Longbing Cao Ana L.C. Bazzan
Vladimir Gorodetsky Pericles A. Mitkas
Gerhard Weiss Philip S. Yu (Eds.)

Agents and Data Mining Interaction

LNAI 5980

6th International Workshop on Agents and Data Mining Interaction, ADMI 2010
Toronto, ON, Canada, May 2010, Revised Selected Papers

Springer

Lecture Notes in Artificial Intelligence 5980

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Longbing Cao Ana L.C. Bazzan
Vladimir Gorodetsky Pericles A. Mitkas
Gerhard Weiss Philip S. Yu (Eds.)

Agents and Data Mining Interaction

6th International Workshop on Agents and Data Mining Interaction, ADMI 2010
Toronto, ON, Canada, May 11, 2010
Revised Selected Papers

Springer

Volume Editors

Longbing Cao
University of Technology, Sydney, Australia
E-mail: LongBing.Cao-1@uts.edu.au

Ana L.C. Bazzan
Universidade Federal do Rio Grande do Sul, Brazil
E-mail: bazzan@inf.ufrgs.br

Vladimir Gorodetsky
St. Petersburg Institute for Informatics and Automation, Russia
E-mail: gor@mail.iias.spb.su

Pericles A. Mitkas
Aristotle University of Thessaloniki, Greece
E-mail: mitkas@eng.auth.gr

Gerhard Weiss
University of Maastricht, The Netherlands
E-mail: gerhard.weiss@maastrichtuniversity.nl

Philip S. Yu
University of Illinois at Chicago, USA
E-mail: psyu@cs.uic.edu

Library of Congress Control Number: 2010932768

CR Subject Classification (1998): I.2, H.3, H.4, H.2.8, F.1, H.5

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-15419-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-15419-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Message from the Workshop Chairs

We are pleased to welcome you to the proceedings of the 2010 International Workshop on Agents and Data Mining Interaction (ADMI 2010), held jointly with AAMAS 2010.

In recent years, agents and data mining interaction (ADMI, or agent mining) has emerged as a very promising research field. Following the success of ADMI 2006 in Hong Kong, ADMI 2007 in San Jose, AIS-ADM 2007 in St. Petersburg, ADMI 2008 in Sydney, ADMI 2009 in Budapest, the ADMI 2010 workshop in Toronto provided a premier forum for sharing research and engineering results, as well as potential challenges and prospects encountered in the coupling between agents and data mining.

The ADMI 2010 workshop encouraged and promoted theoretical and applied research and development, aimed at:

- Exploiting agent-enriched data mining and machine learning, and demonstrating how agent technology can contribute to critical data mining and machine learning problems in theory and practice
- Improving data mining-driven agents and systems, and showing how data mining and machine learning can strengthen agent intelligence and intelligent systems in research and practical applications
- Exploring the integration of agents and data mining toward a super-intelligent system and intelligent information processing
- Identifing challenges and directions for future research and development in agent mining, through the synergy and interaction among relevant fields
- Reporting workable applications and case studies of agent mining

The 15 accepted papers were competitively selected from 37 submissions distributed across 15 countries. ADMI 2010 submissions covered areas from North America, Europe to Asia, indicating the booming of agent mining research globally. The workshop also included an invited talk by a distinguished researcher.

Following ADMI 2009, the accepted ADMI 2010 papers are published as an LNAI proceedings volume by Springer. We appreciate Springer, in particular Alfred Hofmann, for the continuing publication support.

ADMI 2010 was sponsored by the Special Interest Group: Agent-Mining Interaction and Integration (AMII-SIG: www.agentmining.org). We appreciate the guidance of the Steering Committee.

More information about ADMI 2010 is available from the workshop website: http://admi10.agentmining.org/.

Finally, we appreciate the contributions made by all authors, Program Committee members, invited speaker, panelists, and AAMAS 2010 workshop and local organizers.

May 2010

Gerhard Weiss
Philip S. Yu
Longbing Cao
Ana Bazzan
Pericles A. Mitkas
Vladimir Gorodetsky

Organization

General Chair

Gerhard Weiss	University of Maastricht, The Netherlands
Philip S. Yu	University of Illinois at Chicago, USA

Workshop Co-chairs

Longbing Cao	University of Technology Sydney, Australia
Ana Bazzan	Universidade Federal do Rio Grande do Sul, Instituto de Informatica, Brazil
Pericles A. Mitkas	Aristotle University of Thessaloniki, Greece
Vladimir Gorodetsky	Russian Academy of Sciences, Russia

Workshop Organizing Chairs

Yuming Ou	University of Technology Sydney, Australia

Program Committee

Agnieszka Dardzinska	Bialystok Technical University, Poland
Ahmed Hambaba	San Jose State University, USA
Ajith Abraham	Norwegian University of Science and Technology, Norway
Andrea G. B. Tettamanzi	University degli Studi di Milano, Italy
Andreas L. Symeonidis	Aristotle University of Thessaloniki, Greece
Andrzej Skowron	Institute of Decision Process Support, Poland
Boi Faltings	Artificial Intelligence Laboratory, The Swiss Federal Institute of Technology in Lausanne, Switzerland
Daniel Kudenko	University of York, UK
Dionysis Kehagias	ITI-CERTH, Greece
Frans Coenen	University of Liverpool, UK
Ioannis Athanasiadis	Dalle Molle Institute for Artificial Intelligence, Switzerland
Jason Jung	Yeungnam University, Korea
Jiming Liu	Hong Kong Baptist University, China
Joerg Mueller	Technische University Clausthal, German
John Debenham	University of Technology Sydney, Australia
Juan Carlos Cubero	University de Granada, Spain
Karl Tuyls	Maastricht University, The Netherlands
Kazuhiro Kuwabara	Ritsumeikan University, Japan
Ken Kaneiwa	National Institute of Information and Communications Technology, Japan

Luis otavio alvares	Universidade Federal do Rio Grande do Sul, Brazil
Martin Purvis	University of Otago, New Zealand
Matthias Klusch	DFKI, Germany
Mehmet Orgun	Macquarie University, Australia
Michal Pechoucek	Czech Technical University, Czech Republic
Mirsad Hadzikadic	University of North Carolina, Charlotte, USA
Nathan Griffiths	University of Warwick, UK
Ngoc Thanh Nguyen	Wroclaw University of Technology, Poland
Ning Zhong	Maebashi Institute of Technology, Japan
Simeon Simoff	University of Technology Sydney, Australia
Sung-Bae Cho	Yonsei University, Korea
Sviatoslav Braynov	University of Illinois at Springfield, USA
Tapio Elomaa	Tampere University of Technology, Finland
Valerie Camps	University Paul Sabatier, France
Wee Keong Ng	Nanyang Technological University, Singapore
Wen-Ran Zhang	Georgia Southern University, USA
William Cheung	Hong Kong Baptist University, Hong Kong
Yaodong Li	Institution of Automation, Chinese Academy of Sciences, China
Yiyu Yao	University of Regina, Canada
Yves Demazeau	CNRS, France
Zbigniew Ras	University of North Carolina, USA
Yanqing Zhang	Georgia State University, USA
Zili Zhang	Deakin University, Australia

Steering Committee

Longbing Cao	University of Technology Sydney, Australia (Coordinator)
Edmund H. Durfee	University of Michigan, USA
Vladimir Gorodetsky	St. Petersburg Institute for Informatics and Automation, Russia
Hillol Kargupta	University of Maryland Baltimore County, USA
Matthias Klusch	DFKI, Germany
Michael Luck	King's College London, UK
Jiming Liu	Hong Kong Baptist University, China
Pericles A. Mitkas	Aristotle University of Thessaloniki, Greece
Joerg Mueller	Technische Universität Clausthal, Germany
Ngoc Thanh Nguyen	Wroclaw University of Technology, Poland
Carles Sierra	Artificial Intelligence Research Institute of the Spanish Research Council, Spain
Gerhard Weiss	University of Maastricht, The Netherlands
Xindong Wu	University of Vermont, USA
Philip S. Yu	University of Illinois at Chicago, USA
Chengqi Zhang	University of Technology Sydney, Australia

Table of Contents

Part I: Agents for Data Mining

Part II: Data Mining for Agents

Part III: Data Mining in Agents

Part I

Agents for Data Mining

Finding Useful Items and Links in Social and Agent Networks

Sandip Sen

University of Tulsa, USA
sandip-sen@utulsa.edu

Abstract. In this talk, I will overview our recent work addressing the problem of locating useful items and trading partners within large-scale social and agent-organized networks.

In agent-organized networks (AONs), agents have a limited number of collaboration partners at any time, represented by edges in a network of agent nodes. Agents have to learn to estimate the utility of current trading partners and rewire edges, i.e., change partners to improve profitability. We show that an exponentially decaying exploration scheme produces similar utilities with much less rewiring costs compared to a previously proposed random partner selection scheme within a production and exchange economy. We analyze the observed performance and demonstrate the robustness of our proposed rewiring scheme in a more realistic model of the economy that incorporate minimum trade volumes and storage capacities.

We have developed a Social Network-based Item Recommendation (SNIR) system consisting of agents that utilizes the connections of a user in the social network to search for items of interest posted by peers. Our approach enables a focused mining of the huge number of items available in the social network to generate targeted recommendations that can improve the precision of the result returned from a user's query. We have implemented the SNIR agent-based framework in Flickr, a photo-sharing social network, for searching and recommendation of photos by using tag lists as search queries. We discuss the architecture of our system, motivate the searching and referral schemes used, and demonstrate the effectiveness of the SNIR approach by presenting representative results.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, p. 3, 2010.

Integrating Workflow into Agent-Based Distributed Data Mining Systems

Chayapol Moemeng, Xinhua Zhu, and Longbing Cao

Quantum Computing and Intelligent Systems,
Faulty of Engineering and Information Technology,
University of Technology, Sydney
{mchayapol,xhzhu,lbcao}@it.uts.edu.au

Abstract. Agent-based workflow has been proven its potential in overcoming issues in traditional workflow-based systems, such as decentralization, organizational issues, etc. The existing data mining tools provide workflow metaphor for data mining process visualization, audition and monitoring; these are particularly useful for distributed environments. In agent-based distributed data mining (ADDM), agents are an integral part of the system and can seamlessly incorporate with workflows. We describe a mechanism to use workflow in descriptive and executable styles to incorporate between workflow generators and executors. This paper shows that agent-based workflows can improve ADDM interoperability and flexibility, and also demonstrates the concepts and implementation with a supporting the argument, a multi-agent architecture and an agent-based workflow model are demonstrated.

1 Introduction

Workflow is an automation of a business process during which document, information or tasks are passed from one participant to another for action, according to a set of procedural rules [1]. The workflow metaphor is widely accepted and used in various process modeling tasks, such as service, production, security, data mining, etc. With regard to knowledge discovery in databases (data mining, for short), most existing data mining tools, such as SAS Enterprise Miner , PASW , Weka[2], RapidMiner[3], KNIME[4], provide workflow metaphor for process visualization, audition, and monitoring. Process visualization is incredibly useful particularly in distributed environments in which complex process configuration can be quickly and correctly interpreted. In agent-based distributed data mining (ADDM), coordination between distributed nodes is a complex task, the use of workflow metaphor is beneficial to both users and systems mutually. Besides improving user friendliness, the system can use the workflow in coordinating the process. In this work, we integrate the use of workflow with the process coordination in ADDM.

In fact, a workflow is only a document describing the process execution. In order to cope with dynamic changes in resource levels and task availability, the workflow engine is the one who actually deals with those difficulties. Major difficulties relate to (i) dynamic changes of the system environment as well as (ii) workflow execution exceptions [5]. Firstly, system resources, such as CPU, disk storage, security access, etc., may

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 4–15, 2010.

change over time. The plan to execute a workflow must be prepared prior to the execution, because execution plan depends on the current situation hence derived differently over time. Secondly, while a workflow is being enacted, failures during an execution can possibly stall the entire process. Thus, workflow engine must handle various kinds of exceptions and recover so that the execution can be carried on completely.

Attempts to overcome workflow's difficulties with agents have shown very promising results [6–11] by integrating agents into workflows, as known as agent-based workflow (AWF). AWF is a style of workflow management that allows workflow to be fully provisioned by agents in which agents provide solutions to the previously addressed difficulties of traditional workflow management. Regarding dynamic changes of resources, agents autonomously perceive the current situation of the environment and develop proper settings appropriately to the changes of the environment, for example, selecting a network communication channel from current available choices at the run time. Concerning exception handling, a generic approach is to provide a control and monitoring mechanism for the workflow management. However, in a distributed environment, the centralized control and monitoring mechanism fails to take immediate action when the network communication is not available, while some exceptions can be solved by simply re-running the workflow activity again [10]. Agents' pro-activeness is suitable when trivial decision can be made immediately without consulting the central controller. As a result, decentralized workflows have been investigated further and found potential extensibility when integrate with multi-agent system. The critical features of AWF systems are autonomous collaboration and communication [7]. Agents must cooperate, communicate and negotiate with other agents for coordinating the workflow and executing sub tasks of the workflow on various locations. In this work, we argue that AWF can improve ADDM's flexibility and interoperability. Consider an ADDM system, agents are already an integral part and are assigned to various responsibilities, e.g., performing data mining task on remote sites, etc. Therefore, adding an extra responsibility or agents to the existing framework does not cost as much as re-engineering the whole system structure. In conclusion, the integration of workflow into ADDM can enhance ADDM in terms of the following aspects.

First, the flexibility of the system is determined by how well the system can adapt to new tasks. Variations in data mining process can complicate the system, thus workflows are used to represent these variations in a form that can be interpreted by agents. Agents' planning evaluates a descriptive workflow into an executable one which describes specific details, e.g., resource location, related agent containers, etc.

Second, the interoperability concerns how agents work with each other. Given a data mining workflow, an original workflow can be evaluated into different versions depending on the current system situation, e.g., system resources, available agents, etc. Therefore, the best execution plan can be pre-determined prior to the actual execution for the best interoperability. The rest of paper is organized as follows: section 2 shows a review of related work and discusses their limitations, section 3 describes how AWF integrates with ADDM, section 4 shows the experiment system for the integration, section 5 describes the implementation and an example, section 6 is the evaluation, and finally the paper concludes in section 7.

2 Related Work

Existing data mining tools, e.g. SAS Enterprise Miner, PASW, Weka [2], RapidMiner [3], KNIME [4], use workflow to visualize data mining process. SAS, PASW, RapidMiner and KNIME also provide enterprise version of their products distribute its work over multiple processors and to perform simultaneous calculations on multiple models in the client. Their client/server architecture includes workflows that also involve distributed data analysis processes. RapidMiner Enterprise Analytics Server and KNIME Enterprise Server aims to provide data analysis service thru the Internet which users can access the service via web browsers. All computations occur at the enterprise server. Another KNIME enterprise product, Cluster Execution, aims to increase computing power using computer clusters. These systems assume that data must be visible to the entire system; therefore one node cannot hide its data from another.

With regards to AWF, many researchers have been exploring the use of agents to improve process integration, interoperability, reusability and adaptability in distributed environment. [12] analyses benefits of integrating workflow and agent technology for business process management. [10] presents the monitoring and controlling aspects of distributed agent-based workflow systems. [13] develops a multi-agent-based workflow system that supports distributed process models using web services. [8] presents an agent-based approach for coordinating product design workflows that supports distributed dynamic process management. [7] argues that in order for agents to communicate effectively and enhance interoperability, they need to have mutually understandable and standard semantic constructs, and they propose an RDF-based schema to define communication semantics among agents. Unfortunately, no research has applied to the DDM scenario.

Agents have proven its excellence in facilitating particular needs in DDM. A few classic systems, discussed in [14], have demonstrated the use of agents in distributed learning environment, e.g. JAM, PADMA, BODHI, etc. Agents offer desirable properties suitable for complex systems, for example, agent sociability enables DDM to choose appropriate platforms to execute the actions; that is to allow agents to perform locally on the data site without exposing the data. Unfortunately, none of these systems support workflows. Despite both AWF and ADDM researches have been prototyped; still no research presented the integration of both technologies.

3 AWF for ADDM

Data mining process is often customized and varied by the requirement of each particular data mining task. The process can be described in a modeling language, such as PMML [4], as a mean to store and retrieve for later use with other compatible tasks. The main idea of this work is to merge data mining model and workflow into one document that can be used by ADDM.

Initially, the workflow is presented in a descriptive style document which contains relative information about data mining process, such as resource name, relative path of data, etc. The descriptive workflow is then evaluated to an executable style document which contains specific information, such as physical resource location, absolute path

of data, specific agent container. In this way, the responsible unit that creates descriptive workflows (i.e. external entity) does not need to comprehend the knowledge about the system; instead the execution unit which resides internally in the system is responsible for the workflow evaluation. The workflow evaluation is carried out prior to the execution which results in different versions depending on the current situation of the system. Thus the system configuration is protected from external entity.

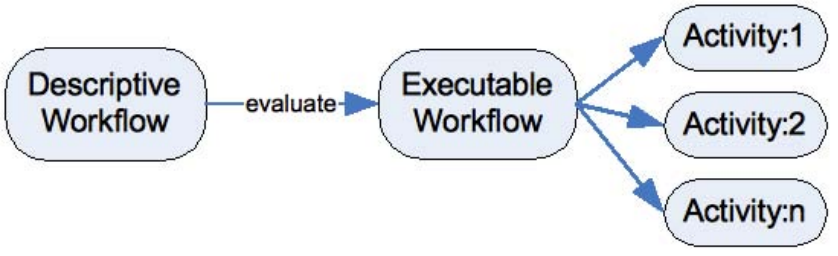


Fig. 1. Workflow transformation

3.1 i-Analyst: An ADDM System

For experimenting purpose, we have created an ADDM system, named i-Analyst. The system aims to provide data mining algorithm testing environment. Each test case is represented as a data mining model (DMM) which describes data mining process as a workflow. Workflows in i-Analyst composes of series of activities, such as data acquisition, transformation, mining, evaluation, and presentation. The infrastructure allows testers to construct their models to test arbitrary algorithms to operate on available data. Both algorithms and data can be uploaded into the system and become available for sharing.

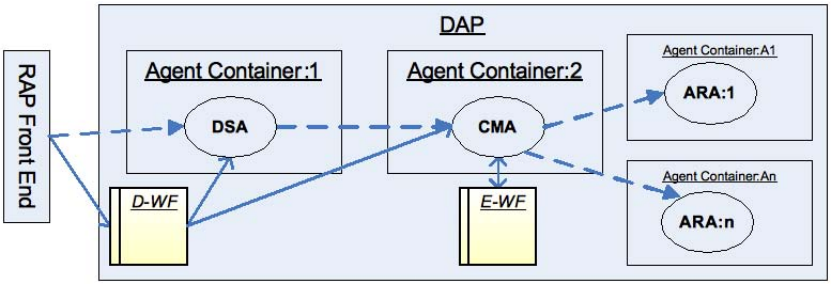


Fig. 2. The i-Analyst architecture

From the technical aspect, i-Analyst is a multi-agent system (MAS) that distributes sub-tasks to run on distributed platforms. The system composes of two core parts: Rich Ajax Platform (RAP) and Distributed Agent Platform (DAP) as shown in figure 2. Java

and its related technologies, e.g. Eclipse for RAP, Jade for agent SDK, are key tools for the system development.

