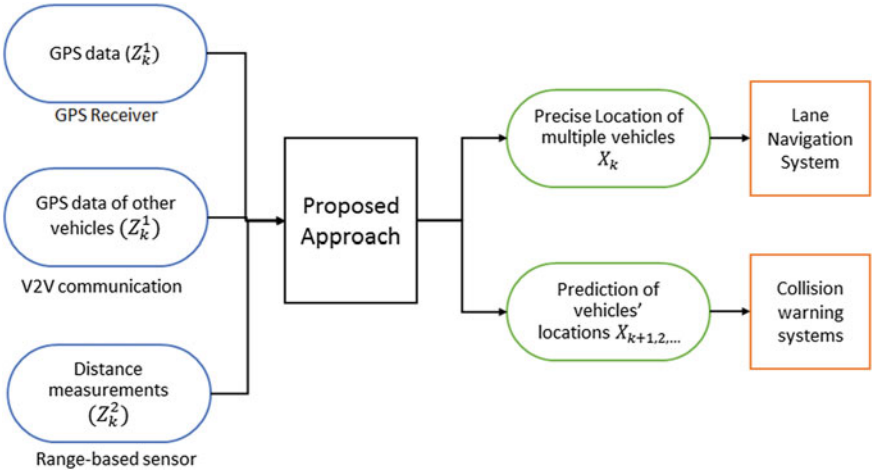


Fig. 1 Block diagram of the proposed localization system together with inputs, outputs and example applications

based filter is derived to jointly estimate target states and their number. Since the resulting equations are not computationally tractable, a Sequential Monte Carlo Probability Hypothesis Density (SMC-PHD) filter [12] is implemented to estimate vehicle states.

3.1 Optimal Bayesian Solution

The RFS, which models the multi-target state, is the union of state vectors that survived from the previous time step, those which have been spawned by existing targets and those which appear spontaneously, and is given by [12]:

$$X_k = \left(\bigcup_{x \in X_{k-1}} S_{k|k-1}(x)\right) \cup \left(\bigcup_{x \in X_{k-1}} B_{k|k-1}(x)\right) \cup \Gamma_k \tag{6}$$

In (6), the first term is the RFS of vehicle states at discrete time index k given the previous states X_{k-1} (survived targets). $S_{k|k-1}(x)$ can take on either $\{x_k\}$ when the target survives, or $\emptyset$ when it leaves the range. In the second term, $B_{k|k-1}(x)$ represents the RFS of new possible states at time step $k-1$. The last term, Γ_k, accounts for spontaneous vehicle emergence in range.

The multi-target measurement at time step k is modeled by the RFS:

$$Z_k = K_k \bigcup \left(\bigcup_{x \in X_k} \Theta_k(x)\right) \tag{7}$$

where $\Theta_k(x)$ is the RFS of measurements from multi-target state X_k and K_k is the RFS of measurements due to false reports (clutter). For a given target x_k the term $\Theta_k(x_k)$ can take on either $\{z_k\}$ when the target is detected with probability $p_{D,k}(x_k)$, or $\emptyset$ when the target is missed with probability $1 - p_{D,k}(x_k)$.

The formal Bayesian solution is given in the form of the multi-target posterior density $p(X_k|Z_{1:k})$. The posterior density is then used to predict and estimate the state of vehicles at the next time step, using the dynamic model equations.

The multi-target Bayes filter propagates the multi-target posterior density $p_k(\cdot|Z_k)$ conditioned on the sets Z_k of observations up to time step k, with the following recursions [12],

Step 1: Prediction

$$p_{k|k-1}(X_k|Z_{k-1}) = \int f_{k|k-1}(X_k|X_{k-1})p_{k-1}(X_{k-1}|Z_{k-1})\mu_s(dx) \tag{8}$$

Step2: Update

$$p_k(X_k|Z_k) = \frac{g_k(Z_k|X_k)p_{k|k-1}(X_k|Z_{k-1})}{\int g_k(Z_k|X_k)p_{k|k-1}(X_k|Z_{k-1})\mu_s(dx)} \tag{9}$$

where the dynamic model is governed by the multi-target transition density $f_{k|k-1}X_k|X)$ and multi-target likelihood $g_k(Z_k|X_k)$ and μ_s is an appropriate reference measure on $\mathcal{F}(\mathcal{X})$ [17]. The randomness in the multi-target evolution and observation is captured in $f_{k|k-1}(.|.)$ and $g_k(.|.)$, respectively [14]. The function $g_k(Z_k|X_k)$ is the joint multi-target likelihood function, or global density, of observing the set of measurements (observations) Z, given the set of target states X, which is the total probability density of association between the measurements in Z and the parameters in X. The parameters for this density are the set of observations Z, the unknown set of targets X, observation noise, probability of detection P_D, false alarm, P_{FA}, and clutter models [18].