RAP is a web-based front-end system that interacts with the user, while DAP is the automatic background process containing multiple agent containers. The main resources of the system are data mining algorithms, data sets, and reports. The three combines as a running instance, that is, mining data sets using a particular algorithm and producing a report. An instance (in this context, data mining model) is a workflow of data mining tasks. DAP consists of three main agents:

- DAP Service Agent (DSA) is the entry point that interacts with RAP. DSA chooses an appropriate Case Mediator Agent to carry out an instance execution.
- Case Mediator Agent (CMA) evaluates the descriptive workflow to a detail execution plan (executable workflow), controls and monitors workflow execution, and interacts with Activity Runner Agents to run activities.
- Activity Runner Agent (ARA) runs an activity assigned by a CMA.

4 Data Mining Model

A generic workflow composes of activities and transitions [15]. Each activity receives inputs and produces outputs. Outputs from the prior task can be used as inputs for the subsequent task only if the structures of inputs and outputs are compatible. Transition is triggered by events, usually task completion or conditions.

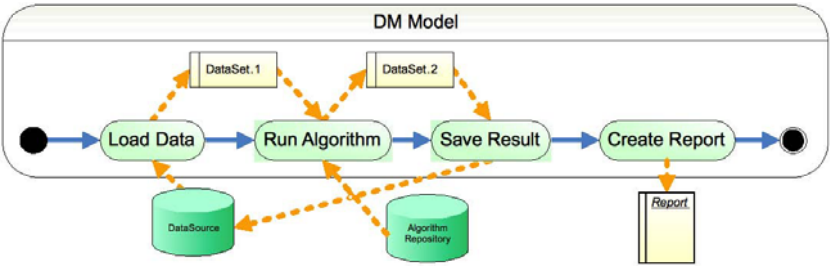


Fig. 3. A simple data mining model

Data mining model (DMM) in i-Analyst is represented as a workflow document which is based on a specific XML schema. DMMs are carried out by DAP. Figure 3 shows a simple data mining model defines a series of activities: LoadData, RunAlgorithm, SaveResult, and CreateReport. Each activity may produce any arbitrary data sets and store them in the model memory for other activities to consume. Sharing inputs and outputs does not occur in a single memory unit, due to distributed environment of the system. Inputs and outputs are determined to be transferred from one agent to another

agent as needed. Since each activity is run by an ARA; ARA stays alive until there is no dependency to it.

5 Descriptive Workflow Evaluation

The schema of DMM is based on an XML Schema Definition (XSD); figure 4 shows a simplified schema. Model is the data mining model itself which contains activities and data sets. Activity is an abstract class which defines generic structure of an activity. Each activity receives input and output dataset; while parameter specification is defined in the actual DMM contents. LoadData, SaveResult, RunAlgorithm, and CreateReport are sub classes extending from Activity. Each concrete activity defines its required parameters that are configured specifically for each model instance (describe in the later section). Figure 5 shows a sample content of LoadData in descriptive format. A system variable, i.e., ${dataset_dir} is to be replaced with actual values of each agent environment. In this example, ARA loads a data set from a CSV file located at ${dataset_dir}/dpriceData.csv into a dataset dpriceData.1.

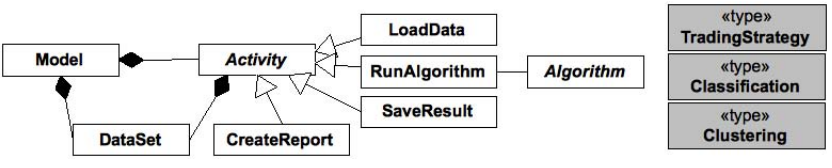


Fig. 4. simplified DMM schema

```
<activity id="1" type="LoadData" name="LD1" nextActivityIds="2">
   <inputDataSources>
      <dataSource id="1111" name="dpriceData.1" type="CSV"
        path="${dataset_dir}/dpriceData.csv">
          <fields>...</fields>
       </dataSource>
   </inputDataSources>
</activity>
```

A sample content of LoadData in descriptive format.

Another example, i-Analyst allows users to upload new algorithms to the system; the algorithms are distinguished by namespaces. An algorithm may contain several related classes, these classes are packed into a file, in this case Java Archive (JAR) file. A descriptive algorithm name only defines fully qualified name of the algorithm including namespaces to be used in the data mining model, e.g., ianalyst.algorithm.Spam. The evaluation process for the algorithm name will add additional information, that are (i) the file path where the algorithm JAR file is located and (ii) the agent name which

is assigned to run this algorithm depending on the algorithm requirements, such as language and libraries. For example,

```
<algorithm name="ianalyst.algorithm.Spam"/>
```

is evaluated to

```
<algorithm name="ianalyst.algorithm.Spam"
 jar="/contents/algorithm/uploads/jar001.jar aid="ara001">
```

Similar to the algorithms, a descriptive data source only denotes the name with fully qualified name, e.g., stock/marketA/dpriceData2009. Other additional configurations can also be supplied, for instance, localAccessOnly indicates whether the activity should be executed on the remote platform where the database is accessible, hence the same network. For example,

```
<dataset name="stock/marketA/dpriceData2009" localAccessOnly="true"/>
```

is evaluated to

```
<dataset name="stock/marketA/dpriceData2009"
 uri="jdbc:mysql:http://analyst.it.uts.edu.au:3306/ianalyst"
 aid="ara001">
   <fields>...</fields>
</dataset>
```

In case of view data sets, the data set that is generated from data processing instructions, e.g. SQL statements, the executable form will be as following

```
<dataset name="stock/marketA/dpriceData2009"
 uri="jdbc:mysql:http://analyst.it.uts.edu.au:3306/ianalyst"
 aid="ara001" dpi="true">
  <dpiSequence>
    <dpi type="sql" key="proc01" value="SELECT * FROM stock WHERE date
       BETWEEN '2009-01-01' AND '2009-12-31'"/>
  </dpiSequence>
  <fields>...</fields>
</dataset>
```

6 Implementation

The agent collaboration model is another critical issue which significantly determines the entire system performance. In this section, we describe how agents work in the system.

6.1 Agent Collaboration Model

Figure 6 shows the sequence diagram for agent message passing. The process starts from the DSA receives a request to run a DMM instance from RAP. It then searches for

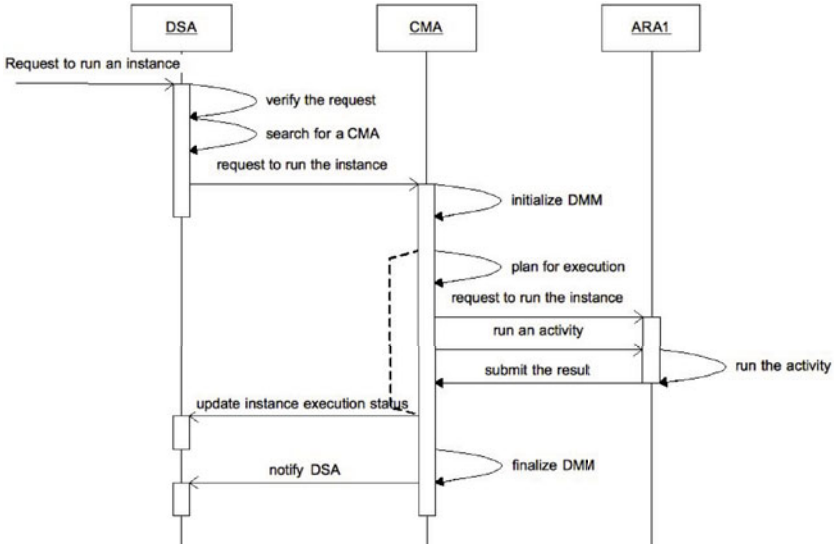


Fig. 5. Sequence diagram for agent message passing

a CMA to run the instance. The CMA receives the forwarded instance running message and retrieves the DDM instance from the system database. The CMA evaluates the DDM instance, in this context, the descriptive workflow, to the executable workflow. The CMA then spawns respective ARAs to run activities. An ARA runs an activity received from the CMA. It requests for required inputs from the CMA, in which the CMA may direct the ARA to an appropriate source if the CMA does not have an access to the data. The ARA completes the activity and produces outputs. A security constraint is applied here; the ARA determines whether to send the outputs back to the CMA or possesses it for future request. The CMA receives completion notification from ARA and forwards the notification to the DSA.

6.2 Example

In this section, we demonstrate how the concept is implemented. From figure 7, system variables, e.g., ${project_classpath}, ${dataset_dir}, ${result_dir}, ${report_dir}, are to be replaced with local values of each agent environment. In this example, we load a data from a CSV file located at ${dataset_dir}/dpriceData.csv into a dataset dpriceData.1. The dataset is visible throughout the model and of course shared with other activities. RA1 requires dpriveData.1 hence CMA sends the dataset to the RA1 ARA. Result of applying MovingAverage algorithm is the transferred into the dataset result. The last activity is to save the dataset into a file. In this example, the model exports the dataset to ${result_dir}/resultData.csv.

```
<instance id="100679" name="Instance1">
  <dataSets>
    <dataSet id="12345" name="dpriceData.1">
      <fields>
        <field name="trading_date" type="xs:string" />
        <field name="sum_daily_price" type="xs:decimal" />
      </fields>
    </dataSet>
    <dataSet id="54321" name="result"><fields>...</fields></dataSet>
  </dataSets>
  <activities first_activity_id="1">
    <activity id="1" type="LoadData" name="LD1" nextActivityIds="2">
      <inputDataSources>
        <dataSource id="1111" name="dpriceData.1" type="CSV"
          path="${dataset_dir}/dpriceData.csv">
          <fields>...</fields>
        </dataSource>
      </inputDataSources>
      <outputDataSets><dataSet id="12345" /></outputDataSets>
    </activity>
    <activity id="2" type="RunAlgorithm" name="RA1"
      previousActivityIds="1" nextActivityIds="3">
      <algorithm id="2222" name="MovingAverage"
        path="${project_classpath}"
        className="ianalyst.algorithm.impl.MovingAverage">
        <parameters>
          <inputParam key="m" value="5" type="xs:int" />
            ...
        </parameters>
      </algorithm>
      <inputDataSets>
        <dataSet id="12345">
          <fieldMapping>
            <field localName="date" orgName="trading_date" />
            <field localName="price" orgName="sum_daily_price" />
          </fieldMapping>
        </dataSet>
      </inputDataSets>
      <outputDataSets><dataSet id="54321" /></outputDataSets>
    </activity>
    <activity id="3" type="SaveData" name="SD1"
      previousActivityIds="2" nextActivityIds="4">
      <inputDataSets><dataSet id="54321" /></inputDataSets>
      <outputDataSources>
        <dataSource id="3331" type="CSV"
```

```
            path="${result_dir}/resultData.csv">
            <fields>...</fields>
          </dataSource>
        </outputDataSources>
      </activity>
      <activity id="4" name="CreateReport1"
        previousActivityIds="333" type="CreateReport">
        <dataFile path="${result_dir}/tt"/>
        <reportTemplate id="4444" name="CR1"
          path="${reporttemplate_dir}/ReportTemplate/RT01.zip">
          <fieldMapping>
            <fields>...</fields>
          </fieldMapping>
        </reportTemplate>
        <report path="${report_dir}/Report/yy" type="PDF"/>
      </activity>
    </activities>
  </instance>
```

Fig. 7. Content of a DMM.

The descriptive DMM content in figure 7 reflects the figure 5. The execution process takes place as follow. The chosen CMA is promoted to carry out the workflow. The DMM is evaluated to an executable workflow as described in section 4.3. The executable workflow which has four activities is and monitored by the CMA. The CMA requests target platforms to spawn respective ARA. The CMA sends activity1 to the respective ARA1, in which the ARA must be on the same site where the data is available. ARA1 loads data from into the memory, maintains in its memory, and notifies the CMA of the completion. The CMA notifies ARA2 to run activity2. The CMA sends the algorithm content to ARA2. ARA2 acquires for input dataset which was produced by activity1. CMA notifies ARA1 to send the data to ARA2. In the planning stage, ARA1 and ARA2 are placed in the same or neighbor network to minimize the communication cost. ARA2 applies the algorithm on the data, then produces a result set, and notifies CMA for the completion. CMA notifies ARA3 to starts activity3. ARA3 fetches required data and saves to its local directory which is also visible to ARA4. After the completion of ARA3, ARA4 receives notification from the CMA and starts activity4. ARA4 merges the output data and a report temple into a report document, and places on the designated directory. ARA4 notifies the completion to the CMA. The CMA receives all completion confirmations and finally notifies the DSA for the workflow completion.

7 Evaluation

The integration method augments the existing ADDM architecture by introducing additional responsibility to the agents to perform according to the assigned activity. The data mining models are improved to be used as workflows, so the system only maintains only one document, the descriptive one. The workflow is used to help planning of agents. With descriptively provided workflow, the responsible agent (CMA) can insert more

details to the workflow for the execution. The benefit is that the system does not need to disclose internal information for the descriptive workflow producer, e.g., third-party front-end system, therefore data and system configuration are hidden. Model evaluation determines the resource utilization. Planning may concerns the cost of communication and computation time. Planning may result in a plan to execute all activities locally relatively to the CMA if the size of data is too excessive to transfer across the networks. A group of consecutive activities may be executed on the same containers as to minimize data transmission cost. In terms of robustness, exception handling is achieved through agent negotiation. Agents can be programmed to handle pre-deterministic exception internally, before acquiring for others assistance. The system benefits from less interruption from exceptions that can be handled. Non-deterministic exceptions are forwarded to the CMA and the CMA decides to start the activity else where.

8 Conclusions

We present an integration of agent-based workflows with agent-based distributed data mining. The main idea is to blend the two technologies together by treating the data mining models as workflows. Since agents are already an integral part of the system, augmenting the agents responsibility to enact the workflow does not require revising the entire system structure. We have demonstrated the use of workflow in our existing system which is used for the ongoing research focusing on supporting more complex data mining tasks and how to reflex the model schema as agent ontology as to improve the system implementation when the two concepts are closely mapped.

References

1. Specification, W.M.C.: Workflow Management Coalition Terminology & Glossary (Document No. WFMC-TC-1011). Workflow Management Coalition Specification (1999)
2. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explorations 11(1), 10–18 (2009)
3. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: rapid prototyping for complex data mining tasks. In: KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 935–940. ACM, New York (2006)
4. Berthold, M.R., Cebron, N., Dill, F., Gabriel, T.R., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K., Wiswedel, B.: KNIME: The Konstanz Information Miner. In: Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007). Springer, Heidelberg (2007)
5. Shepherdson, J.W., Thompson, S.G., Odgers, B.R.: Decentralised workflows and software agents. BT Technology Journal 17(4), 65–71 (1999)
6. Cai, T., Gloor, P.A., Nog, S.: Dartflow: A workflow management system on the web using transportable agents (1997)
7. Huang, C.J., Trappey, A.J., Yao, Y.H.: Developing an agent-based workflow management system for collaborative product design. Industrial Management and Data Systems 106(5), 680 (2006)
8. Judge, D.W., Odgers, B.R., Shepherdson, J.W., Cui, Z.: Agent-enhanced workflow. BT Technology Journal 16(3), 79–85 (1998)

9. Odgers, B.R., Shepherdson, J.W., Thompson, S.G.: Distributed workflow co-ordination by proactive software agents. In: Intelligent Workflow and Process Management. The New Frontier for AI in Business IJCAI-99 Workshop (1999)
10. Savarimuthu, B.T.R., Purvis, M., Fleurke, M.: Monitoring and controlling of a multi-agent based workflow system. In: ACSW Frontiers '04: Proceedings of the Second Workshop on Australasian Information Security, Data Mining and Web Intelligence, and Software Internationalisation, pp. 127–132. Australian Computer Society, Inc., Australia (2004)
11. Yoo, J.J., Suh, Y.H., Lee, D.I., Jung, S.W., Jang, C.S., Kim, J.B.: Casting mobile agents to workflow systems: On performance and scalability issues. In: Mayr, H.C., Lazansky, J., Quirchmayr, G., Vogel, P. (eds.) DEXA 2001. LNCS, vol. 2113, pp. 254–263. Springer, Heidelberg (2001)
12. Yan, Y., Maamar, Z., Shen, W.: Integration of workflow and agent technology for business process management. In: The Sixth International Conference on CSCW in Design, pp. 420–426 (2001)
13. Savarimuthu, B.T.R., Purvis, M., Purvis, M., Cranefield, S.: Agent-based integration of web services with workflow management systems. In: AAMAS '05: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1345–1346. ACM, New York (2005)
14. Moemeng, C., Gorodetsky, V., Zuo, Z., Yang, Y., Zhang, C.: Agent-based distributed data mining: A survey. In: Cao, L. (ed.) Data Mining and Multi-agent Integration, pp. 47–58. Springer, Boston (2009)
15. Fischer, L.: BPM & Workflow Handbook. Workflow Management Coalition (2007)

Pilot Study: Agent-Based Exploration of Complex Data in a Hospital Environment

Ted Carmichael[1], Mirsad Hadzikadic[1], and Ognjen Gajic[2]

[1] University of North Carolina at Charlotte, College of Computing and Informatics, Charlotte, North Carolina, USA
{mirsad,tdcarmic}@uncc.edu
[2] Mayo Clinic
Rochester, Minnesota
gajic.ognjen@mayo.edu

Abstract. We present the results of a pilot study created to explore a sub-set of complex data, utilizing an agent-based model simulation tool. These results center on data taken from hospital admission records, tracking patient attributes and how they relate to patient outcomes. The focus of this work is to highlight three design principles: 1) using an iterative process between the modeling of a system and the grounding of that simulation with real-world data; 2) a focus on agent primitives, emphasizing bottom-up emergence of effects, rather than top-down control; and 3) integration of various "theories" of patient care and hospital effectiveness as a method for experimentation with Complex Adaptive System-based data mining.

Keywords: Complex Adaptive Systems, Agents, Data Mining, Machine Learning.

1 Introduction

Hospitals represent a complex environment for patient care, with many variables affecting patient outcomes. These variables can often interact in surprising ways, producing non-linear effects in a dynamic environment. Complex Adaptive Systems (CAS) [1–4] techniques can be used to design an Agent-based Model (ABM) for simulating these systems [5], allowing for a meaningful exploration of large datasets related to patient outcomes.

1.1 Design Principles

In building an ABM for a complex environment, such as that found in a hospital setting, our design plan incorporates three distinct principles, designed to lead to a robust and useful simulation environment. The first of these is that the initial simulation is not intended to be the final product. Building a meaningful simulation is by necessity an iterative process, and the first stage is to have a model that is both flexible enough to allow calibration to real-world data, yet powerful enough to make meaningful predictions that can be used to suggest what additional data needs to be collected.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 16–26, 2010.

The second principle we follow relates to both of these points, in that the model emphasizes the use of agent primitives. By focusing on primitives that use simple rules of interaction, bottom-up emergence of salient phenomena is favored over other methods that explicitly incorporate such phenomena in a top-down manner. Often, models that attempt to prescribe too many effects lack the flexibility to adequately handle or predict unexpected outcomes.

The third principle is that the model should not be intended to replicate a single hospital environment, but rather to be generally applicable to a variety of environments. One way to accomplish this is to incorporate additional flexibility in the simulation by allowing operator control over key attributes. This allows for more general applicability by allowing various scenarios – in this case, theories of patient care and patient tracking – to be individually initialized for repeated and varied experimental simulation runs. This is similar to the approach used by Whitmeyer et al. [6] allowing for multiple social theories to be incorporated in a simulation test-bed environment.

1.2 CAS Data Mining

Complex adaptive systems differ from machine learning and data mining tools in that CAS simulations use all patients (or *agents*) to play out (one could call it "predict") what will happen over time with this particular population. Even the time component of each simulation is important, as it indicates how long it will take to reach a specific conclusion. In CAS simulations, insights emerge over time. For example, as we will see in the examples below, the fact that the patient population formed three clusters in the first experiment was not predefined and the varied shapes/cohesiveness of these clusters emerged over time. We can run numerous simulations just to see whether differing random distributions of patients or parameter settings would make any difference in the final outcome.

Data mining and machine learning tools work differently. They require that the study designer divide the data set into two disjoint groups: the training group and the testing group. The algorithm then uses the training set to produce, say, the classification tree, while the testing set is used subsequently to evaluate the predictive accuracy of this tree. We can combine the two approaches in the following way: use CAS to produce emergent properties of the population, which are then evaluated using the testing set.

1.3 Outline

Here we present the results from the first stage of this long-term research plan for building such a system. We identify the five patient attributes used in this pilot study, and detail the simulation tool used to experiment with these attributes. Experimental results are presented, which are used to highlight the inherent flexibility of the CAS paradigm, as well as the three design principles mentioned above. First, we perform a CAS analysis of the population. Then, we combine a CAS and a data mining method. These results are discussed and generalized in order to present a template for the applicability of agent-based CAS models to a wide variety of complex data.

2 Simulation Details

2.1 Patient Data

A large dataset of anonymous patient data has been collected from the Mayo Clinic in Rochester, Minnesota. This data represents information that is generally available in any hospital setting, representing ~ 200 patient variables and attributes across 688 patients. All of the patients are aged 65 or older and were subsequently admitted to the ICU (Intensive Care Unit). Each were tracked throughout the hospital admission process, from initial contact to final discharge, and the data gathered includes such information as: demographics, diagnostics, types of patient care, and final outcomes.

2.2 Agent Attributes

In this pilot study we look at five patient attributes. The reason why we selected only five variables has to do with the property of CAS that "less is more," indicating that a few key variables often model the underlying phenomenon better than if we use all, both relevant and irrelevant, variables for the same task. We have also found that it is both more fruitful and more clarifying to start with a minimum number of variables, and then build up to higher levels of complexity.

These five attributes are divided into two groups: the influence attributes and the target attribute. In this way the experimental design will explore the effects of the four influence attributes, in terms of how they affect the target. The target attribute in this study is 60-day mortality, a common (if simple) measure of patient outcome. The four influence attributes are further sub-divided: two of them – "patient gender" and "age group" – are demographic variables, representing factors that are unalterable; the remaining two – "hour of the day" (time of admission) and "time to ICU admission" – are variables that are subject to some measure of control across all patients.

The chart in Table 1 shows the range of attribute values for each influence variable. The target variable, "60-day mortality," has two possible values: 'positive' or 'negative.'

Table 1. Attribute values of four influence variables

	Fixed Attributes		Controllable Attributes	
Attribute	Gender	Age Group	Hour of the day	Hours to ICU
# of values	2	3	24	5
Range of values	"M" or "F"	65-74, 75-84, and >85	0 through 23	<10 min., <1 hr., <5 hrs., <100 hrs., and >100 hrs.

2.3 Simulation Environment

The simulation environment is a square torus with 75x75 lattice cells, and presents mobile "agents" to represent each patient as found in the existing data. These agents are initialized with the values for each of the five attributes illustrated above. At initialization, the agents are randomly placed within the simulation environment.

The simulation environment is implemented using NetLogo [7] and the general simulation tool developed by two of the authors [8, 9].

2.4 Experimental Design

In this pilot study, each "patient" agent selects another agent at random, calculates her similarity to that agent, and then moves towards or away from that agent based on the degree of similarity. Operator controls allow for alteration of the relative weights for each of the four attributes, in terms of calculating the similarity measure. There is also a controller to adjust the gradient of movement – i.e., the speed in which agents move towards or away from each other. Figure 1 shows these controls in this pilot simulation interface.

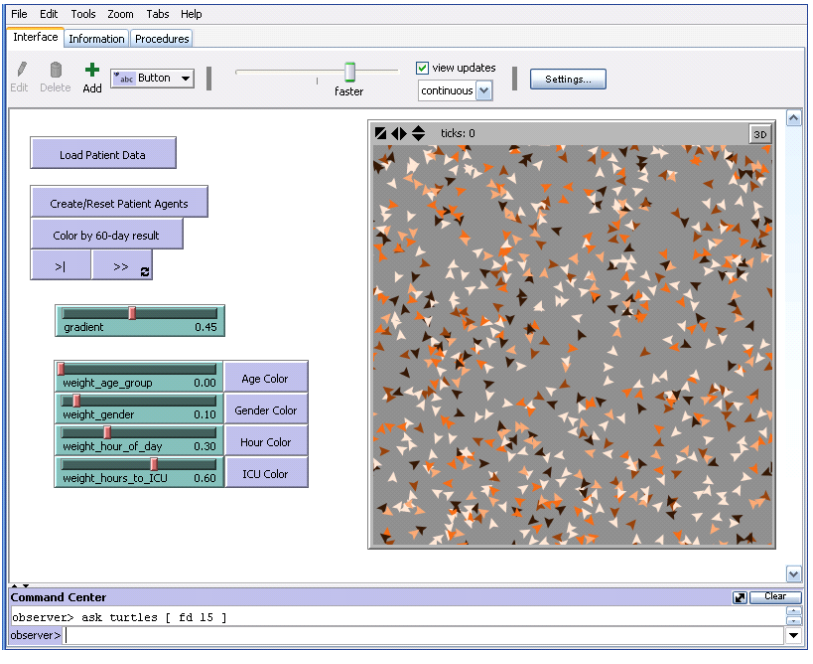


Fig. 1. Pilot simulation environment and interface controls

Here (Figure 1) the agents are colored to represent their current attribute values for "hours to ICU." (Every patient in this dataset is admitted to ICU at some point.) The lighter colored agents represent patients who have a quick admission to ICU while the darker colored agents are those patients who have a longer wait to admission. These colors can be altered to represent any of the attribute values – or combinations of these values – for enhanced visualization analysis.

The similarity algorithm is scaled for each influence attribute and assigned a value between [-1, 1]. For example, the attribute "gender" has only two possible values: [1] if

the two agents are the same gender and [-1] if they are different genders. The attribute "hour of the day" for hospital admission has a positive value between [0, 1] if the two agents were admitted between zero and six hours of each other, a value of [0] if they were admitted exactly six hours difference, and a value between [-1, 0] if they were admitted between six and twelve hours difference. Thus, the closer to the same time of day two agents were admitted, the higher the similarity measure is between them for this attribute-value.

The independent calculations for each attribute-value similarity are then combined together, using the slider weights as shown in Figure 1. In this way, the operator can perform multiple experiments on the data while giving either more or less importance to various factors.

In general, the CAS-based experiments will follow the subsequent protocol:

1. Gather data and divide them into two sets: training and testing
2. Identify potentially relevant attributes and target variables
3. Define operator controls (simulation parameters)
4. Define the number of runs and stopping criteria
5. Run simulations with the training data set
6. Evaluate the model with the testing set
7. Modify parameters and re-run the simulations if necessary

3 Experimental Results

Multiple simulation experiments were run using the pilot simulation clustering environment, and analyzed based on the resulting clusters. The results from two of these sets of experiments are presented here; the first using "hours to ICU admission" as the dominant similarity measure; the second introducing the training and testing sets, and using "hour of the day" as the dominant measure.

3.1 First Experiment Results

In Figure 2 we see: the initial state; the mid-state; and the end-state of a typical simulation run. In this example, the "hours to ICU admission" is used as the dominant measure of similarity among the agents, as well as for the visualization color scheme.

In the final frame of Figure 2 we see that the agents are clearly segregated according to the values for the dominant clustering attribute; however, the clusters are not perfect, due to the influence of other attributes. In a second simulation run, "hours to ICU admission" was used as the only clustering attribute, and the results were similar but more clearly defined (Figure 3).

By examining the makeup of each cluster, in terms of the 'target' variable "60-day mortality," we can draw some simple inferences about these clusters. The lightest colored cluster represents agents that were admitted to the ICU less than one hour from initial patient tracking. The 60-day mortality rate of these patients is the highest of the three clusters: 83 of 350 patients, representing approximately 23.7%. This makes intuitive sense, as the newly admitted patients in most dire need of ICU are quickly processed: many (the lightest color) in under 10 minutes. The medium cluster, representing

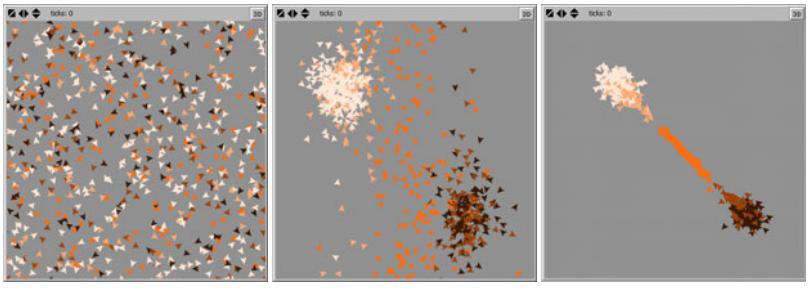


Fig. 2. Three stages of a sample simulation run. Lighter colors indicate a shorter time to ICU admission; dark colors represent a longer time.

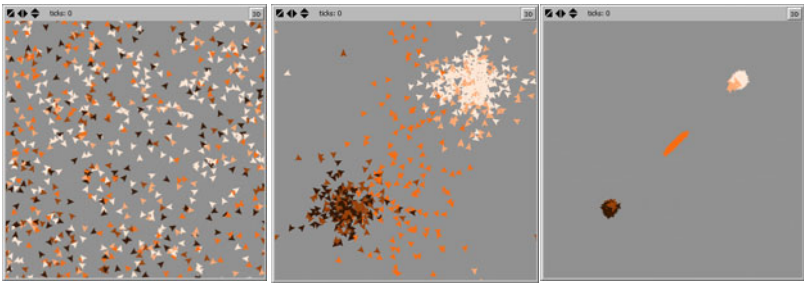


Fig. 3. Three stages of a simulation run. Same as in Figure 2, except only a single clustering attribute is used. Final clusters are much more distinct without the influence of additional variables.

ICU admissions between one and five hours, has a 60-day mortality rate of only 16 out of 128 patients, or 12.5%.

Interestingly, however, the last cluster – those admitted to the ICU between 5 and 100 hours (light brown) or after >100 hours (dark brown) – represents an increase in the 60-day mortality rate over the medium cluster: 39 out of 211 patients, or ~18.5%. This could indicate an area of concern for the hospital. Is there an undue delay in getting some patients to the ICU? Could quicker processing improve the outcomes for these patients?

3.2 Second Experimental Result – Training vs. Testing Data Sets

The data mining approach to the design of this pilot study starts with the division of the patient data set into the training and the testing groups. The target variable is again the 60-day mortality of the patients; the clustering attribute for this experiment is the "hour of the day" for admission to the hospital. The clustering algorithm is run for 5000 simulation time steps, with the gradient beginning at 15.0, and moved during the run down to 1.0.

Figure 4 shows three stages of a sample simulation run, as the agents move into two clusters. Multiple simulation runs conducted with these same settings show that these two clusters form repeatedly: one cluster of 143 patients that represent the late-night

and early-morning hours of the day (from 11:00 pm to 10:00 am) and one with 201 patients representing the daytime and early evening hours (approximately 11:00 am to 10:00 pm). Analysis of these clusters show that the “early hours” group tends to have a much lower 60-day mortality rate than the other group: 16.1% vs. 22.9%.

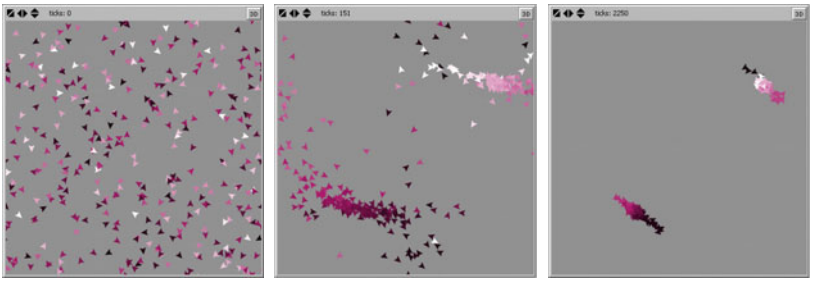


Fig. 4. Three stages of a sample simulation run using “time of the day" as the clustering attribute. White represents hour 0 (12:00 midnight), black represents hour 23 (11:00 pm) and shades from light to dark indicate hours 1 through 22.

Thus, we can make two predictions for our test data: 1) that two dominant clusters will form, representing the same hours of the day as with the training data; and 2) the late-night and early-morning cluster will have the lower mortality rate, approximating 16%.

The test simulation run closely matched these predictions. Two dominant clusters formed, cluster 1 representing the hours 11:00 pm – 9:00 am (125 patients) and cluster 2 representing 11:00 am – 9:00 pm (197 patients); two secondary clusters also formed, one for the hour beginning at 10:00 am (9 patients) and one for 10:00 pm (13 patients). As with the training data, the late-night and early morning hours had the lower mortality rate: cluster 1 has a 60-day mortality rate of 12.8%, while cluster 2 has a 60-mortality rate of 24.4%. Table 2 presents a summary of these results.

Table 2. Summary of results. Hour 0 represents 12:00 midnight; hour 23 represents 11:00 pm

Training set	Count	Hours	60-day Mortality
Cluster 1:	143 patients	0 – 10, 23	16.1%
Cluster 2:	201 patients	11 – 22	22.9%
Test set			
Cluster 1:	125 patients	0 – 9, 23	12.8%
Cluster 2:	197 patients	11 – 21	24.4%
Cluster 3:	9 patients	10	22.2%
Cluster 4:	13 patients	22	15.4%

As with the previous experiment, these results indicate an area of concern for the hospital. It seems clear that those who are admitted to the hospital during the late-night and early morning hours have a lower 60-day mortality rate, when compared to the patients who are admitted during the daytime hours.

4 Discussion

4.1 Iterative Process between CAS Simulation and Real-World Data

One must be careful in assigning causal factors to correlations in the data. The first experimental results do not represent a definitive conclusion that ICU evaluation-and-admission needs to occur more rapidly. There are many, many other factors that could come into play. Nor do the correlations between the time of day of admission and the subsequent mortality rate necessarily indicate a causal factor. Consequentially, the value of these simulation results is that they suggest research questions that need to be investigated. The connection between ICU admission times and patient outcomes should be looked at more carefully, and more precise data on these events collected. Similarly, the second experimental results may indicate that there are circumstances or practices that differ, between daytime admissions and those that occur during the late night/ early morning hours.

In practice, the CAS simulation can be designed based on as much relevant information as can be gathered. In this was, the simulation represents a "hypothesis" of patient outcomes, making assumptions about which factors are important, their degree of importance, and how they inter-relate. As a hypothesis, the simulation results must be not only calibrated to real-world data, but also validated by making testable predictions. If necessary, the simulation is then manually re-calibrated with additional data.

Once this process is complete, the simulation can be run with data from an independent sample, and validated with the results of these independent patient outcomes while the patients are in the hospital. A rigorous, iterative process ensures that the simulation results are sound, thus providing a useful tool for evaluating new hospital environments in real-time. In this way, the iterative process between simulation experiments and real-world data will be able to produce actionable results that can improve patient care.

4.2 Focus on Primitives

An important consideration in the design of an ABM is the amount of flexibility the system represents. It should be noted that the results of this pilot simulation can be replicated with other statistical techniques and traditional methods of data mining. For example, conceptual clustering/unsupervised machine learning methods, e.g. INC that one of the authors is very familiar with [10], can use complete agent descriptions and a predefined similarity function to drive the formation of clusters, which are subsequently used for prediction of category membership for the remaining, previously unseen cases.

However, CAS studies allow the project designer to let many aspects of the underlying phenomenon emerge, instead of being specified ahead of time. Hence, time is an extremely valuable aspect of complex adaptive systems that cannot be easily replicated with other statistical or machine learning methods. CAS simulations can be run numerous times under varying conditions to find both the attributes subsets and starting conditions that produce the most desirable results. In this study, we chose only a limited set of patient data, and only five out of ~200 possible variables, in order to illustrate both the power and the flexibility of CAS simulation methods.

Even though other methods of data analysis exist, in this pilot study neither the agents nor the system itself were programmed to utilize these methods. Yet they produced

these same results. The system "learned" the patient outcomes of various settings and demonstrated these learned insights in a simple, visually accessible way. Such flexibility is important, in that it inherently allows for surprising and unexpected results, commonly referred to as emergence in CAS scientific literature [11–13].

A system as complex as a hospital has many factors that can affect patient outcomes. Anything from what time of day a patient is first admitted; to which doctors and nurses are currently working; to the general procedures the hospital follows in terms of data tracking, or sanitation, or a host of other considerations. In such a complex environment, the more a simulation avoids prescribing certain effects, and instead allows them to emerge from primitive agents and their interactions, then the less likely the simulation results will be due to explicit assumptions programmed into the model. In this way, an ABM that can learn known and validated effects in a hospital environment is preferred over one that simply assumes these effects.

4.3 Integration of Theories in ABM Design

In this pilot study, the operator has control over the various weights assigned to each influence attribute, and is thus able to alter the relative importance each has in determining agent similarity. This is analogous to our planned research design, in which the simulation acts as a testbed for deep and complex data analysis. In our proposed methodology, the operator will be able to set certain controls that input various theories of patient care. For example, one theory may be "caregiver fatigue;" another may be "germ susceptibility;" and a third could be "information tracking performance."