The state-transition in discrete time can be expressed as:

$$S_{k|k-1}(x) = \begin{bmatrix} x_k\,[1] + \tau x_k\,[3] \\ x_k\,[2] + \tau x_k\,[4] \\ x_k[3] \\ x_k[4] \end{bmatrix} + w_k. \tag{10}$$

In (10) $x_k\,[i]$ denotes the ith component of vector x_k. The process noise w in (10) is assumed to be a zero-mean white Gaussian noise. Components of w_k, except the first two, are set to zero based on the assumption that the velocity v is constant during the motion. In the implementation $\tau = 1$ is used, for convenience.

3.2 Estimation with the SMC-PHD Filter

The recursion (8) and (9) involves infinite integrals on the space $\mathcal{F}(\mathcal{X})$, which are computationally intractable [14]. PHD filter [19] is an approximation that propagates the first-order statistical moment, or intensity, of the RFS of states in time [12]. This approximation was developed to alleviate the computational intractability in the multi-target Bayes filter. The PHD filter operates on the single-target state space and avoids the combinatorial problem that arises from data association [14].

For an RFS X on $\mathcal{X}$ with probability distribution P, its first order moment is a non-negative function v on $\mathcal{X}$, called the intensity or Probability Hypothesis Density (PHD) [12] in the tracking literature, such that for each region $S \subseteq \mathcal{X}$

$$\int |X \cap S| \, P(dX) = \int_S v(x)dx. \tag{11}$$

Hence, the total mass

$$\widehat{N} = \int v(x)dx \tag{12}$$

gives the expected number of elements of X that are in S which, in other words, is the number of vehicles. The local maxima of the intensity v are points in $\mathcal{X}$ with the highest local concentration of expected number of elements, and therefore can be used to generate estimates for elements of X [14].

Let v_k and $v_{k|k-1}$ denote the respective intensities associated with the multi-target posterior density p_k and the multi-target predicted density $p_{k|k-1}$ in the recursions (8) and (9). It can be shown that the posterior intensity can be propagated in time via the PHD recursion [19]:

Step 1: Prediction

$$\begin{aligned} v_{k|k-1}(x) = &\int p_{s,k}(\varsigma) f_{k|k-1}(x|\varsigma) v_{k|k-1}(\varsigma) d\varsigma \\ &+ \int \beta_{k|k-1}(x|\varsigma) v_{k-1}(\varsigma) d\varsigma + \gamma_k(x), \end{aligned} \tag{13}$$

Step 2: Update

$$v_k(x) = G_k^1(x) \ldots G_k^R(x).v_{k|k-1}(x) \tag{14}$$

in which G_k^r for each sensor r is given by:

$$\begin{aligned} G_k^r(x) = 1 - p_{D_r,k}(x) + \\ \sum_{z \in Z_{r,k}} \frac{p_{D_r,k}(x) g_{r,k}(z|x)}{\mathcal{K}_{r,k}(z) + \int p_{r,k}(\xi) g_{r,k}(z|\xi) v_{k|k-1}(\xi) d\xi} \end{aligned} \tag{15}$$

where $\gamma_k(x)$ is the intensity of the birth RFS Γ_k, $\beta_{k|k-1}(.|\varsigma)$ is the intensity of the spawn RFS $B_{k|k-1}$, and $p_{s,k}(\varsigma)$ is the probability of survival based on the previous

state set ς. For each sensor r, $p_{D_r,k}$ is the probability of detection, $\mathcal{K}_{r,k}$ is the intensity of clutter RFS K_k, and $g_{r,k}(z|x)$ is the likelihood function. As mentioned in assumptions 1 and 2, two different sensors $R = 2$ are considered with the likelihood functions corresponding to the observation type (distance or x- and y-coordinates).

Since the PHD recursion involves infinite integrals that have no closed form solutions in general, an approximate solution is developed. Generic Sequential Monte Carlo techniques have been proposed to propagate the posterior intensity in time (see [20] and the references therein). In this approach, state estimates are extracted from the particles representing the posterior intensity using clustering techniques such as K-means or expectation maximization [21, 22]. The K-means approach is adopted for this study due to its wide spread adoption and straight forward implementation.

The SMC-PHD filter approximates the intensity of posterior PDF $p_k(X_k|Z_k)$ by a weighted random sample. Given a sequence of measurement sets $Z_{1:k}$, the approximation at time step $k > 0$ is given as follows.

In the initialization stage, particles are distributed across the state space randomly. The initial intensity function v_0 is given by [17]

$$v_0(x) = \sum_{i=1}^{L_0} w_0^{(i)} \delta(x - x_0^{(i)}). \tag{16}$$

Here $\delta(.)$ is the Dirac delta function and $\left\{x_k^i, i = 1, \ldots L_k\right\}$ are support points or particles with associated weights $\left\{w_k^i, i = 1, \ldots L_k\right\}$ constructing a random measure $\left\{x_k^{(i)}, w_k^{(i)}\right\}_{i=1}^{L_k}$, where L_k is the particle count for step k. The weights are selected based on the principle of importance sampling and are normalized such that $\sum_i w_k^i = 1$.