Consider one scenario, where a hospital implements a new information tracking system to confirm medication types and amounts. Any patient data that must be communicated across doctors and nurses will suffer some form of information degradation; i.e., every time information passes from one person to another, it can be restated, have a different emphasis, be posed with a different interpretation, or even mis-communicated with wrong information. The hope is that this degradation is a small as possible. Thus, a complicated question for hospitals is how best to implement a system that is not only rigorous towards exact and unbiased information passing, but also avoids being so onerous that it won't be followed by the staff.

The simulation operator may wonder if his hospital's new method of information passing/tracking is effective: is it actually improving patient outcomes? He may surmise that the new method is not being utilized correctly. With a rigorous CAS model available, the operator can input the "information tracking performance" theory, and then observe what level of information degradation the model requires in order to produce the known, real-world outcomes.

If the simulation settings that match the hospital's true outcomes require an elevated level of poor communications, then this would lead the operator to question the performance of the hospital's new methods. Further intervention – such as increased emphasis on the importance of robust communication, or perhaps more user-friendly design of tracking methods – may result in reducing errors and improving patient outcomes. This improved outcome could subsequently be re-run in the simulation to show that indeed, higher levels of accurate information passing in the model match the real-world results of improvements in patient care.

Thus, a rigorous and well-designed CAS implementation can be used to 1) indicate an area of concern – in this case, patient information that degrades too much among hospital staff; and 2) subsequently confirm that a new system of communication is more robust, accounting for an improvement in patient care.

5 Conclusions

The above scenario of "information tracking performance" envisions a robust CAS model that is the desired result of our proposed research program. In this example, the operator controls represent only part of the strength of such a system; complementing this aspect is the inherent flexibility of the CAS tools themselves. Flexibility is the key to implementing a simulation robust enough to capture surprising effects in the data-rich environment of a hospital.

The path of patient care – from first contact to final discharge – is a tremendously dynamic system. As such, it is rife with non-linear effects and multiple levels of causal feedback. The power of CAS comes from its focus on primitives, such that top-down programming of presumed outcomes is avoided in favor of bottom-up emergence of these effects. Such flexibility is inherently important to the iterative process outlined above.

As with the pilot model demonstrated here, a well-designed, calibrated, and validated ABM can learn which attributes are important. More importantly, such a system can also learn how these many factors either amplify or mitigate outcomes as the agents interact with each other. The iterative process outlined above is the key to this process.

Finally, since every hospital environment is different, a well-designed simulation tool should allow an operator to input various theories of patient care. In this way, the operator can tailor an experimental design that more precisely fits the environment being studied, while still benefiting from the vast amount of data available across many different environments.

References

[1] Waldrop, M.M.: Complexity: The Emerging Science at the Edge of Order and Chaos. 1st Touchstone edn. Simon & Schuster, New York (1993)
[2] Holland, J.H.: Complex Adaptive Systems. Daedalus 121(1), 17–30 (1992)
[3] Boccara, N.: Modeling Complex Systems. Springer, New York (2004)
[4] Levin, S.A.: Complex Adaptive Systems: Exploring the Known, the Unknown and the Unknowable. Bulletin-American Mathematical Society 40(1), 3–20 (2003)
[5] Gilbert, G.N.: Agent-based Models. Sage Publications, Los Angeles (2008)
[6] Whitmeyer, J., Carmichael, T., Eichelberger, C., Hadzikadic, M., Khouja, M., Saric, A., Sun, M.: A Computer Simulation Laboratory for Social Theories. In: 2008 IEEE/WIC/ACM International Conference on Intelligence Agent Technology, Sydney, Australia (December 2008)
[7] Wilensky, U.: NetLogo. Center for Connected Learning and Computer-Based Modeling. Northwestern University, Evanston (1999), http://ccl.northwestern.edu/netlogo/

[8] Dréau, D., Stanimirov, D., Carmichael, T., Hadzikadic, M.: An Agent-based Model of Solid Tumor Progression. In: 1st International Conference on Bioinformatics and Computational Biology (2009)
[9] Carmichael, T., Hadzikadic, M., Dréau, D., Whitmeyer, J.: Towards a General Tool for Studying Threshold Effects Across Diverse Domains. In: Ras, Z., Ribarsky, W. (eds.) Advances in Information and Intelligent Systems. Springer, New York (2009)
[10] Hadzikadic, M., Bohren, B.F.: Learning to Predict: INC2.5. IEEE Transactions on Knowledge and Data Engineering 9(1), 168–173 (1997)
[11] Holland, J.H.: Emergence: From Chaos to Order. Addison-Wesley, Reading (1998)
[12] Bedau, M.A., Humphreys, P.: Emergence. In: Contemporary Readings in Philosophy and Science. MIT Press, Cambridge (2008)
[13] Johnson, S.: Emergence: The Connected Lives of Ants, Brains, Cities, and Software. Scribner, New York (2001)

Multi-agent Information Retrieval in Heterogeneous Industrial Automation Environments

Stephan Pech and Peter Goehner

Universitaet Stuttgart, Institute of Industrial Automation and Software Engineering
Pfaffenwaldring 47, 70550 Stuttgart, Germany
{stephan.pech,peter.goehner}@ias.uni-stuttgart.de
http://www.ias.uni-stuttgart.de

Abstract. The trend towards the integration of application domains and technologies in the field of industrial automation and information technology (e.g. process control systems, computer technology and telecommunication) leads to an increasing complexity of systems and applications. Even though integration has always been one of the major goals of automation, it is still challenging. Currently, users have to run a large number of specific software tools to aggregate unstructured bunches of data into meaningful information. This is coupled with the need for specific knowledge to handle the different tools. This paper presents a concept for an agent-based information retrieval system, which tackles the challenges of information retrieval in heterogeneous environments. This is accomplished by representing data sources, search queries and the data retrieval process itself by means of software agents and ontologies.

Keywords: Multi-Agent Systems, Information Gathering, Industrial Automation Systems, Ontologies.

1 Introduction

The last twenty years brought enormous advances in information technology, both in the office applications and in the field of industrial automation. This is manifested e.g. by the success of internet technologies, which are widely adopted into local area networks, as well as widely standardised field bus communication. Since networking in both fields nowadays is mature, many promising approaches are developed to bring both worlds together to achieve a horizontal and vertical integration of information, including the temporal integration aspects for dynamically changing environments. The aim is a seamless integration of all levels of an enterprise with direct access to data sources from the management level down to the field level which contains e.g. sensors and actuators and the directly associated hardware for process monitoring and control. The work routine in industrial automation sites has become more and more like ordinary work routine in office environments. However, the tools developed for office environments are not always suitable for industrial automation environments, especially for information gathering tasks. Compared to users of ordinary information systems, users in industrial automation have different needs. Whereas in ordinary information systems state changes or information changes are the result of user activities, in industrial automation the process and the associated industrial automation system is primarily running on its own

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 27–39, 2010.

without the need for user interaction [1]. A typical issue is the involvement and integration of human intelligence and interaction in information gathering systems. It is a challenging task to include the different roles, the empirical experience and the qualitative intelligence of humans. Furthermore, it is challenging to support the different views on the system structure, data and functionality from the different individual users, such as engineers, process personnel and managers [2]. This is one of the reasons why alternative approaches, like intelligent recommendation frameworks for ERP systems [3] or agent-based data management and decision support systems [4] are employed.

Industrial automation systems are developing towards distributed, heterogeneous but also interconnected systems offering a huge amount of measured, stored and available live data which leads to ever more complex systems. Due to the distribution and heterogeneity of the database and the variety of the different software tools used to obtain data, information retrieval e.g. in industrial automation systems is processed by several manual steps and is therefore very time-consuming. Users have to query a lot of structurally and semantically heterogeneous data sources and have to handle different software tools to get the information they want. In addition, the demands on keeping pace with the steadily growing market competition force an increase in productivity which is regularly tackled with higher integration of the industrial automation systems. Therefore, new approaches, facilitating more efficient information retrieval processes are necessary. We developed an agent-oriented approach that enables an integrated access to data sources in industrial automation systems. In this paper, we propose a combination of the agent-oriented paradigm for building flexible decentralised systems and ontologies for representing knowledge. The paper is organised as follows. Section 2 gives a general overview over the problem area. Section 3 describes the required technologies for the solution approach. The proposed agent-oriented architecture is discussed in section 4 followed by a working example in section 5. Section 6 concludes with a brief summary and an outlook.

2 Information Retrieval Process in Industrial Automation Systems

Industrial automation systems such as those deployed in large industrial plants are known to be complex hardware and software systems. The automation system used within an industrial plant consists of distributed and heterogeneous data sources which are accessed via specific applications. Fig. 1 shows a typical structure of information elements in an industrial automation environment. The layered architecture of an industrial automation system consists typically of the process automation equipment like sensors and actuators and the directly associated hardware for process monitoring and control like Programmable Logic Controller (PLC) or Process Control Systems (PCS). Industrial IT-systems like Manufacturing Execution Systems (MES), Supervisory Control and Data Acquisition Systems (SCADA) establish interfaces to the different kinds of external business systems like Enterprise Resource Planning Systems (ERP), Supply Chain Management Systems (SCM) or Office Systems.

The steadily growing size and complexity of industrial automation systems makes it difficult to find out if, when and in which way a certain kind of information is available. If the information is available, it is still to decide whether the information is relevant for the current information gathering situation or not.

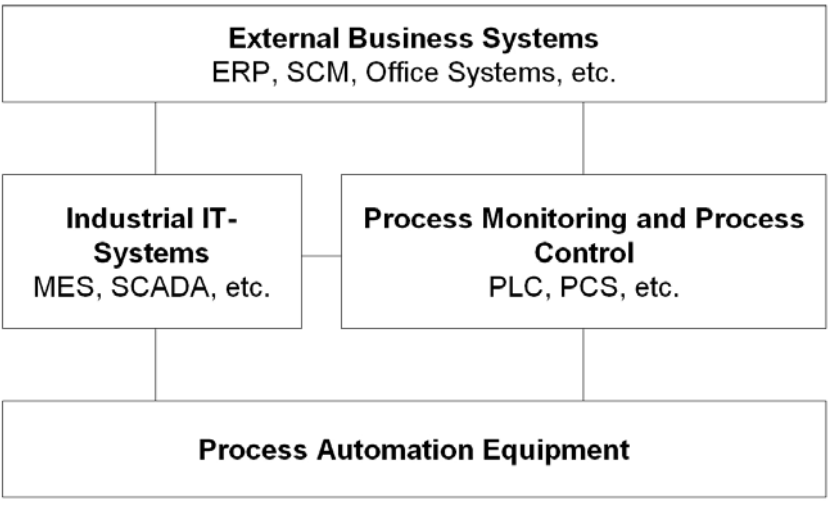


Fig. 1. Information elements in an industrial automation environment

2.1 Information Retrieval Process

For the holistic execution of search queries such as those concerning process analyses and optimisation, information from various systems has to be combined, and retrieved within heterogeneous systems. Driven by the aim to determine certain information from the industrial automation system, the user plans how to conduct a query. In a process-oriented perspective, this depends mainly on the users' practical knowledge and leads to a manual procedure (see Fig. 2).

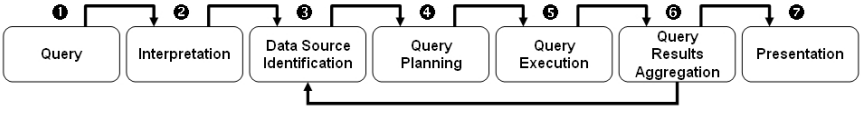


Fig. 2. Elementary query sequence

The procedure of information gathering consists of several steps. Step 1 and 2 are implicitly achieved by the user, as soon as he formulates a certain information demand, e.g. in the form of an SQL query. Steps 3 and 4, as well as step 6 are carried out manually by the user, whereas step 5 is executed by the system. To solve a query, the identification of the relevant data sources, depends on user knowledge. In step 4, the user plans the order in which e.g. the partial query steps have to be processed. The execution of a query, e.g. execution of a SQL statement, is done by the system. To precisely answer a query, an aggregation of certain query results (step 6) based on user specific knowledge, is often required. Therefore, system experts have to interpret the query results and expand or reduce the search space if needed. If search results are ambiguous,

step 3 to step 6 can be repeated. In sense of the query, every single data represents a valid search result.

This procedure could not be sufficiently automated until now, because current software systems are unable to interpret the query completely in form and content. Moreover, they are almost incapable to combine single search results correctly, because semantic relationships are missing [5].

The process of the sequence covers different steps and uses a bunch of software tools. It is crucial for the development of future automation environments to relieve human operators from this kind of mental work load. Therefore, it is necessary to develop an approach that, by applying information technology, assists the user in gathering information.

3 Required Technologies for the Solution Approach

The current industrial automation infrastructure is based on hierarchical multi-layer networks, connecting different information elements that communicate with each other using various methods and standards. As stated in [6], software agents are the appropriate technology for building a system which has to integrate distributed data and resources, or make legacy systems working together. To fully utilise the information available on all levels of a production enterprise, we combined the benefits of software agents with that from ontologies [7].

3.1 Agent-Oriented Concepts

Why are agent-oriented concepts relevant to information retrieval in industrial automation? Generally speaking, this question can be answered by looking at the properties of current industrial automation systems summarized from section 1:

- Industrial automation systems require the involvement and integration of human intelligence and interaction because engineers, process personnel and managers have individual views on structure, data and functionality.
- Industrial automation systems are distributed, heterogeneous but also interconnected systems.
- Industrial automation systems require the integration of flexible and adaptive software due to the long lifetime of the systems were changes affecting the software can be driven e.g. from data, structure or operating sequences of the system.

As highlighted in [8] these properties are similar to the general properties of complex decentralised systems. In the scope of our research work, the concept of a software agent is seen as an encapsulated software entity with a defined objective. An agent permanently tries to reach its objective with autonomous actions. Therefore it can interact with its environment and other agents, while keeping a persistent state. The agent-oriented paradigm supports the abstraction of a given problem into agents by means of six basic concepts that are autonomy, interaction, reactivity, goal-orientation, proactivity, and persistency. This allows analysing and designing complex systems with distributed information, functionalities, and decision processes.

Agent-oriented software development provides concepts, methods, procedures, and tools for the systematic development of multi-agent systems. Agent-orientation is especially suited for problem domains and systems that show logic distribution, structural changes during run-time, complex workflows, or that require comprehensive communication and coordination. Based on an agent oriented solution approach it is possible to develop flexible and adaptive software systems. Such software systems represent the distributed nature of tasks, different views, or contradicting concerns that origin in the problem domain.

3.2 Ontologies

To perform communication and decision tasks related to the knowledge stored in industrial automation systems, computer systems need a representation of the underlying semantic concepts, the relationships among them and their related context. Ontologies can be used for the integration of heterogeneous data sources, to establish an efficient way of information retrieval in industrial automation systems. In most cases, ontology-based information integration approaches are applied for the explicit description of the semantics of data sources. Ontologies provide a description of the terms used in a specific problem area, e.g. the description of terms used in a manufacturing process. They are also used to represent the syntax and the content of messages exchanged among agents. In our approach, ontology is regarded as a set of classes, entities, properties and relationships, describing a standardised terminology [9].

4 An Agent Based Approach for Information Retrieval in Industrial Automation Systems

The aim of this section is to describe the challenges of information retrieval in industrial automation systems and to develop a suitable agent oriented architecture.

In industrial automation systems, distributed and heterogeneous data sources are accessed via specific software tools. Beside the particular applications, accessing data sources also requires appropriate user knowledge and particular working experience for the localisation of the data sources that are relevant for a specific query. Up to now, this remains a manual procedure, because current software systems do not have sufficient understanding of the semantic of the queries and the interrelated data. Users of current software systems often complain that the search results are of a large amount or irrelevant. Compared to software systems, experienced users benefit from their ability to concatenate partial query results to meaningful information. This user specific knowledge is bounded to one person while other users, if any, can participate only in form of rigidly defined search queries. To tackle these challenges, we developed an agent oriented approach that addresses: automating the manual procedure of querying data sources, building flexible search queries and sharing user specific knowledge across the industrial automation system.

At the architectural level, design decisions for the multi-agent system structure focus on the decomposition of system functionalities and the connection to heterogeneous data sources. The approach is based on the comprehensive representation of system

elements including the user behavior through software agents. The basic idea of the agent-oriented approach for information retrieval is to represent search requests and data sources as well as the needed partial steps for solution composition by software agents. An idea is that the desired functionality of each system element is regarded as role. The role is derived from the task, a system element has to fulfill, in the overall structure of the system. Industrial automation systems have a long life time (e.g. up to 30 years) and the information gathering tasks are not directly involved in the technical process, they just provide additional functionalities. Therefore, it is required to support the integration of the multi-agent system architecture into the current industrial automation system without the need to modify the current systems' structure. Agent technology provides powerful methodologies and tools for developing autonomous systems that can handle cooperation, communication and organisation in a distributed environment with different kinds of system elements. The solution scales well as the number of devices in the network is variable because no central point of failure or communication bottleneck exists. It is also a promising software paradigm dealing with system complexity, like distribution or involving and handling human interaction. In our approach we are focusing on the practical issues in gathering information from industrial automation systems. Agent technology in industrial automation benefits from recent research efforts achieved mostly in other disciplines like computer science, cognitive sciences and mathematics.

4.1 Agent-Oriented Problem Analysis

As previously mentioned, the basic idea of the agent-oriented approach is the representation of data sources, as well as the combination of query sequences and user specific search processes by autonomous agents. Multi-agent Systems Engineering (MaSE) is a generic agent-oriented software development methodology [10]. The engineering process of MaSE is based on a top-down approach which consists of several steps that can be performed in an iterative fashion. According to the software engineering process of the MaSE methodology, system goals have to be specified in the analysis step. Three main goals need to be achieved in order to provide an information retrieval system, capable to support distributed heterogeneous data sources in industrial automation systems. These main goals are:

1. Provision of user interaction
2. Processing of queries
3. Integration of data sources

Based on the main goals, the decomposition of system goals is conducted as follows.

The provision of user interaction is a crucial aspect of the information retrieval process in industrial automation systems and represents the main goal. The user needs to formulate his search requests and delegate its solution to the information retrieval system. Therefore the system needs to be capable of managing user requests. Over the course of the processing of a search request, the user also needs to interact with the system in order to narrow down the search scope, to select subsets of search results, or to make manual inputs. These types of interaction are referred to a guide for the user in

composing queries. After a search request has been processed by the system, the query results need to be prepared and presented to the user. This includes the integration of partial results, the filtering of result sets, and the visualisation of the actual query result.

The second main goal is the actual processing of queries in the industrial automation system. The decomposition of this goal is derived straight forward from the basic solution idea. Before a query can be processed, it has to be analysed and decomposed into sub queries and translated into local naming schemes. The system then has to execute the respective steps that constitute the solution for a concrete query.

The integration of data sources represents the third main goal of the system. Since an information retrieval system for industrial automation systems has to deal with various heterogeneous data sources, it must be able to connect to the different data source technologies. Beside the physical data source connection, it must be possible to perform queries on the specific data sources.

4.2 Identification of Roles and Agent Types

Based on the goal hierarchy analysis in the previous section, it is possible to derive four basic roles for the information retrieval systems. Each of these roles is assigned to one of the following agent types.

User Agents
User interaction is provided by user agents (UAs), which offer an integrated view of the systems data sources and guide the user in composing queries. They may suggest a set of possible search targets that helps users to formulate queries. UAs translate the agent communication into a clear text representation form the user is able to understand. As a particular form of a top-level ontology, the user ontology contains the user specific terminology and supports UAs interacting with the user. There is one dedicated UA for each user, which remains in the system, as long as the user is using the system.

Query Management Agents
Supported by the top-level ontology, query management agents (QMAs) work on solving queries by cooperating with UAs and query agents (QAs) within the network. They coordinate the processing and the execution of single search and processing steps, needed to answer a query. Based on the identification of distinct keywords, the QMA decomposes the search query into partial search sequences, which can be delegated. The individual sequences are sent to specialised QAs, which are capable to process partial search sequences. The representation of search queries by QMAs enables the simultaneous processing of and the response to different search queries.

Query Agents
According to the mappings provided by the domain ontologies, query agents work on the creation of query plans which are used by information retrieval agents (IRAs) to query data sources. QAs play the role of query sequence processors. They encapsulate semantic information for the execution of particular query sequences. The knowledge to create a query sequence exists in the respective domain ontology. Thus, the user needs to have less information about the industrial automation system structure and the

processing of queries. A QA decides autonomously within its decision scope whether to create a query plan or a further decomposition and delegation is needed. If the QA is able to process a query sequence completely, it returns the intermediate result to the QMA.

Information Retrieval Agents
Information retrieval agents control the relevant data sources. They are responsible for maintaining a data connection with their corresponding data sources. In addition to the realisation of the technical access to the different data sources, a uniform description of the data source content is needed which translates the ontology-based representation into the target technology. Therefore metadata information is applied. IRAs are introduced for abstracting data storage from data management. IRAs play the role of data source managers. They are used to process the actual query on the data source. They encapsulate data sources within the agent system. Although one IRA is used per one type of data source, IRAs manage metadata to describe the content of data sources and provide a standardised and technologically independent access to data sources. Queries are translated into the specific technological language of the actual data sources. On the way back, query results are returned to the requesting QA. The advantage of the conceptual separation of IRAs and data sources is the flexible extensibility concerning new data sources during runtime - regardless of the type of data source. A new type of data source is represented by the corresponding IRA in the system and is immediately available for use by QAs.

4.3 Ontologies

On top of the challenge of integrating various data source technologies, the system needs to deal with the different naming schemes that are used in the individual data sources. Typically, these various naming schemes result in a high manual effort and require single manual translation and mapping of data types and identifiers. Towards an efficient support of the user, these translation tasks are described by ontologies and need to be performed by the information retrieval system.

We developed a hybrid ontology architecture [11] that contains a top-level ontology which provides a global view on a specific field of knowledge, user ontologies that represent the user vocabulary and several domain ontologies which represent the content and the specific structure of data sources connected to the system. As shown in Fig. 3, domain ontologies contain distinct domain-specific terms like measuring units, semantically equivalent terms and terms with equal meaning in different contexts. In addition, associations among terms are captured. For example, “Production_Job”, is a link between the domain ontologies “Production” and “Laboratory”.

The domain ontologies are interconnected via global logical relations which are part of the top-level ontology. For instance, in view of the chemical process automation, the domain ontology “Production” contains a term “Production_Job” which is logically linked to the equivalent production job in the domain ontology “Laboratory” describing a set of sampling results.

The semantic mapping between the terms, represented by the domain ontologies and the underlying data sources, is provided by metamodels which describe the structure and content of data sources.

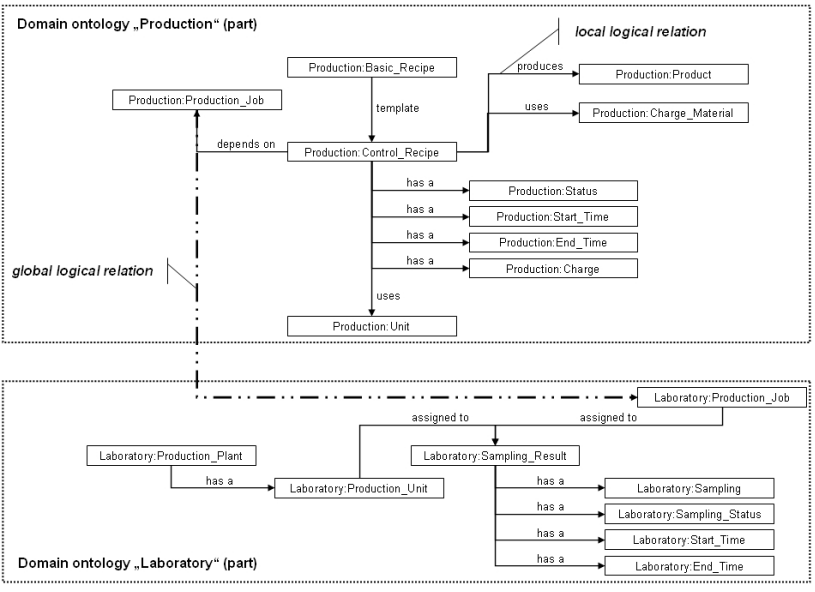


Fig. 3. Illustration of domain ontologies (part)

4.4 Proposed System Architecture

The presented role-based approach and the different agent types support the evolutionary development of the system and favor the flexibility to handle new queries and new data sources during run-time. According to the identification of roles and agent types in the previous section, we propose the following layered architecture of the multiagent system shown in Fig. 4. Specifically, agent functionalities have been dedicated to three functional and cooperative layers:

1. The Data Retrieval Layer, which is responsible for acquiring and preprocessing data from data sources.
2. The Data Processing Layer, where the different search processes are coordinated and information from the Data Retrieval Layer are aggregated.
3. The User Interaction Layer, that implements flexible user interfaces and manages the user interaction by receiving search queries and passing search results to the user.

Inside the multi-agent system, the number and composition of agents depend on the number of queries and vary over time.

Fig. 5 depicts a sequence diagram that illustrates the corresponding search steps which are accomplished by the multi-agent system.

It is worth noting that there can be several instances of agent types and different ontologies involved in the query process.

In step 1, the user formulates its information demand and assigns the query to the system. In step 2, a user agent interprets the query by means of the user ontology and

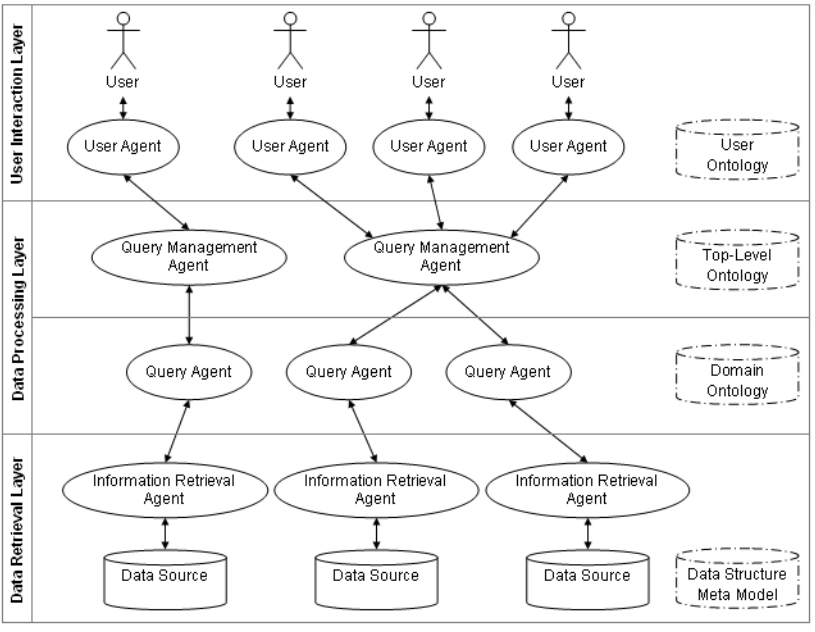


Fig. 4. Agent-oriented system architecture

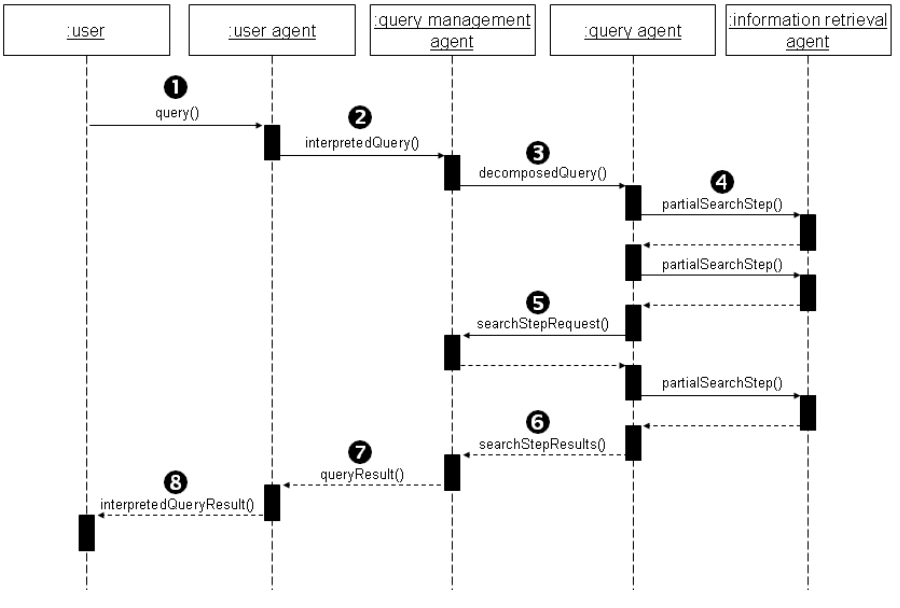


Fig. 5. Sequence diagram of a simplified query process

transmits the query to the query management agent. In step 3, according to the relationships between the terms that are stored in the top-level ontology, the query management agent decomposes the query into manageable domain specific sub queries. In step 4, the query agent builds a search tree, based on the semantic network which is represented by the domain ontologies. After the information retrieval agent has received the sequence of partial search steps, it executes the query on the data sources. In case of step 5 (optional), the query agent requires additional information from the query management agent to conduct the sub queries. In step 6, the query agent delivers the aggregated search step results to the query management agent which, in step 7, combines the results from other domains to constitute the overall query result. In step 8, the user agent delivers the query result to the user.

5 Application Scenario: Query Sequence in the Context of a Chemical Production Environment

In this section, we consider an application scenario regarding the calculation of the material consumption for the production of the chemical product "sulfuric acid" in a time period from 2010/01/01 - 2010/02/28. As described below, the former manual steps performed by the user are now automated by software agents. Fig. 6 illustrates the query process.

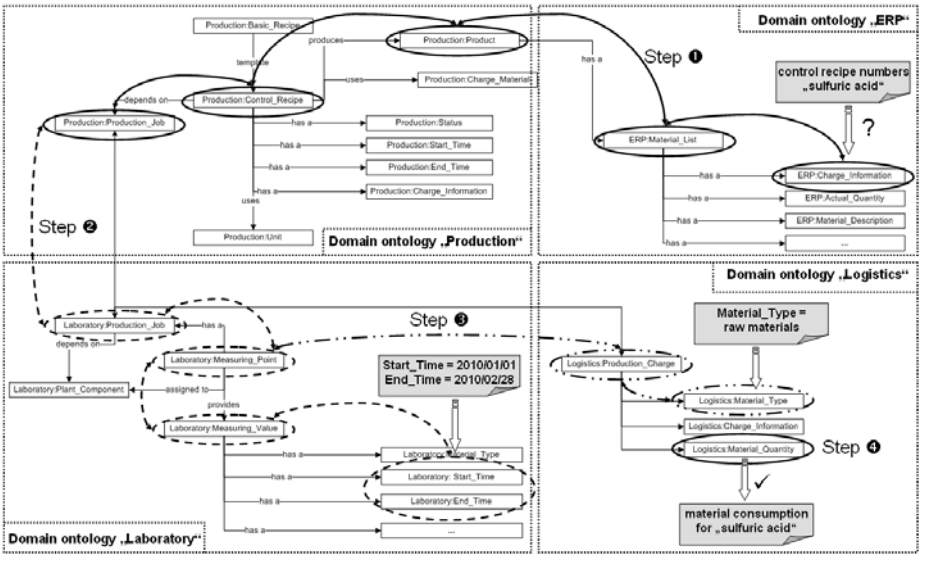


Fig. 6. Query process on domain ontologies (part), performed by software agents

In step 1, the user has to determine the appropriate control recipe numbers for "sulfuric acid" from the data source which contains a list of materials. In step 2, the quantity of determined control recipe numbers is delimited by the given time period from

2010/01/01 - 2010/02/28. In step 3, the user has to decide either to determine raw materials or finished products. In step 4, the determined quantities of used materials or products are added to a total sum, which is the solution to the material consumption for the chemical product “sulfuric acid”.

6 Concluding Remarks and Outlook

Information retrieval in industrial automation systems is a challenging task. It requires very special knowledge from the user about data structures and methodologies for information search. Often, an elaborate manual information search and treatment has to be done by the user. Because of decentralised, heterogeneous data sources a demand on a consistent access to the different data sources exists without dealing with different tools and technologies acting as mediators. This requirement is satisfied by the given agent oriented approach. Based on the systematic analysis and decomposition of queries, the processing of search queries is done by specialised software agents. Agent types are characterised by the encapsulation of the knowledge regarded to process the search queries. This hides the complexity of the system from the user and simplifies the information gathering. Information Retrieval Agents have the appropriate expertise to manage data sources and process queries on them. Current work on information retrieval in industrial automation systems deals with the integration of semantic search capabilities in the search process to offer more flexibility in case of the formulation of search queries to the user. At the same time, a prototypical realisation of the system was evaluated in a real-life chemical production environment.

References

1. Pirttioja, T., Halme, A., Pakonen, A., Seilonen, I., Koskinen, K.: Multi-Agent System Enhanced Supervision of Process Automation, In: IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS’06), pp. 151–156 (2006)
2. Wagner, T.: An agent-oriented approach to industrial automation systems. In: 3rd International Symposium on Multi-Agent Systems, Large Complex Systems, and E-Businesses - MALCEB 2002 (2002)
3. Symeonidis, A.L., Chatzidimitriou, K.C., Kehagias, D., Mitkas, P.A.: An Intelligent Recommendation Framework for ERP Systems. In: Hamza (ed.) IASTED International Conference on Artificial Intelligence and Applications, Innsbruck, pp. 715–720 (2005)
4. Athanasiadis, I.N., Milis, M., Mitkas, P.A., Michaelides, S.C.: A multi-agent system for meteorological radar data management and decision support. Environmental Modelling & Software 24(11), 1264–1273 (2009)
5. Shah, U., Finin, T., Joshi, A., Cost, R., Mayfield, J.: Information retrieval on the semantic web. In: McLean (ed.) Proceedings of the Eleventh International Conference on Information and Knowledge Management, Virginia, pp. 461–468 (2002)
6. Wooldridge, M.J., Jennings, N.R.: Software Engineering with Agents: Pitfalls and Pratfalls. IEEE Internet Computing 3, 20–27 (1999)
7. Nyunt, P., Thein, N.: Software Agent Oriented Information Integration System in Semantic Web. In: 6th Asia-Pacific Symposium on Information and Telecommunication Technologies, Yangon, Myanmar, pp. 266–271 (2005)

8. Jennings, N.R., Wooldridge, M.J.: Agent-Oriented Software Engineering, London, pp. 5–6 (2000)
9. World Wide Web Consortium; Web Standards, http://www.w3.org
10. Wood, M.F.: Multiagent systems engineering: A methodology for analysis and design of multiagent systems. In: Master Thesis, School of Engineering, Air Force Institute of Technology, Ohio (2000)
11. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Huebner, S.: Ontology-based integration of information - a survey of existing approaches. In: Stuckenschmidt (ed.) IJCAI-01 Workshop: Ontologies and Information Sharing, pp. 108–117 (2001)

Part II

Data Mining for Agents

A Data Mining Approach to Identify Obligation Norms in Agent Societies

Bastin Tony Roy Savarimuthu, Stephen Cranefield,
Maryam Purvis, and Martin Purvis

University of Otago, Dunedin, P. O. Box 56, Dunedin, New Zealand
{tonyr,scranefield,tehrany,mpurvis}@infoscience.otago.ac.nz

Abstract. Most works on norms have investigated how norms are regulated using institutional mechanisms. Very few works have focused on how an agent may infer the norms of a society without the norm being explicitly given to the agent. This paper describes how an agent can make use of the proposed norm identification architecture to identify norms. This paper explains how an agent using this architecture identifies one type of norm, an obligation norm. To this end, the paper proposes an Obligation Norm Inference (ONI) algorithm which makes use of association rule mining approach to identify obligation norms.

1 Introduction

Most works on norms in normative multi-agent systems have concentrated on how norms regulate behaviour (e.g. [1]). These works assume that the agent somehow knows (*a priori*) what the norms of a society are. For example, an agent may have obtained the norm from a leader [2] or through an institution that prescribes what the norms of the society should be [3].

Only a few researchers have dealt with how an agent may infer what the norms of a newly joined society are [4]. Recognizing the norms of a society is beneficial to an agent. This process enables the agent to know what the *normative expectation* of a society is. As the agent joins and leaves different agent societies, this capability is essential for the agent to modify its expectations of behaviour, depending upon the society of which it is a part. As the environment changes, the capability of recognizing a new norm helps an agent to derive new ways of achieving its intended goals. Such a norm identification mechanism can be useful for software agents that need to adapt to a changing environment. In open agent systems, instead of possessing predetermined notions of what the norms are, agents can infer and identify norms through observing patterns of interactions and their consequences. For example, a new agent joining a virtual environment such as Second Life [5] may have to infer norms when joining a society as each society may have different norms. It has been noted that having social norms centrally imposed by the land owners in Second Life is ineffective and there is a need for the establishment of community driven norms [6]. When a community of agents determines what the norm should be, the norm can evolve over time. So, a new agent joining the society should have the ability to recognize the changes to the norms.

This work aims to answer the question of how agents infer norms in a multi-agent society. To that end, we propose an internal agent architecture for norm identification. The

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 43–58, 2010.

architecture is based on observation of interactions between agents. It enables an autonomous agent to identify the obligation norms in a society using the Obligation Norm Inference (ONI) algorithm presented here. Using an auction example, we demonstrate how an agent makes use of the norm identification framework.