The particles are propagated in the prediction stage using the dynamic model Eqs. (13) and (14). Particles are also added to allow for new vehicles representing the term Γ_k. The predicted intensity function $v_{k|k-1}$ at time step k is [17]

$$v_{k|k-1}(x) = \sum_{i=1}^{L_{k-1}+J_k} \widetilde{w}_{k|k-1}^{(i)} \delta(x - \tilde{x}_k^{(i)}) \tag{17}$$

where

$$\tilde{x}_k^{(i)} \sim \begin{cases} q_k(.|x_{k-1}^{(i)}, Z_k), & i = 1, \ldots, L_{k-1} \\ p_k(.|Z_k), & i = L_{k-1} + 1, \ldots, L_{k-1} + J_k \end{cases} \tag{18}$$

and

$$\widetilde{w}_{k|k-1}^{(i)} = \begin{cases} \frac{\phi_{k|k-1}(\tilde{x}_k^{(i)}, x_k^{(i)})}{q_k(\tilde{x}_k^{(i)}|x_{k-1}^{(i)}, Z_k)} w_{k-1}^{(i)}, & i = 1, \ldots, L_{k-1} \\ \frac{1}{J_k} \frac{\gamma_k(\tilde{x}_k^{(i)})}{p_k(\tilde{x}_k^{(i)}|Z_k)}. & i = L_{k-1} + 1, \ldots, L_{k-1} + J_k \end{cases} \tag{19}$$

In Eqs. (18) and (19), q_k and p_k are two important sampling proposal densities by which the samples are obtained. Here the L_{k-1} particles are predicted forward by

the kernel $\phi_{k|k-1}$ that captures the dynamic model equations, and an additional J_k particles are drawn to detect new vehicles.

The prediction steps are carried out until a set of measurements of reported cases Z_k becomes available, at time index k. When the measurements are received, weights are calculated for the particles based on their likelihoods, which are determined by the statistical distance of the particles to the set of observations. Given that the particle representation of the predicted intensity function is available at time step k, the updated intensity function v_k is given by [17].

$$v_k(x) = \sum_{i=1}^{L_{k-1}+J_k} \widetilde{w}_k^{(i)} \delta(x - \tilde{x}_k^{(i)}). \tag{20}$$

In this formulation the $\widetilde{w}_k^{(i)}$ s are given by

$$\widetilde{w}_k^{(i)} = \left[(1 - p_{D,k}(\tilde{X}_k^{(i)})) + \sum_{z \in Z_k} \frac{p_{D,k}(\tilde{x}_k^{(i)}) g_k(z|\tilde{x}_k^{(i)})}{\mathcal{K}_k(z) + C_k(z)} \right] \widetilde{w}_{k|k-1}^{(i)} \tag{21}$$

where

$$C_k(z) = \sum_{j=1}^{L_{k-1}+J_k} p_{D,k}(\tilde{x}_k^{(j)}) g_k(z|\tilde{x}_k^{(j)}) \widetilde{w}_{k|k-1}^{(j)}. \tag{22}$$

The preceding analysis therefore provides a discrete weighted approximation of the true posterior $p_k(X_k|Z_k)$. The sum of the weights gives the estimated number of vehicles.

Particles are then resampled from the weighted particle set to give an unweighted representation of the PHD. The role of resampling is to eliminate (in a probabilistic manner) the particles with low importance weights and to clone the samples with high importance weights. This is carried out by sampling with replacement, with the probability of sampling each $x_k^{(i)}$ equal to the normalized importance weight $w_k^{(i)}$. The result is mapping of the random measure $\left\{x_k^{(i)}, w_k^{(i)}\right\}_{i=1}^{L_k}$ into a new random measure with uniform weights.

The complete position estimation process of the proposed approach is as follows:

1. In the initialization step, current states of vehicles are generated randomly.
2. Each vehicle predicts the next state of the vehicles (X_k) based on previous states X_{k-1} and the motion model of vehicles, as in (17).
3. Each vehicle receives its own position via GPS receiver and shares the position with other vehicles in the communication range. It also measures the inter-vehicle distance to all vehicles in the sensor range.
4. Each vehicle updates vehicles states using (19) based on the new observations received in steps 3.
5. Steps 2 to 4 are preformed repeatedly until the vehicle stops.

The proposed solution has a practical computational complexity for real time applications because it is independent of the number of targets in range. The only dependency that should be considered is the number of particles which has to be proportional to the maximum expected number of targets which leads to a linear complexity. The reader can refer to [13] for more information.

4 Simulation Results and Discussion

4.1 Test Scenario

Aimsun 8, as a microscopic traffic simulator [23], was used as a test environment for the proposed method. Aimsun continuously models each vehicle's behavior in the network according to the driver's behavior such as car flowing and lane changing. The traffic simulation provides the collective behavior of all vehicle units within the range of network geometries. Figure 2 depicts the general layout of the simulation study area in Aimsun.