The paper is organized as follows. Section 2 provides a background on normative multi-agent systems (NorMAS). Section 3 provides an overview of the norm identification framework. Section 4 describes the components of the framework in the context of an e-market scenario (buying and selling goods). Section 5 provides a discussion on the work that has been achieved and the issues that need to be addressed in the future. Concluding remarks are presented in section 6.

2 Background

Due to multi-disciplinary interest in norms, several definitions for norms exist [4]. The definition of normative multi-agent systems as described by the researchers involved in the NorMAS 2007 workshop is as follows [7]. *A normative multi-agent system is a multi-agent system organized by means of mechanisms to represent, communicate, distribute, detect, create, modify and enforce norms, and mechanisms to deliberate about norms and detect norm violation and fulfillment*. Researchers in multi-agent systems have studied how the concept of norms can be applied to artificial agents. Norms are of interest to multi-agent system (MAS) researchers as they help in sustaining social order and increase the predictability of behaviour in the society. Researchers have shown that norms improve cooperation and collaboration [8,9].

Research in normative multi-agent systems can be categorized into two branches. The first branch focuses on normative system architectures, norm representations, norm adherence and the associated punitive or incentive measures. Several architectures have been proposed for normative agents (refer to [10] for an overview). The second branch of research is related to emergence of norms. Several researchers have worked on both prescriptive (top-down) and emergent (bottom-up) approaches to norms (refer to [11]).

The work reported in this paper falls under the bottom-up approach to the study of norms. Many researchers in this approach have experimented with game-theoretical models for norm emergence [12,8]. Agents using these models learn to choose a strategy that maximizes utility. The agents in these works do not possess the notion of "normative expectation". Many research works assume that norms exist in the society and the focus is on how the norms can be regulated in an institutional setting such as electronic institutions [13]. Very few have investigated how an agent comes to know the norms of the society.

Our objective in this work are two-fold. First, we propose an architecture which can be used by individual agents to identify what the norms of the society are. The architecture is based on observation of agent interactions and the inference mechanism considers "signalling" (positive and negative) to be a top-level construct for identifying potential norms when the norm of a society is being shaped. We note that a sanction may not only imply a monetary punishment, it could also be an action that could invoke emotions (such as an agent yelling at another potentially invoking shame or embarrassment on another agent), which can help in norm spreading. Agents can recognize

such actions based on their previous experience. Second, based on association rule mining [14], we propose an algorithm for norm inference, called the Obligation Norm Inference (ONI) algorithm, which can be adapted by an autonomous agent for flexible norm identification. The ONI algorithm identifies potential obligation norms. An example obligation for an agent participating in an online auction is to pay for the item it has won ($O_{A,B}(p|w)$)[1]. When obligations are violated, sanctions can be imposed on the violating agent by other agents. In this architecture we assume that agents have the ability to recognise sanctions. The rest of the paper describes how an observer agent will be able to infer a norm of the society.

3 Overview of the Norm Identification Architecture

In this section we provide an overview of the norm identification framework (called the norm engine) that we propose for an agent to infer norms in the agent society in which it is situated. The norm identification framework takes into account the social learning theory [15] that suggests that new behavior can be learnt through the observation of punishment and rewards. Figure 1 shows the architectural diagram of the norm identification framework. An agent's norm engine is made up of several components. The circles represent information storage components. The rounded boxes represent information processing components, and the diamonds represent decision making components, and the lines represent the flow of information between the components.

An agent employing this architecture follows a four-step process.

Step 1: An agent actively perceives the events in the environment in which it is situated.

Step 2: When an agent perceives an event, it stores the event in its belief base. The events observed by an observer are of two types: regular events and signalling events. In the context of an auction, a regular event is an event, such as an agent bidding for an item and winning the item. Special events are signalling events that agents understand to be either encouraging or discouraging certain behaviour. For example when an agent wins a particular item but does not pay in accordance to the norm of the society, the agent can be sanctioned by the auction authorities[2]. Let us assume that the signal in this context is the occurrence of the shaming event which is a form of a sanction. In this work we assume that an agent has the ability to recognize signalling events based on its previous experience.

Step 3: When a special event occurs, the agent stores the special event in the special events base. It should be noted that all events are stored in an agent's belief base but only special events are stored in the special events base.

Step 4: If the perceived event is a special event an agent checks if there exists a norm in its *personal norm* (*p-norm*) base or the *group norm* (*g-norm*) base. An agent may

[1] This is to be read as A is obliged to B to bring about *p* given *w* has occurred.

[2] For example, when the winner who is obliged to submit a cheque immediately after winning does not submit the auctioneer may denounce the winner, black list the winner or even report it to authorities such as the Police.

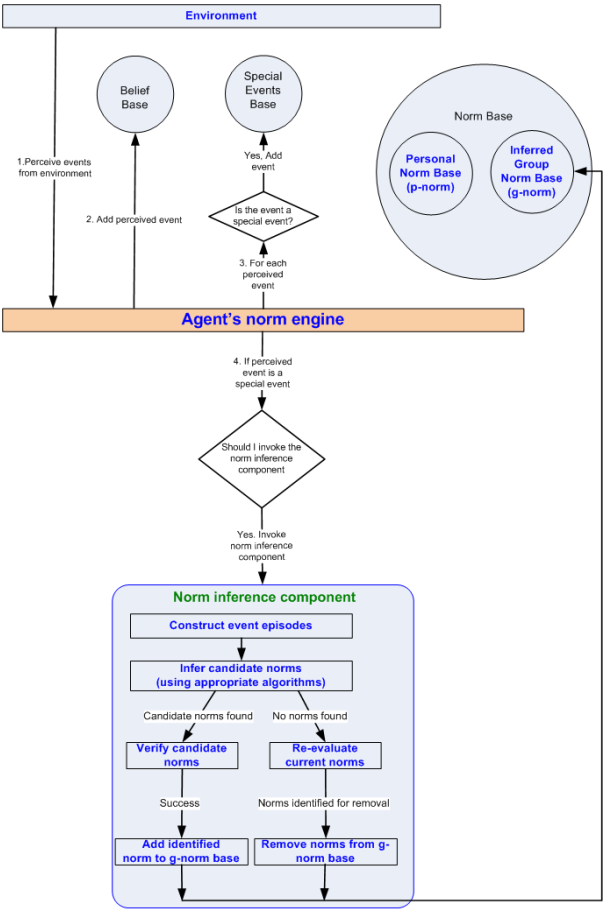


Fig. 1. Norm identification architecture of an agent

possess some p-norms[3] based on its past experience or preference. A *p-norm* may vary across agents, since a society may be made up agents with different backgrounds and experiences. A *g-norm* is a norm which an agent infers, based on its personal interactions as well as the interactions it observes in the society. An agent infers g-norms using the norm inference component.

When a special event occurs an agent may decide to invoke its norm inference component to identify whether a previously unknown norm may have resulted in the occurrence of the special event. In the context of the auction scenario, an agent observing a sanctioning event may invoke its norm inference component to find out what events that

[3] A *p-norm* is the personal value of an agent. For example an agent may consider that bidding for an item of the same type that the bidder has won previously is an action that should be prohibited in a society. This personal value may not be shared by the agents in a society.

had happened in the past (or that had not happened in the past) may have triggered the occurrence of the special event[4]. The invocation of the norm inference component may result in the identification of a *g-norm*, in which case it is added to the *g-norm* base.

An agent, being an autonomous entity, can also decide not to invoke its norm inference component for every occurrence of a special event but may decide to invoke it periodically. When it invokes the norm inference component, it may find a new *g-norm* which it adds to its *g-norm* base. If it does not find a *g-norm*, the agent may change some of its norm inference parameters and repeat the process again in order to find a *g-norm* or may wait to collect more information.

At regular intervals of time an agent re-evaluates the g-norms it currently has, to check whether those norms hold. When it finds that a *g-norm* does not apply (e.g. if it does not find any evidence of sanctions), it deletes the norm from the *g-norm* base. The operational details of the norm inference component are explained in Section 4.3. What an agent does with the norms once it has inferred the norms is out of the scope of this paper.

4 Obligation Norm Identification

In this section we explain how obligation norms can be identified using the framework described in Section 3. Obligation norms can be identified by an agent in our architecture using obligation norm inference (ONI) algorithm proposed here. First, we describe the domain in which an obligation norm is identified. Second, we describe the event storage components of the architecture (steps 2 and 3 of Figure 1). Third, we describe the ONI algorithm.

4.1 e-Market Scenario

Let us assume that agents participate in an electronic market such as an auction in a virtual environment (e.g. Second Life). A new agent joining a society may not be aware of the norms associated with buying and selling goods in an electronic market. For example, in one society the winning bidder may be obliged to deposit the money within a day while the norm in another society could be that the winner may be obliged to deposit the money within an hour. Failure to fulfill the obligation may result in a sanction.

4.2 Event Storage Components

Let us assume that an agent is situated in an auction house where multiple auctions take place at the same time (the electronic market scenario [16]). Let us also assume that a new agent is not aware of the norms of the auction house. In this architecture an agent would first observe the interactions that occur between the agents in the society. The interactions could be of two types. The first type of interaction is the one in which the

[4] Prohibition norms may be identified by inferring the relevant events that happened in the past. For identifying obligation norms the agent may have to reason about what events that did not happen in the past are the likely reason for a sanction (i.e. not fulfilling an obligation).

agent itself is involved and is called a *personnel interaction* (e.g. bidding). The second type of interaction is an interaction between other agents that is observed by an observer agent, referred to as an *observed interaction*. The agent records these interactions (as events) in its belief base. An agent in the society can assume one or more of the following three roles: a participant (P) that is involved in a personal interaction, an observer (O) and a signaller (S).

In the auction scenario, the agent is aware of the actions performed by an agent, which are bidding for an item (*bid*), winning an item (*win*), paying for a particular item (*pay*) and receiving the item (*receive*). The agent also has the ability to recognize a signalling action such as *yell* or *shame*[5]. Signalling events can either be positive (e.g. rewards) or negative (sanctions)[6].

Let us assume that a new agent (an observer) is situated in the auction. The observer records interactions that occur in the context of the auction. Let us assume that a sanctioning event occurs. Even though an observer may know that a sanctioning event has occurred, it may not know the exact reason for sanctioning (i.e. it may not know the norm because it only observes a sequence of actions and the agent does not know which of the events that happened in the past or the absence of which event(s) triggered the sanction). It will infer norms using the norm inference mechanism.

An event that is perceived by an agent consists of an event index, an observed action, and the agent(s) participating in that event. For example an agent observing an agent bidding will represent this as *happens(1,bid(X,item1,Y)*. This implies the observer believes that the first event was generated by agent X which bids for item1 to agent Y. A sample representation of events observed by an agent is given in list (1). An agent situated in an environment can sense these actions through observation or through action logs that may be available[7].

$$\begin{pmatrix} happens(1, bid(C, item1, B) \\ happens(2, bid(A, item1, B) \\ happens(3, win(A, item1, B) \\ happens(4, bid(A, item2, C) \\ happens(5, win(A, item2, C) \\ happens(6, sanction(B, A)) \\ happens(7, pay(A, item2, C) \\ happens(8, receive(A, item2, C) \end{pmatrix} \tag{1}$$

An agent records these events in its belief base. Event 6 is a sanctioning event, where agent B sanctions agent A. The reason for the sanction is that agent A failed to pay for the item it has bought. For an observer (the agent) it may not be possible to know the reason

[5] We assume that sanctioning events such as an agent yelling at another agent for violating a norm or an agent publicly shaming another agent by announcing that the agent is blacklisted are observable. We note that recognizing and categorizing a sanctioning event is a difficult problem. In our architecture we assume such a mechanism exists (e.g. based on an agent's past experience).

[6] In this work we focus on the negative signals (i.e. sanctions).

[7] For example, in Massively Multi-Player Online Role Playing Games (MMORPGs), the logs of user interactions may be available for the observer through chat channels [7].

for this sanction unless it was specified *a priori* by the agent's designer. In open agent societies such as open e-markets, the norms of the society may not be known to an agent ahead of time. Additionally, the norms may evolve over time. In order to infer a norm of the society the agent will make use of the norm inference mechanism proposed here.

The agents have a filtering mechanism, which identifies signalling events and stores them in the special events base. It should be noted that special events, such as *yell* and *shame*, are categorized by an agent as sanctioning events and they are stored in the special events base under the *sanction* event.

4.3 Norm Inference Component

An agent may choose to invoke its norm inference component based on its preference. For example, it can invoke the component every time it perceives a signalling action, or it may invoke this component periodically.

The norm inference component of an agent is made up of two sub-components. The first sub-component makes use of the Obligation Norm Inference (ONI) algorithm to generate candidate obligation norms. Candidate obligation norms are the norms that an agent considers to be potential candidates to become the norms in a society. The second sub-component is the norm verification component, which verifies whether a candidate norm can be identified as a norm in the society.

This sub-section is organized as follows. Firstly we explain the parameters of the ONI algorithm. Secondly we describe the internal details of ONI algorithm using the auction example.

4.3.1 Definitions of Parameters Used in the Algorithm

The parameters that are used in the Obligation Norm Inference algorithm are explained below.

History Length (HL): An agent keeps a history of the observed interactions for certain window of time. This period of time is represented by the History length (HL) parameter. For example, if HL is set to 20, an agent will keep the last 20 events it observes in it its memory.

Event Sequences (ES): An event sequence is the record of actions that an agent observes in the history. For example the event sequence observed by an agent where HL=8 is given in list (1).

Special Events Set (SES): An agent has a set of events it identifies to be special. These events are the signalling events. For example, the special event set can contain events such as yell (SES = $\{yell\}$). An agent also has the capability to categorize events into two types, sanctions and rewards. For example the action mentioned above can be identified as a sanctioning action.

Unique Events Set (UES): This set contains the number of distinct events that occur within a period of time. For example, a unique events set for the example given in list (1) contains the following events[8], UES = $\{bid,win,pay,receive,\ sanction\}$.

[8] Assume that event occurrences can be modelled as simple propositions.

Occurrence Probability (OP): The occurrence probability of an event E is given by the following formula.

$$OP(E) = \frac{\text{Number of occurrences of E}}{\text{Total number of events in ES}}$$

Window size (WS): When an agent wants to infer norms, it looks into its history for a certain number of recent events. For example, if the WS is set to 3, an agent constructs an *event episode* (EE) with the last three events that were exchanged between agents involved in the interaction (e.g. a pair of agents: the buyer and the seller). Construction of event episodes is described in the next sub-section. It should be noted that an EE is a subsequence[9] of ES.

Norm Identification Threshold (NIT): When coming up with candidate norms, an agent may not be interested in events that have a lower probability of being a norm. For example, if an agent sets NIT to be 50 (in a scale from 0 to 100), it indicates it is interested to find all sub-episodes[10] of an event episode that have a 50% chance of being a candidate norm. The algorithm uses the NIT on three occasions. As the values of NIT for each of the occasions can be varied by an agent, there are three variables which are NIT_a, NIT_b and NIT_c.

Norm Inference Frequency (NIF): An agent may choose to invoke a norm inference component every time it observes a special event, or it may invoke the component periodically. An agent has a parameter called the norm inference frequency (NIF) that specifies what the time interval between two invocations of the norm inference component are. An agent, being an autonomous entity, can change this parameter dynamically. If it sees that the norm in a society is not changing, then it can increase the waiting period for the invocation of the norm inference component. Alternatively, it can reduce the time interval if it sees the norm is changing.

4.3.2 Obligation Norm Inference (ONI) Algorithm

4.3.2.1 Overview of the Algorithm. There are four main steps involved in the Obligation Norm Inference algorithm (see algorithm 1). First, event episodes of a certain length are extracted from event sequences that an agent observes that are exchanged between agents. These event episodes are stored in the event episode list (EEL). Second, based on the events in the special event set (e.g. sanctioning events), the event episodes in EEL are separated into two lists. The first list contains all event episodes that contain at least one sanctioning event called the Special Event Episode List (SEEL). The second list contains all event episodes that do not contain sanctioning events called the Normal Event Episode List (NEEL). Third, using SEEL, all sub-episodes which have occurrence probabilities greater than or equal to NIT_a are extracted and stored in Norm-Related Event Episode List (NREEL) based on a modified version of the WINEPI algorithm [17]. Fourth, for each event episode in NREEL, all supersequences whose

[9] A subsequence is a sequence that can be generated from a sequence by removing certain elements from the sequence without altering the order of the elements in the sequence. For example, “anna” is a subsequence of “banana”. Conversely, one of the supersequences of “anna” is “banana”.

[10] A sub-episode is a subsequence of an event episode.

occurrence probabilities are greater than or equal to NIT_b are extracted and stored in a temporary list called *tempEEList*. Based on the supersequences stored in *tempEEList*, the modified version of the WINEPI algorithm can identify all permutations of supersequences whose occurrence probabilities are greater than or equal to NIT_c which are stored in Candidate Obligation Norm List (CONL). These four steps are explained in detail in the following sub-sections.

Algorithm 1. Obligation Norm Inference algorithm (main algorithm)

```
1 begin
2     Create event episodes list (EEL);
3     Create special event episodes list(SEEL) and normal event episodes list
      (NEEL);
4     Extract norm related event episodes list (NREEL) from SEEL;
5     Create Candidate Obligation Norm List (CONL) using NEEL and NREEL ;
      /* Algorithm 2 */
6 end
```

4.3.2.2 Creating Event Episodes. An agent records other agents' actions in its belief base. We call the sequence of events that were recorded in the belief base *event sequences* (ES). Let us assume that there are three agents A, B and C, participating in an auction as given in list (1) and an observer D. Agent C bids for item1 from agent B. Agent A bids for item1 from agent B. Agent A wins item1. Agent A then bids for item2 from agent C. Agent A wins item2. Agent A is sanctioned by agent B. Agent A pays agent C for item2. Agent A receives item2 sent by agent C[11]. An agent has a certain history length (HL). An agent at any point of time stores the history of observed interactions for the length equal to HL. When the norm inference component is invoked, the agent extracts *n* events that happened between a pair of agents from the recorded history (event sequences (ES)) where *n*=*WS*. We call the retrieved event sequence the *event episode* (EE). A sample event episode from an observer's view-point (agent D) is given below. The left hand side of the arrow indicates that the agents involved in the interaction are A and B. The right hand side of the arrow contains the event episode. A hyphen separates one event from the next.

$$\{A, B\} \rightarrow (happens(2, bid(A, item1, B) - happens(3, win(A, item1, B) - happens(6, sanction(B, A)))$$

Based on the what an agent observes (e.g. the event sequence given in list (1)), the observer may assume that something that agent A did in the past may have caused the sanction. It could also be the failure of agent A to perform certain action(s) might have caused a sanction. In this work we concentrate on the latter[12]. Agent D then extracts the sequence of events (the *event episode*) that took place between A and B based on the event sequence stored in its history. To simplify the notation here, only the first letter

[11] Note that the representation of these actions are from one agent's point of view (e.g. agent A's point of view). Therefore actions such as *send* which is from the viewpoint of agent C are not modelled, but a related action i.e. *receive* is modelled from A's view-point.