The simulation scenario has been constructed based on a grid in downtown Calgary, Canada bounded by 4th Ave SE to the north, 6th Ave SW to the south, Center Street to the west and Macleod Trail to the east (Fig. 3). The study area has a posted speed limit of 50 km/hr and includes nine traffic signals. Vehicles have variable speed

Fig. 2 Simulated city layout grid in Aimsun

Fig. 3 Satellite view of the Calgary city grid as the basis of the simulation (Image from Google earth)

and accelerations. (Aimsun allows specification of the maximum, minimum and average values for speed and vehicle acceleration). The traffic intensity values were set based on the data provided by the City of Calgary [24].

4.2 Simulated Sensor Observations

The vehicle trajectories produced by Aimsun were selected as ground truth for our method. We assumed, each vehicle is equipped with a GPS receiver and a radio-based ranging sensor. Results are generated assuming the method runs on one specific vehicle referred to, henceforth, as the reference vehicle. The ranging sensor on the reference vehicle provides the method with distance measurements of the other vehicles in vicinity. GPS data of the other nearby vehicles are provided through V2V communication. Each vehicle broadcasts its own coordinates received by GPS receiver at the beginning of a simulation step. The simulation step is 0.8 s in our experiments as Aimsun provides location data per 0.8 s. The step time can be either increased or decreased. Although, decreasing this time can improve the accuracy it will increase the processing and communication overhead.

MATLAB software is used to simulate GPS and range sensor observations and also the communication channel. In order to generate GPS data and inter-vehicle distances, we used the imported vehicle trajectories from Aimsun. For the experiment, we have assumed that the position estimated by GPS differs from the true position according to a Gaussian distributed random variable with standard deviation of 6 m, which is consistent with real GPS error levels of 3–10 m [25]. For the range sensor data simulation, the same method as used for the GPS case was applied and the error

Table 1 Sensor simulation parameters

Parameter	GPS	Range sensor
Range		100 m
Standard deviation of noise	6 m	3 m
Probability of detection	0.98	0.98
Clutter rate per step	0.5	0.5

in inter-vehicle distances is modeled by a zero mean Gaussian distribution with a standard deviation of 3 m.

Table 1 summarizes the parameters used for sensor data simulation. In this table, range defines the maximum distance of observations, probability of detection accounts for missed observations and clutter rate defines the average number of false positives within sensor range.

4.3 Communication Channel Simulation

Based on IEEE 802.11p, a vehicle can communicate with the other vehicles in its communication range. In [26], this range is 500 m in highways, however it is more limited in urban areas due to an increase in the number and size of obstacles like buildings or vegetation. In this study, this range is arbitrarily set to 100 m as it does not change the results other than increasing the number of detected targets. The propagation environment for vehicle-to-vehicle communication is modeled by distance-dependent path loss and log-normal shadowing. Hence, the random path loss (in dB) between the receivers at a distance $d > d_0$ from the transmitter is given by

$$PL(d) = PL_0 + 10\gamma log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \tag{23}$$

where PL_0 is the path loss at a reference distance d_0. In our simulation, we set $PL_0 = 62$ dB, $\gamma = 4$ and $\sigma = 6$ dB based on previous work on path loss modeling for vehicle-to-vehicle communication in an urban environment [27]. The variates of the shadowing loss are generated (during a simulation run) by calling the Gaussian distribution with zero mean and standard deviation of $\sigma = 6$ dB.

4.4 Results

Table 2 lists the parameters used for the implemented SMC-PHD filter. These values were selected after a set of trial runs. Depending on the application and data noise or

Table 2 Parameters of SMC-PHD filter

Probability of detection p_D	0.99
Clutter rate per scan $\mathcal{K}$	2
Probability of birth p_B	0.0001
No. of particles for survived targets	50,000
No. of particles for birth targets	10,000
Max no. allowable particles	60,000

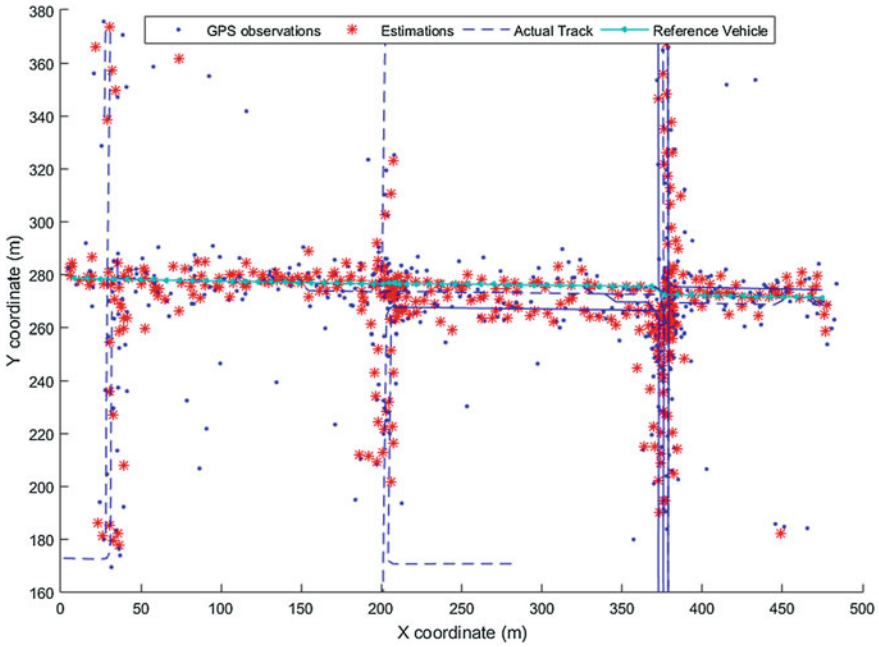


Fig. 4 Scatter plot of vehicle locations

missed sensor data, the values can be fine-tuned. Figure 4 shows a scatter plot of the location estimation method run by the reference vehicle entering the simulation area through 4th Ave from the east. Its true trajectory is depicted by a dashed green line while the dashed blue lines show the ground truth trajectory of vehicles in the range of the reference vehicle. Blue dots represent the GPS observations received by the reference vehicle through communication or directly through its own GPS receiver. Red stars are the estimation location of targets.

In a multi-target tracking problem, the number of targets (vehicles) is unknown and need to be estimated. To study the accuracy of detecting the correct number of vehicles, Fig. 5 shows the cardinality during the course of the simulation. The ground truth cardinality (blue solid line) represents the number of vehicles in the field-of-view of the distance sensor and communication range (including the reference vehicle itself). The dotted red line depicts the average number of detected vehicles at each

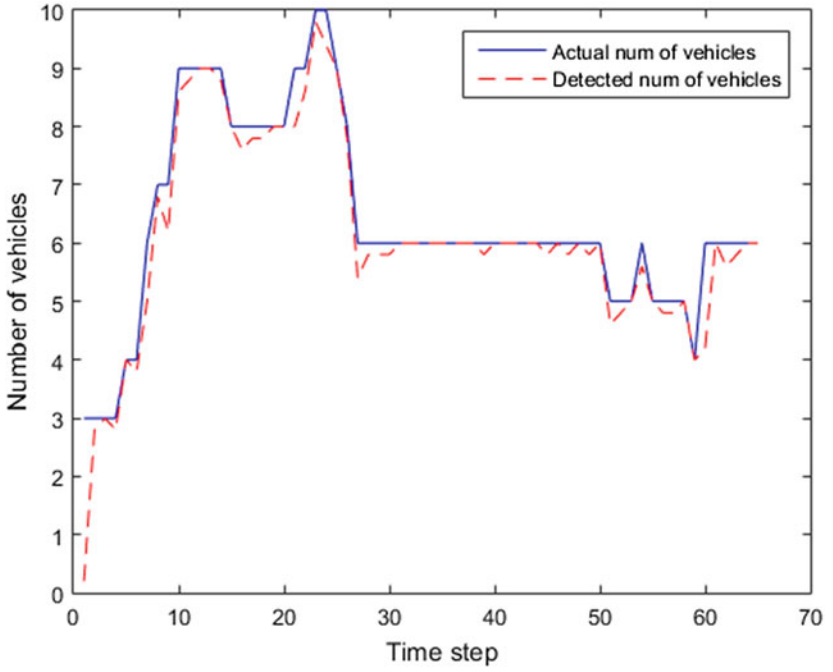


Fig. 5 Cardinality of estimated vehicles at each time step (averaged over 50 runs) versus the ground truth cardinality

time step over 50 Monte Carlo run simulations with the same trajectory data. As shown in Fig. 5, the proposed approach is very accurate in estimating the correct number of vehicles. However the accuracy reduces for short periods of time when the target cardinality changes until the method reaches convergence again.

The accuracy of the proposed method is quantified by the root-mean-square error (RMSE) metric defined as:

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(\mathrm{x}_{est,i} - \mathrm{x}_i)^2 + (\mathrm{y}_{est,i} - \mathrm{y}_i)^2}{N}} \tag{24}$$

where $\mathrm{x}_{est,i}$ and $y_{est,i}$ are the estimated values of vehicle i's coordinate at each time step and N is the number of vehicles. In order to calculate the RMSE metric, a data association algorithm need to recover the correct correspondence between the estimations and the grand truth targets. As mentioned in assumption (5), data association is not needed to take place in the vehicles and it is only done for the sake of evaluation. A simple method was used in which an exhaustive search of all combinations of estimation-target assignments was employed to minimize the total Euclidean distance between estimations and their associated targets.