[12] In previous work we have demonstrated how the former can be identified [18].

of each event will be mentioned from here on (e.g. *b* for *bid*) and also the agent names are omitted. As the sequence caters for temporal ordering of events, the event ids are omitted. Thus the event episode for interactions between agents A and B shown above will be represented as

$$\big(\{A, B\} \rightarrow b - w - s\big)$$

The following list (2) shows a sample event episode list (EEL) that contains ten events occurring between a pair of agents that are observed by an agent where WS=5. Note that the Unique Event Set (UES) in this case include events $\{b,w,p,r,s\}$ which stand for $\{bid,win,pay,receive,sanction\}$ respectively.

$$\begin{pmatrix} \{A, B\} \rightarrow (b - w - s) \\ \{C, D\} \rightarrow (w - p - r) \\ \{E, F\} \rightarrow (b - w - p - r - b) \\ \{G, H\} \rightarrow (p - r - b - w) \\ \{I, J\} \rightarrow (r - b - b - w - s) \\ \{K, L\} \rightarrow (b - w - p) \\ \{M, N\} \rightarrow (r - b - w - p - r) \\ \{O, P\} \rightarrow (b - w - p - r) \\ \{R, S\} \rightarrow (r - b - w - s) \\ \{T, U\} \rightarrow (r - b - w - p) \end{pmatrix} \tag{2}$$

4.3.2.3 Creating Special and Normal Event Episode Lists. Note that some event episodes in EEL have sanctions as one of the events. The agent identifies the sanction events from the special events set (SES). Using EEL, an agent creates two lists for further processing, one with event episodes that contain a sanctioning event and the other containing event episodes without sanctions. The list that contains event episodes with sanctioning events is called the special event episode list (SEEL). The other list is called the normal event episode list (NEEL).

The SEEL obtained from EEL is given in the left in (3). NEEL has the remaining episodes that do not contain a sanctioning action (shown in the right of (3)).

$$\begin{pmatrix} (b - w - s) \\ (r - b - b - w - s) \\ (r - b - w - s) \end{pmatrix}, \begin{pmatrix} (w - p - r) \\ (b - w - p - r - b) \\ (p - r - b - w) \\ (b - w - p) \\ (r - b - w - p - r) \\ (b - w - p - r) \\ (r - b - w - p) \end{pmatrix} \tag{3}$$

4.3.2.4 Generating the Norm Related Event List (NREEL). From the SEEL, an agent can identify events that have the potential to be associated with sanctions. For example, from the SEEL shown in the left of (3), the agent may infer that the sub-episodes *b-w*, *b*, or *w* could be the reason for a sanction as they occur in all the event episodes in SEEL. In the case of prohibition norms the events that precede a sanction can be potentially linked to sanction due to causality. In the case of obligation norms, it is the absence of an event or a sequence of events that might be the cause of the sanction. In both these

types of norms, the agent has to identify the sequences of events that occur frequently before the occurrence of a sanctioning action. In the case of a prohibition norm the frequency of occurrence may correlate with norm identification (reported in previous work [18]). In the case of an obligation norm, the agent first has to find the frequently occurrence sequence(s), which are then stored in the norm-related event list (NREEL). Let us refer to an event episode in NREEL as α. Second, an agent has to identify all the supersequences of α in NEEL whose occurrence probability is greater than or equal to NIT_a, which is added to the candidate obligation norm list (CONL). The construction of NREEL is discussed in this sub-section and the construction of CONL is discussed in the next sub-section.

In order to identify these norm-related events the agent uses a modified version of the WINEPI algorithm [17], an association rule mining algorithm[13]. The modification, reported in previous work [18], can identify candidate norms that are obtained by considering "permutations with repetition" when constructing sub-episodes. Based on the SEEL, an agent can generate the NREEL. Expression (4) shows the SEEL on the left of the arrow and the NREEL generated from the SEEL on the right of the arrow when NIT_a is set to 0. The occurrence probability of an event episode in NREEL is given in square brackets. When NIT_a is set to 0, all possible subsequences of event episodes in SEEL are generated. When NIT_a is set to 100% the algorithm identifies the following norm-related event episode list {*b-w,b,w*}. An agent, being an autonomous entity, can vary the NIT_a parameter to identify the norm-related events. Note that if an event episode is frequent, all its subsequences are also frequent. For example if *b-w* appears 100% of the time (i.e. the occurrence probability is 1), all its subsequences also appear 100% of the time.

$$\begin{pmatrix} (b-w-s) \\ (r-b-b-w-s) \\ (r-b-w-s) \end{pmatrix} \rightarrow \begin{pmatrix} (b-w)[1] \\ (b)[1] \\ (w)[1] \\ (r-b-w)[.66] \\ (r-b)[.66] \\ (r-w)[.66] \\ (r)[.66] \\ (r-b-b-w)[.33] \\ (r-b-b)[.33] \\ (b-b-w)[.33] \\ (b-b)[.33] \end{pmatrix} \tag{4}$$

[13] Association rule mining [14] is one of the well known fields of data mining where relationships between items in a database are discovered. For example, interesting rules such as 80% of people who bought diapers also bought beers can be identified from a database.Some well known algorithms in the data mining field can be used for mining frequently occurring episodes (i.e mining association rules) [19,17]. A limitation of the well-known Apriori [19] algorithm is that it considers combinations of events but not permutations (e.g. it does not distinguish between event sequences *b-w* and *w-b*). WINEPI [17] addresses this issue, but it lacks support for identifying sequences that are resultants of permutations with repetition (e.g. from sub-episodes of length one, e.g. *b* and *w*, the algorithm can generate sub-episodes of length two which are *bw* and *wb*, but not *bb* and *ww*). Permutations with repetition are important because there could be a norm which sanctions an agent from performing the same action twice.

4.3.2.5 Identifying Candidate Obligation Norm List (CONL). The pseudo code for generating CONL is given in Algorithm 2. In order to identify the obligation norms, the agent has to identify those supersequences in NEEL, that contain the event episodes in NREEL whose occurrence probabilities are greater than or equal to NIT_b. These supersequences are stored in a list (*tempEEList* in this case).

Based on the supersequences stored in *tempEEList*, the algorithm (previous work [18]) can identify all permutations of supersequences whose occurrence probabilities are greater than or equal to NIT_c. Such supersequences are stored in the candidate obligation norm list (CONL).

For example, let us consider an event epsiode *b-w* is the only event episode stored in the NREEL list. Assume this NREEL is the input to Algorithm 2 and the NIT_b is set to 50%. Expression (5) shows the NEEL on the left of the arrow and the *tempEEList* that is generated from the NEEL on the right. Note that the NEEL on the left contains seven event episodes but *tempEEList* contains six out of seven event episodes that contain *b-w*. These six event episodes are supersequences of *b-w*.

Algorithm 2. Pseudocode to create candidate obligation norm list (CONL)

Input: Norm-Related Event Episode List (NREEL), Normal Event Episode List (NEEL), Norm Identification Threshold (NIT)
Output: Candidate Obligation Norm List (CONL)

```
1  CONL = ∅;
2  for an event episode NREE ∈ NREEL do
3      tempEEList= ∅;
4      tempCounter = 0;
5      occurrenceCounter = 0;
6      foreach event episode EE ∈ NEEL do
7          occurrenceCounter++;
8          if EE is a supersequence of NREE then
9              Add EE to tempEEList;
10             tempCounter++;
11         end
12     end
13     OP(tempEEList) = tempCounter/occurrenceCounter;
14     if OP(tempEEList) ≥ NIT_b then
15         Use modified WINEPI algorithm to extract all candidate obligation
           norms (Input: tempEEList, Unique Event Set (UES), Window Size(WS),
           Norm Inference Threshold (NIT_c), Output: Candidate norms);
16         Add candidate obligation norms to CONL;
17     end
18     return CONL;
19 end
```

$$\begin{pmatrix} (w-p-r) \\ (b-w-p-r-b) \\ (p-r-b-w) \\ (b-w-p) \\ (r-b-w-p-r) \\ (b-w-p-r) \\ (r-b-w-p) \end{pmatrix} \rightarrow \begin{pmatrix} (b-w-p-r-b) \\ (p-r-b-w) \\ (b-w-p) \\ (r-b-w-p-r) \\ (b-w-p-r) \\ (r-b-w-p) \end{pmatrix} \tag{5}$$

From *tempEEList* the CONL can be generated. The left hand side of expression (6) shows the *tempEEList*. The right hand side of expression (6) contains all permutations of supersequences of *b-w* that can be obtained from *tempEEList* and their occurrence probabilities in *tempEEList* (in square brackets).

$$\begin{pmatrix} (b-w-p-r-b) \\ (p-r-b-w) \\ (b-w-p) \\ (r-b-w-p-r) \\ (b-w-p-r) \\ (r-b-w-p) \end{pmatrix} \rightarrow \begin{pmatrix} (b-w-p)[0.83] \\ (b-w-r)[0.5] \\ (b-w-p-r)[0.5] \\ (r-b-w)[0.5] \\ (b-w-b)[0.16] \\ (b-w-p-b)[0.16] \\ (b-w-p-r-b)[0.16] \\ (p-r-b-w)[0.16] \\ (p-b-w)[0.16] \end{pmatrix} \tag{6}$$

Assuming that NIT_c is set to 50%, the supersequences that will be identified as CONL are *(b-w-p,b-w-r,b-w-p-r,r-b-w)* whose occurrence probabilities are *(0.83,0.5,0.5,0.5)* respectively. As the occurrence probabilities of *(b-w-b,b-w-p-b,b-w-p-r-b,p-r-b-w,p-b-w)* are less than NIT_c, these are not included in the CONL. Note that the modified WINEPI algorithm is used twice, first time to obtain the NREEL from the SEEL (not shown here) and the second time for obtaining the CONL from the NEEL using the NREEL (line 15 of Algorithm 2).

For every event episode in the NREEL, a new CONL is generated. Having compiled a set containing candidate obligation norms, the agent passes this information to the norm verification component to identify norms. This process is iterated until there are no elements in NREEL. The norm verification process is explained in the next sub-section.

4.4 Norm Verification

In order to find whether a candidate norm is a norm of the society, the agent asks another agent in its proximity. This happens periodically (e.g. once in every 10 iterations). An agent A can ask another agent B, by choosing the first candidate norm (say *b-w-p* for which it has a higher occurrence probability) and asks B if it knows whether the obligation norm $O_{X,Y}(p|(b-w))$ is a norm of the society (i.e. an agent is obliged to pay after bidding and winning). If the response is affirmative, A stores this norm in its set of *identified norms*. If not, A moves on to the second candidate norm in its list[14].

[14] Other alternative mechanisms are also possible. For example, an agent could ask for all the candidate norms from another agent and can compare them locally.

In the case of the running example, the supersequence *b-w-p* is chosen to be communicated to the other agent. It asks another agent (e.g. an agent who is the closest) whether it thinks that the given candidate norm is a norm of the society. If it responds positively, the agent infers $O_{X,Y}(p|(b-w))$ to be a norm. If the response is negative, this norm is stored in the bottom of the candidate norm list.

It then asks whether the failure to fulfill the obligation norm $O_{X,Y}(r|(b-w))$ is the reason for the sanction. Otherwise, the next event episode in the candidate norm list is chosen for verification. This process continues until a norm is found or no norm is found from the event episodes in the candidate norm list. If no norm is found, the agent considers the next event episode in the NREEL and uses Algorithm 2 to identify candidate obligation norms. This process continues until there are no event episodes in the NREEL. Even in the case of identifying a candidate norm, the agent continues the process to identify any co-existing norms.

Note that an agent will have two sets of norms: candidate norms and identified norms. Expression (7) shows the two sets of norms, the candidate norms on the left of the arrow and the identified norms on the right. Once an agent identifies the norms of the system and finds that the norms identified have been stable for a certain period of time, it can forgo using the norm inference component for a certain amount of time (based on the norm inference frequency (NIF)). It invokes the norm inference component periodically to check if the norms of the society have changed, in which case it replaces the norms in the identified list with the new ones (or deletes the norms which are no more applicable).

$$\begin{pmatrix} (b-w-p) \\ (b-w-r) \\ (b-w-p-r) \\ (r-b-w) \end{pmatrix} \rightarrow \big(b-w-p\big) \tag{7}$$

5 Discussion

The main contributions of the paper are two-fold. First, the issue of norm identification has not been dealt with by many researchers in the field of normative multi-agent systems. To this end, in this paper we have proposed an architecture for norm identification and have demonstrated how one type of norm - the obligation norm can be identified by an agent. Secondly, we have proposed the Obligation Norm Inference (ONI) algorithm, an algorithm based on data mining, that can be used to generate candidate obligation norms. Using a simple example, we have demonstrated how the norm inference mechanism works. An adaptive agent employing the norm inference mechanism can infer obligation norms by varying different parameters of the algorithm.

We believe this architecture can be used in several settings apart from e-commerce environments. For example, the norm identification architecture can be used to infer norms in Massively Multi-player Online Role Playing Games (MMORPGs) such as World of Warcraft (WoW). Players involved in massively multi-player games perform actions in an environment to achieve a goal. They may play as individuals or in groups. When playing a cooperation game (e.g. players forming groups to slay a dragon), individual players may be able to observe proscriptions of actions (prohibition norms) and

obligations that need to be satisfied (obligation norms). The normative architecture proposed in this paper can be used to identify norms that are being formed. For example a norm could be that a player who has helped another player twice to escape from a dragon expects the other player to help him escape from the dragon if the need arises. This norm may not be part of the protocol defined for playing the game but may evolve during the game. Such a norm can be identified by this mechanism.

An interesting addition to this work is on identifying conditional norms. For example, in one society, the norm associated with the deadline for the payment (i.e. obligations with deadlines as in [20]) may be set to 120 minutes after winning the item. Depending upon what an agent has observed, agents may have subtly different norms (e.g. one agent may notice that p follows w after an average of 100 minutes while another may notice this to happen after 150 minutes). Both these agents could still infer the obligation norm but the deadlines they had noticed can be different. Another interesting avenue for research is to investigate employing string based pattern matching algorithms used in the field of bio-informatics to extract interesting sequences and missing sequences.

6 Conclusion

This paper addresses the question of how obligation norms can be identified in an agent society. To this end, this paper uses the norm inference architecture for identifying obligation norms. This paper proposes Obligation Norm Inference (ONI) algorithm. An agent that employs ONI algorithm makes use of data mining approach to infer obligation norms. This has been demonstrated in the context of a simple e-market scenario.

References

1. López y López, F.: Social Powers and Norms: Impact on Agent Behaviour. PhD thesis, Department of Electronics and Computer Science, University of Southampton, United Kingdom (2003)
2. Boman, M.: Norms in artificial decision making. Artificial Intelligence and Law 7(1), 17–35 (1999)
3. Vázquez-Salceda, J.: The role of norms and electronic institutions in multi-agent systems applied to complex domains the harmonia framework. AI Communications 16(3), 209–212 (2003)
4. Andrighetto, G., Conte, R., Turrini, P., Paolucci, M.: Emergence in the loop: Simulating the two way dynamics of norm innovation. In: Boella, G., van der Torre, L., Verhagen, H. (eds.) Normative Multi-agent Systems, Dagstuhl Seminar Proceedings, vol. 07122, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl (2007)
5. Rymaszewski, M., Au, W.J., Wallace, M., Winters, C., Ondrejka, C., Batstone-Cunningham, B., Rosedale, P.: Second Life: The Official Guide. SYBEX Inc., Alameda (2006)
6. Stoup, P.: Athe development and failure of social norms in second life. Duke Law Journal 58(2), 311–344 (2008)
7. Boella, G., Torre, L., Verhagen, H.: Introduction to the special issue on normative multiagent systems. Autonomous Agents and Multi-Agent Systems 17(1), 1–10 (2008)
8. Shoham, Y., Tennenholtz, M.: Emergent conventions in multi-agent systems: Initial experimental results and observations. In: Proceedings of Third International Conference on Principles of Knowledge Representation and Reasoning, pp. 225–231. Morgan Kaufmann, San Mateo (1992)

9. Walker, A., Wooldridge, M.: Understanding the emergence of conventions in multi-agent systems. In: Lesser, V. (ed.) Proceedings of the First International Conference on Multi–Agent Systems, pp. 384–389. MIT Press, San Francisco (1995)
10. Neumann, M.: A classification of normative architectures. In: Proceedings of World Congress on Social Simulation (2008)
11. Savarimuthu, B.T.R., Cranefield, S.: A categorization of simulation works on norms. In: Boella, G., Noriega, P., Pigozzi, G., Verhagen, H. (eds.) Normative Multi-Agent Systems, Dagstuhl Seminar Proceedings, vol. 09121. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany (2009)
12. Axelrod, R.: An evolutionary approach to norms. The American Political Science Review 80(4), 1095–1111 (1986)
13. Arcos, J.L., Esteva, M., Noriega, P., Rodríguez-Aguilar, J.A., Sierra, C.: Environment engineering for multiagent systems. Engineering Applications of Artificial Intelligence 18(2), 191–204 (2005)
14. Ceglar, A., Roddick, J.F.: Association mining. ACM Computing Surveys 38(2), 5 (2006)
15. Bandura, A.: Social Learning Theory. General Learning Press (1977)
16. Rodríguez-aguilar, J.A., Martín, F.J., Noriega, P., Garcia, P., Sierra, C.: Towards a test-bed for trading agents in electronic auction markets. AI Communications 11, 5–19 (1998)
17. Mannila, H., Toivonen, H., Inkeri Verkamo, A.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery 1(3), 259–289 (1997)
18. Savarimuthu, B.T.R., Cranefield, S., Purvis, M.A., Purvis, M.K.: Norm identification in multi-agent societies. Discussion Paper 2010/03, Department of Information Science, University of Otago
19. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) Proceedings of 20th International Conference on Very Large Data Bases (VLDB'94), Santiago de Chile, Chile, pp. 487–499. Morgan Kaufmann, San Francisco (1994)
20. Cardoso, H.L., Oliveira, E.: Directed deadline obligations in agent-based business contracts. In: Proceeding of the International Workshop on Coordination, Organization, Institutions and Norms in Agent Systems, COIN@AAMAS 2009 (2009)

Probabilistic Modeling of Mobile Agents' Trajectories

Štěpán Urban, Michal Jakob, and Michal Pěchouček

Agent Technology Center, Dept. of Cybernetics, FEE Czech Technical University
Technická 2, 166 27 Praha 6, Czech Republic
{urban,jakob,pechoucek}@agents.felk.cvut.cz

Abstract. We present a method for learning characteristic motion patterns of mobile agents. The method works on two levels. On the first level, it uses the expectation-maximization algorithm to build a Gaussian mixture model of the spatial density of agents' movement. On the second level, agents' trajectories as expressed as sequences of the components of the mixture model; the sequences are subsequently used to train hidden Markov models. The trained hidden Markov models are then employed to determine agent type, predict further agent movement or detect anomalous agents. The method has been evaluated in the maritime domain using ship trajectory data generated by the AGENTC maritime traffic simulation.