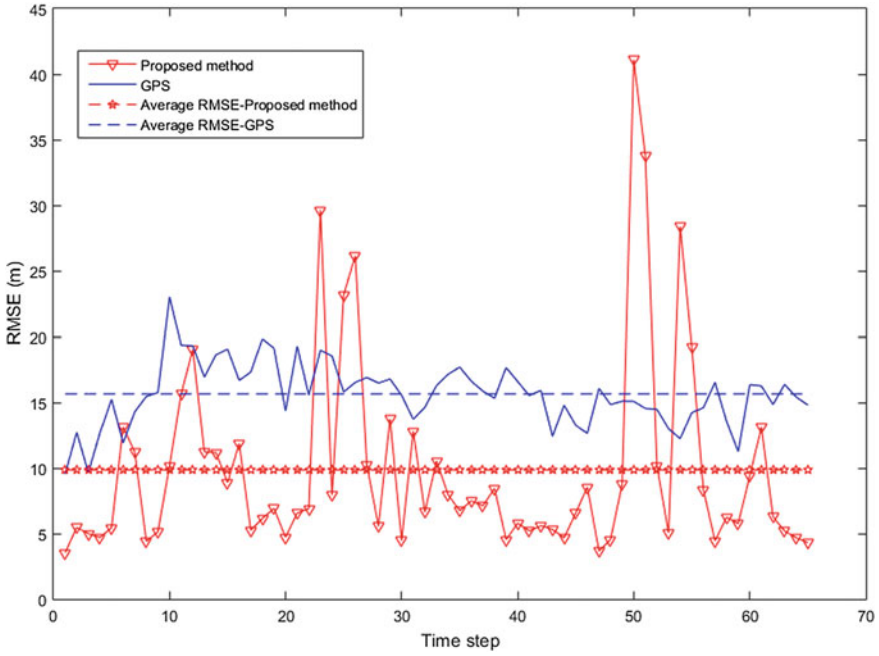


Fig. 6 RMSE of the proposed localization method and received GPS observations averaged over 50 trials

Figure 6 compares the RMSE of the proposed method and received GPS observations. The red line depicts the average value over 50 runs of Monte Carlo simulation with the same trajectory data during the first 70 time steps of the simulation. As shown in Fig. 6, the proposed method has lower RMSE and a few sporadic peaks. These peaks of estimation error happen when a sudden change occurs in the surrounding environment in terms of vehicles movement or cardinality. The method however recovers quickly. Overall, the average localization error is decreased by 30 % compared to that of the GPS data.

In the simulation, not only the position of the vehicles evolves in time, but also the number of target vehicles changes. The varying number of targets is due to the entrance and exit of other vehicles into and from the reference vehicle's range. SMC-PHD filter here involves jointly estimating the number of vehicles and their coordinates from the observations. The RMSE error alone cannot be considered to evaluate a multi-object tracking process since it does not give any sense of the difference between the number of actual targets and the estimated number at the same time. Optimal Sub Pattern Assignment distance (OSPA) [28] can be used to evaluate multi-target tracking methods (referred to as OSPA distance).

OSPA distance objective is to evaluate the differences between the two sets of estimated and true targets' positions. Considering two sets $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, where $m, n \in N = \{0, 1, \ldots\}$, OSPF distance is defined as the

distance between X and Y. The OSPA distance of order $1 \leq p \leq \infty$ and cut-off value c and $m \leq n$ is calculated as [28]:

$$D_{p,c}(X, Y) = \left[\frac{1}{n} \left(\min_{\pi \in \Pi_n} \sum_{i=1}^{m} (d_c(x_i, y_{\pi(i)}))^p + (n - m)\, c^p \right) \right]^{1/p} \tag{25}$$

where vectors $x \in X$ and $y \in Y$ are taking values from the state space, while sets X and Y can be a set of position vectors of all finite subset of vehicles. Here, the cut-off distance, $d_c(x, y)$, is defined as $d_c(x, y) = \min\{c, d(x, y)\}$. We considered $d(x, y)$ to be the Euclidean distance between x and y. Π_n represents the set of permutations of length m with elements taken from {1, 2,…, n}. For the case m > n, the definition is simply $D_{p,c}(X, Y) = D_{p,c}(Y, X)$.

The average values of OSPA distance over 50 runs of the simulation with cut-off distance c = 25 m and order p = 2 are shown in Fig. 7. As seen from the figure, the performance of the proposed method is significantly improved over that of received GPS data. The average OSPA of the proposed method, shown by the dotted red line, is about 30 % less than that of the GPS data, which means that the proposed method performs better both in localization and in detecting the correct number of targets.

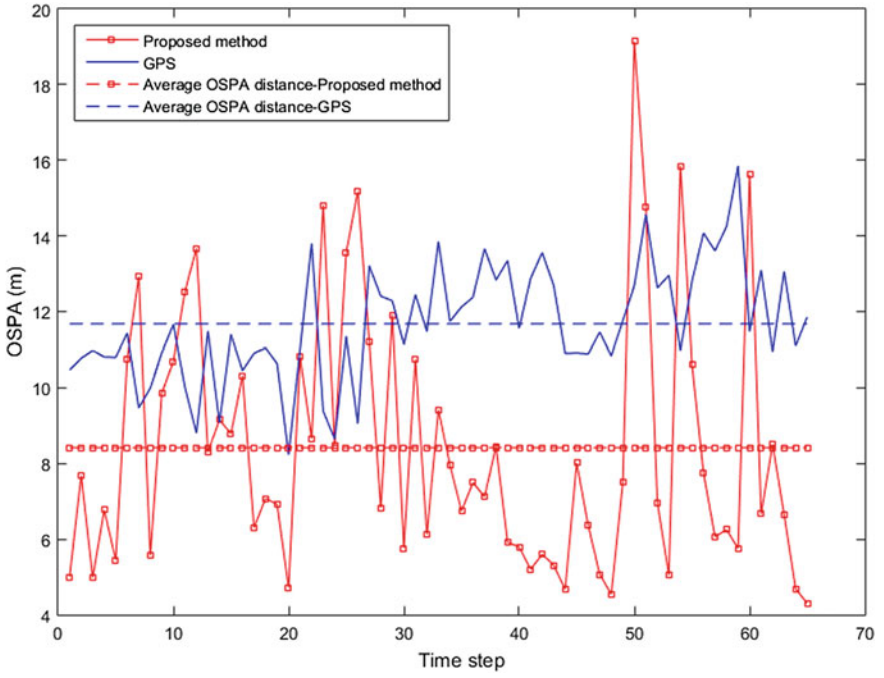


Fig. 7 Performance comparison, using OSPA distance, between the proposed method and GPS observations for cut-off = 25 m and p = 2

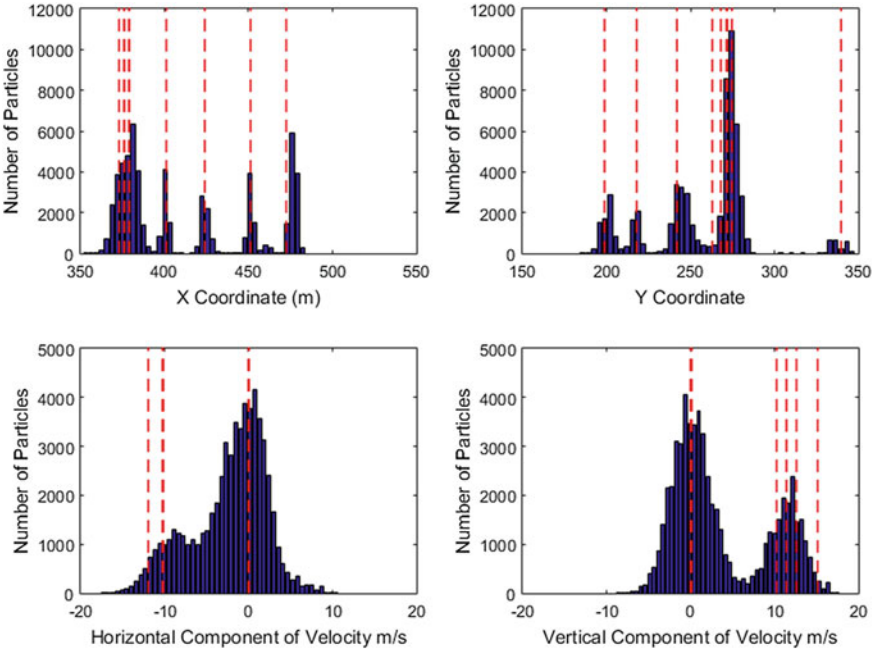


Fig. 8 Estimation results of SMC-PHD filter after 10 simulation steps. The true values of vehicles are shown by *dashed lines* and the *blue bars* represent particles

Figure 8 shows the histograms of the PHD filter estimated values of x coordinate, y coordinate and velocity component for in range vehicles, after 10 simulation time steps. True values are depicted by red dashed lines and the blue bars show the number of particles. This figure reveals that the uncertainty in x and y parameters is low. However, the uncertainty for v components is not as low as that for x and y, indicating that these parameters are difficult to estimate.

5 Conclusion

This research presents a method based on cooperation among neighboring vehicles for estimation and prediction of vehicle locations. The localization problem is formulated as a multi-target multi-source filtering problem and the locations are estimated based on a SMC-PHD filter implementation. The usefulness of the proposed method is tested in an urban scenario, in downtown Calgary, with realistic number of vehicles in roads. Experimental studies show that the suggested method serves its purpose in the presence of noise and a highly dynamic simulated environment. The performance of this method, in terms of the accuracy in estimating the number and position of vehicles is about 30 % higher than that of GPS data received by V2V communication.

Based on this result, it is concluded that the proposed method serves as a viable alternative for terrestrial vehicle localization in ITS applications. Our ongoing research includes implementing the method on a suitable processor for vehicles like the graphical processing unit and then optimizing the processing time. Furthermore, our future plan includes investigating the effect of the proposed technique in collision prediction and avoidance systems as well as other safety related personalized ITS solutions such as real time monitoring and warning system that can monitor and detect pedestrians on blind left turns; blind merge warning; curve speed warning; rollover warning; emergency vehicle traffic signal pre-emption; highway/rail collision warning; intersection collision warning (rear end or vehicle running a red light); vehicle-based road condition warning; wrong way driver warning; stop sign violation warning; traffic signal violation warning and work zone warning.

Acknowledgments This work is partially supported by Alberta Innovative Technology Futures (AITF), Calgary, AB, Canada.

References

1. Karam, N., Chausse, F., Aufrere, R., Chapuis, R.: Collective localization of communicant vehicles applied to collision avoidance. In: 2006 IEEE International IEEE Conference on Intelligent Transportation Systems, pp. 442–449 (2006)
2. Bonsen G.V., Ammann, D., Ammann, M., Favey, E., Flammant, P.: Continuous navigation: combining GPS with sensor-based dead reckoning. GPS World **16**, 47–54 (2005)
3. Parker, R., Valaee, S.: Vehicular node localization using received-signal-strength indicator. IEEE Trans. Veh. Technol. **56**(6), 3371–3380 (2007)
4. Parker, R., Valaee, S.: Cooperative vehicle position estimation. In: 2007 IEEE International Conference on Communications, pp. 5837–5842, Jun 2007
5. IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE)—Networking Services. IEEE Stand. (2007)
6. Zhu, J., Roy, S.: MAC for dedicated short range communications in intelligent transport system. IEEE Commun. Mag. **41**(12), 60–67 (2003)
7. Rohani, M., Gingras, D., Vigneron, V., Gruyer, D., Livic, I.-I.M.: A new decentralized bayesian approach for cooperative vehicle localization based on fusion of GPS and inter-vehicle distance measurements. In: International Conference on Connected Vehicles and Expo (ICCVE) (2013)
8. Wang, Z., Luo, J., Zhang, X., Member, S.: A novel location-penalized maximum likelihood estimator for bearing-only target localization. IEEE Trans. Sig. Process. **60**(12), 6166–6181 (2012)
9. Zhang, F., Buckl, C., Knoll, A.: Multiple vehicle cooperative localization with spatial registration based on a probability hypothesis density filter. Sensors (Basel) **14**(1), 995–1009 (2014)
10. Zhang, F., Stahle, H., Chen, G., Buckl, C., Knoll, A.: Multiple vehicle cooperative localization under random finite set framework. In: IEEE International Conference on Intelligent Robots and Systems, pp. 1405–1411 (2013)
11. Mahler, R.: "Statistics 101" for multisensor, multitarget data fusion. Aerosp. Electron. Syst. Mag. **19**(1), 53–64 (2004)
12. Mahler, R.P.S.: Multitarget bayes filtering via first-order multitarget moments. IEEE Trans. Aerosp. Electron. Syst. **39**(4), 1152–1178 (2003)
13. Vo, B.-N., Singh, S., Doucet, A.: Sequential monte carlo implementation of the PHD filter for multi-target tracking. In: Proceedings of the Sixth International Conference Information Fusion, vol. 2, pp. 792–799 (2003)

14. Vo, B.-N., Ma, W.-K.: The Gaussian Mixture Probability Hypothesis Density Filter. IEEE Trans. Sig. Process. **54**(11), 4091–4104 (2006)
15. Sick Sensor Intelligence: Distance Sensors. http://www.sick.com/us/en-us/home/products/product_portfolio/distance_sensors/Pages/distance_sensors_long_range.aspx. Accessed 10 Nov 2015
16. Mahler, R.P.S.: Random-set approach to data fusion. In: Proceedings of SPIE (1994)
17. Vo, B., Singh, S., Doucet, A.: Sequential Monte Carlo methods for multitarget filtering with random finite sets. IEEE Trans. Aerosp. Electron. Syst. **41**(4), 1224–1245 (2005)
18. Clark, D.: Multiple target tracking with the probability hypothesis density filter. Heriot-Watt University (2006)
19. Mahler, R.P.S.: A theoretical foundation for the Stein-Winter 'probability hypothesis density (PHD)' multitarget tracking approach. In: MSSSSDF (2000)
20. Stordal, A.S.: Sequential Monte Carlo Methods for Bayesian Filtering. University of Bergen, Norway (2008)
21. Sidenbladh, H.: Multi-target particle filtering for the probability hypothesis density. In: 6th International Conference on Information Fusion, pp. 800–806 (2003)
22. Zajic, T., Mahler, R.: Particle-systems implementation of the PHD multitarget-tracking filter. In: Proceedings of SPIE (2003)
23. TSS—Transport Simulation Systems (2013). http://www.aimsun.com/. Accessed 10 Nov 2015
24. The City of Calgary Transportation Planning 2015. http://www.calgary.ca/Transportation/TP/Documents/data/traffic_flow_maps/2014_flowmap_DT.pdf. Accessed 10 Nov 2015
25. U.S. Department of Defense: Global Positioning System Standard Positioning Service (2008)
26. Paikari, E., Tahmasseby, S., Far, B.: A simulation-based benefit analysis of deploying connected vehicles using dedicated short range communication. In: 2014 Proceedings of IEEE Intelligent Vehicles Symposium (IV), pp. 980–985 (2014)
27. Rohde, Schwarz: WLAN 802.11p measurements for vehicle to vehicle (V2V) DSRC. Rohde Schwarz Application Note, pp. 1–25 (2009)
28. Schuhmacher, D., Vo, B.T., Vo, B.N.: A consistent metric for performance evaluation of multi-object filters. IEEE Trans. Sig. Process. **56**(8), 3447–3457 (2008)