Keywords: Trajectory modeling, spatio-temporal learning, mobile agents, maritime traffic.

1 Introduction

Trajectories represent an important external manifestation of mobile agents' behavior. Modeling agents' trajectories therefore provides a valuable tool applicable in a number of scenarios ranging from categorizing unknown agents, predicting their future movement (and possibly action) up to identifying agents acting in an unusual and potentially malicious way.

In this paper, we propose a method which captures the spatial and temporal aspects of agents' movement using two different representations, a Gaussian mixture model and a hidden Markov model. This decomposition allows to reduce the computational complexity of the overall learning problem.

Our work is inspired by the work on motion pattern learning (e.g. [1,2]) as well as work on agent behavior modeling (e.g. [3,4]). The proposed method is general, though it has been developed as part of our larger work on modeling, detecting and disrupting illegal maritime activities [5,6].

After introducing our two-level model in Section 2, we show how it can be employed in two specific classification scenarios in Section 3. Evaluation on the domain of maritime traffic is presented in Section 4. Section 5 summarizes related work and we conclude with final remarks in Section 6.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 59–70, 2010.

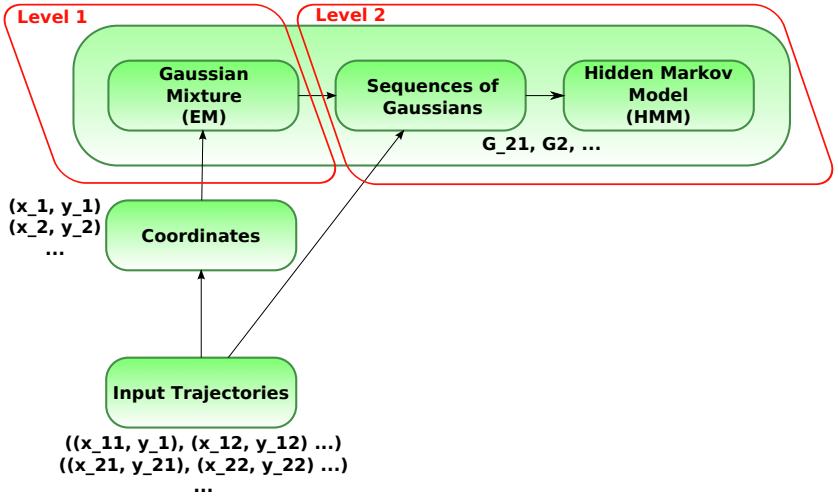


Fig. 1. Schema of the two-level trajectory modeling method

2 Two-Level Trajectory Model

As already mentioned, our approach represents agents' movement at two levels.

The proposed method works on two levels. On the first level, spatial properties of the traffic are represented using a Gaussian mixture model (GMM). On the second level, temporal aspects are captured using a hidden Markov model (HMM).

The input to the algorithm is a set of agent trajectories $\mathcal{T} = \{T_1, \ldots, T_m\}$. Each trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$ is a tuple representing a sequence of 2d coordinates $\boldsymbol{x} = (x_{lat}, x_{long}) \in \mathbb{R}^2$. The output of the algorithm is a Gaussian mixture model (Level 1) and a hidden Markov model (Level 2) best approximating the trajectories. Figure 1 gives a graphical overview of our approach.

2.1 Spatial Modeling

On the first level, the method uses the expectation-maximization algorithm [7] to build a Gaussian mixture model of the spatial density of the agents' motion. The algorithm disregards the sequential aspect of the agents' trajectories and treats them as unordered sets of agent positions. It then approximates the empirical distribution of these positions using a mixture of 2d Gaussian kernels. The process can be viewed as clustering the very large number of agents' positions into a significantly lower number of spatial clusters which are then used as the basis for temporal modeling.

More specifically, each Gaussian kernel

$$\phi(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x}-\boldsymbol{\mu})\right) \quad (1)$$

is parameterized by its mean $\boldsymbol{\mu} \in \mathbb{R}^2$ (the position of Gaussian kernel's center) and its 2-by-2 covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{2,2}$. Assuming the mixture consists of m components, the algorithm is looking for a tuple of parameters

$$\Theta = (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, w_1, \ldots, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m, w_m)$$

so that the mixture model

$$\Phi(\boldsymbol{x}|\Theta) = \sum_{i=1}^{m} w_i \phi(\boldsymbol{x}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) \tag{2}$$

best approximates the spatial distribution of the agents' positions in the trajectory set $\mathcal{T}$. The expectation-maximization algorithm [7] is used to find maximum likelihood estimates of parameters Θ^*, i.e.,

$$\Theta^* = arg\max_{\Theta} \Phi(\mathcal{T}|\Theta) = arg\max_{\Theta} \prod_{T=(\boldsymbol{x_1},\ldots,\boldsymbol{x_n})\in\mathcal{T}} \prod_{i=1}^{n} \Phi(x_i|\Theta) \tag{3}$$

for a given set of trajectories $\mathcal{T}$. An example of such a set is given in Figure 2.

The set of Gaussian kernels obtained is used as a basis for expressing agents' trajectories in the second level of the algorithm.

2.2 Temporal Modeling

The second level captures the temporal structure of agents' trajectories. Agents' trajectories are expressed as sequences of the Gaussian components of the mixture model; the sequences are subsequently used to train hidden Markov model.

Specifically, assume we have a Gaussian component model Φ (see (2)) and a trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$. For each point $\boldsymbol{x}$, the algorithm looks for a Gaussian kernel component to which the point belongs

$$i^* = \arg\max_{1\le i\le m} \phi(\boldsymbol{x}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) \tag{4}$$

By applying the above to all points, we obtain a sequence $Q = (i_1, \ldots, i_n)$ expressing the original sequence of locations as a sequence of (the indices of) the components of the GMM; we term such a sequence a *GMM-based trajectory*. After applying the above to all trajectories $\mathcal{T}$, we obtain a set of GMM-based trajectories $\mathcal{Q}$.

In the next step, a hidden Markov model is sought which best fits the sequences in $\mathcal{Q}$. We assume that the states of the model are observable and directly correspond to the components of the GMM-based model. The *Baum-Welch algorithm*[8] is then applied to learn a set $\mathcal{P}$ of transition probabilities $P(i_j|i_k)$ specifying the probability that a agent moves from an geographical region corresponding to component j to a region corresponding to component k.

Once a hidden Markov model is obtained, it can be used for evaluating the closeness of specific trajectories. Also in this case, the classified trajectory $T = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$

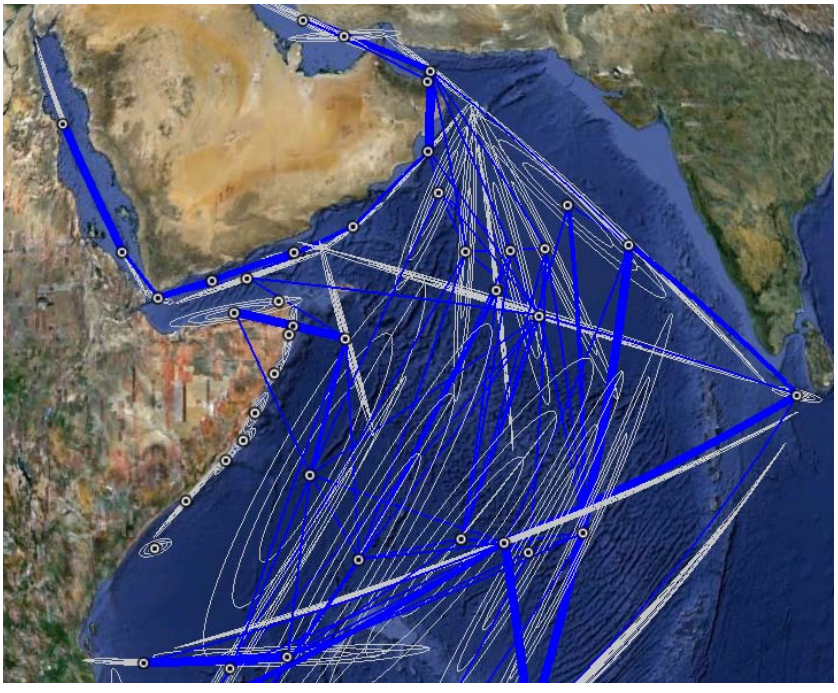


Fig. 2. Example two-level model. The white ellipsoids represent Gaussian kernels in the Level-1 GMM; blue overlay graph represents the Level-2 hidden Markov model. The thickness of the edges corresponds to the transition probability between the two kernels, i.e., between regions they represent.

is first converted into its GMM-based representation $Q = (i_1, \ldots, i_n)$ using (4). The probability of the sequence Q being produced by a model $\mathcal{P}$ is then calculated as

$$p(Q|\mathcal{P}) = \prod_{j=2}^{n} P(i_j|i_{j-1}) \tag{5}$$

An example of a HMM obtained using the outlined method is given in Figure 2.

3 Classification Modes

The two-level model can be employed in two modes (1) for identifying anomalous and thus possibly illegitimate agents, and (2) for categorizing agents into a predefined set of classes.

3.1 Agent Categorization

The agent categorization mode assumes there are labeled trajectory sets available for all categories of agents under consideration. On Level 1, all trajectories are used for

creating a single spatial model of the agent traffic; this is shared across all categories. On Level 2, an individual HMM for each category is created.

When an unknown agent is to be classified, it is evaluated for closeness against all HMMs (using (5)) and the category of the closest HMM is used as the category of the agent.

We refer to the classifier with the above described structure and operation as the *agent categorizer*.

3.2 Illegitimate Agent Detection

The illegal agent detection mode assumes that only trajectories for legitimate agents are known (and labeled); there are no known trajectories for illegitimate agents[1]. The learning phase of the algorithm is similar to the agent categorizer, only now HMMs are created only for legitimate agents.

When an unknown agent is to be classified, it is evaluated for closeness against all HMMs. However, because the HMMs now do not cover all categories of agents, the agent is classified as "legitimate" only if the closeness of its trajectory to the closest HMM is higher than a defined *closeness threshold*. Otherwise, the agent is categorized as "illegitimate". By varying the closeness threshold, we can moderate the trade-off between false negatives (an anomalous agents is classified into one of the legitimate classes) and false positives (a legitimate vessel is classified as anomalous); the trade-off can be quantified using an ROC curve.

We refer to the classifier with the above described structure and operation as the *illegitimate agent detector*.

4 Experimental Evaluation

We have evaluated the proposed method in the domain of maritime traffic. Vessel trajectory data were obtained from a data-driven simulation utilizing real-world data sources and FSM-based agent behavior models.

4.1 Experimental Testbed

The testbed used for evaluation has been developed as part of our larger activity on applying agent-based techniques to fight maritime crime. At the center of the testbed is an agent-based simulation platform incorporating a range of real-world data sources in order to provide a solid computational model of maritime activity. The platform can simulate the operation of thousands vessels of the following type:

- **Long-range transport vessel** – large- to very large-size vessels transporting cargo over long distances (typically intercontinental); these are the vessels that are most often targeted by pirates.

[1] This setting is often referred to as *learning from positive examples only*.

- **Short-range transport vessel** – small-to medium-size vessels carrying passengers and/or cargo close to the shore or across the Gulf of Aden.
- **Fishing vessel** – small- to medium-size vessels performing fishing within designated fishing zones; fishing vessels launch from their home harbors and return back after the fishing is completed.
- **Pirate vessel** – medium-size vessels operating within designated *piracy zones* and seeking to attack a long-range transport vessel. The pirate control module supports several strategies some of which can employ multiple vessels.

The behavioral models for individuals categories of vessels have been synthesized from the information about real strategies [9]. The vessel operational characteristics (length, tonnage, max speed etc.) are based on real-world data[2] too. More information about the testbed including a brief video overview can be found in on project's web site[3] and in [6].

4.2 Experimental Data

The simulation was run for 3215 simulation seconds and the trajectories sampled with 10-minute resolution. All four vessel classes implemented by three different FSMs were active in the simulation. The experimental scenario contained 500 vessels of each class. The execution of the simulation produced traces containing altogether $3,588,909$ coordinates which were used for all subsequent analyses. A subset of produced trajectories is shown in Figure 3.

4.3 Method Configuration

A key parameter of the method is the number of Gaussian kernels used by the Level 1 mixture model. To determine how many Gaussian distributions should be used to approximate agent spatial distribution with sufficient accuracy, we run the EM algorithm for different number of components C (from $C = 1$ to $C = 100$). Low number of Gaussian distributions cannot approximate real distribution well enough; on the other hand, for high number it will be difficult to learn the HMM (on level 2) because HMM learning time is increasing exponentially with number of hidden states. To see how well a GMM fits the data, we observe the likelihood function used in EM algorithm. From the likelihood function plot shown in Figure 4, we determine that we need at least 40 Gaussian distributions, which is the value used in our experiments.

Probabilities of all plausible transitions in the HMMs were initialized with small non-zero positive value (0.01) to avoid the case where sequences (especially long ones) are classified as having a zero degree of membership to a given HMM just because a particular transition was not present in the training data. This initialization procedure results in a significantly higher accuracy of classification.

[2] E.g. http://aislive.com, http://vesseltracker.com
[3] http://agents.felk.cvut.cz/projects/agentc/

Fig. 3. An subset of vessel trajectories used in the evaluation of the method (while lines)

4.4 Results

We now present the results for both classification modes.

4.5 Results for Agent Categorization

Agent categorization mode described in 3.1 was used to classify agents on experimental data described in 4.2. The classification accuracy was evaluated using 10-fold cross-validation. The resulting dependency of the accuracy on the length of test sequences, i.e. the sequences being classified, is given in Figure 5. In order to gain better insight into the operation of the classifier, we have also calculated the confusion matrix (Table 1).

We also calculate the confusion matrix for the classifier to understand the structure of misclassifications. The confusion matrix is given in Table 1. It follows that long-range transport vessels are the most easily identifiable; this is because they have completely different trajectories and visit different locations than other vessels. Local transport vessels are sometimes misclassified with fishing vessels (and vice-versa); that is because both classes of vessels operate close to the coast and their trajectories overlap. For similar reasons but with lower rate, pirate vessels are sometimes misclassified as local transport.

It follows that long-range transport vessels are the most easily identifiable; this is because they have completely different trajectories and visit different locations than other vessels. Local transport vessels are sometimes misclassified with fishing vessels

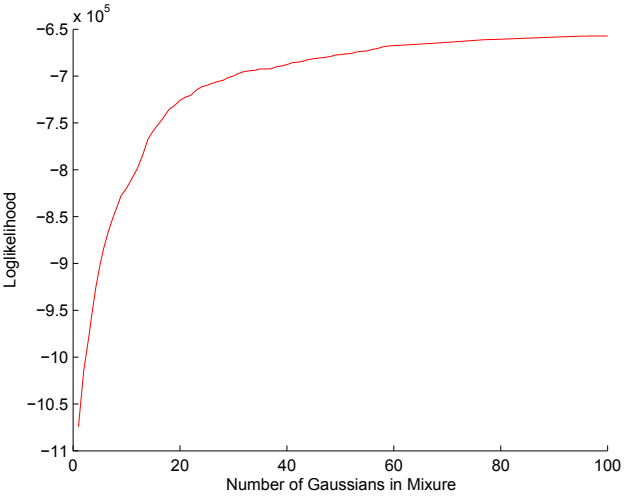


Fig. 4. Log-likelihood function in the dependence of number of the Gaussian kernels in the Level 1 model

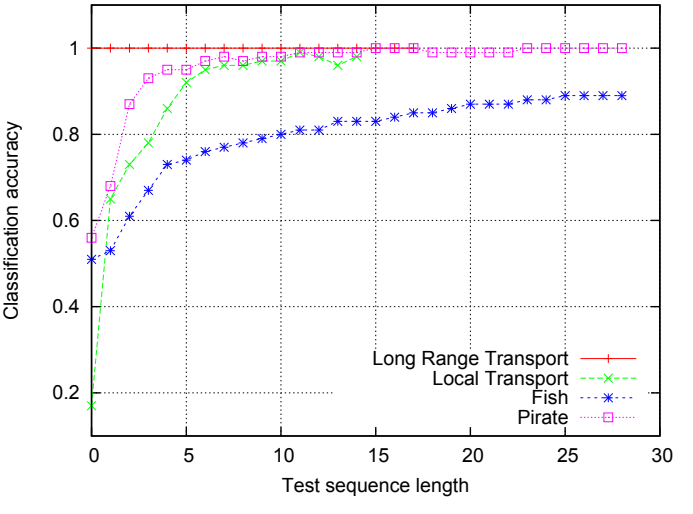


Fig. 5. Classification accuracy of the agent categorization classifier for different trace lengths

Table 1. Confusion matrix of the agent categorizer

	Actual			
Predicted	Long-range transport (1)	Local transport (2)	Fishing (3)	Pirate (4)
(1)	1	0	0	0
(2)	0	0.72	0.18	0.01
(3)	0	0.20	0.77	0.03
(4)	0	0.08	0.05	0.95

(and vice-versa); this is because both classes of vessels operate close to the coast and their trajectories overlap. For similar reasons but with lower rate, pirate vessels are sometimes misclassified as local transport.

4.6 Results for Illegitimate Agent Detection

For the detection classifier (Section 3.2), only data corresponding to the three legitimate types of vessels (long-range transport, short-range transport and fishing) were used. Again, 50 Gaussian kernels were used in the Level 1 model.

The accuracy of classification into legitimate vs. illegitimate agent was again evaluated using 10-fold cross-validation. As noted above, the trade-off between false negatives and false positives can be regulated by setting the closeness threshold of the classifier. By modifying the similarity threshold, we can moderate the trade-off between the false positive (an anomalous agent is classified into one of the legitimate classes) and false negative (a legitimate agent is classified as anomalous) rate. The trade-off can be quantified using an ROC curve[4] in Figure 6 (blue crosses). A test with perfect discrimination (no overlap in the two distributions) has a ROC plot that passes through the upper left corner (100% sensitivity, 100% specificity).

From the confusion matrix, we can compute sensitivity and specificity and compare this measure with the values obtained by the legitimate agent detection classifier. This value is only one number in ROC curve 6 and is depicted by a red cross. The position of the cross in the upper left corner shows the superior performance of the all class classifier over the illegitimate agent classifier described previously. This is not surprising given that the all class classifier model is constructed from a more representative learning base.

5 Related Work

Perhaps the approach most similar to our work is described in [1] where agent trajectories are represented as sequences of flow vectors, each vector consisting of four elements representing position and velocity of the object in 2D space. The patterns of trajectories are formed by a neural network. In [10], the authors use the EM algorithm

[4] *Receiver Operating Characteristic* (ROC), or simply *ROC curve*, is a graphical plot of the sensitivity vs. (1 - specificity) for a binary classifier system as its discrimination threshold is varied.

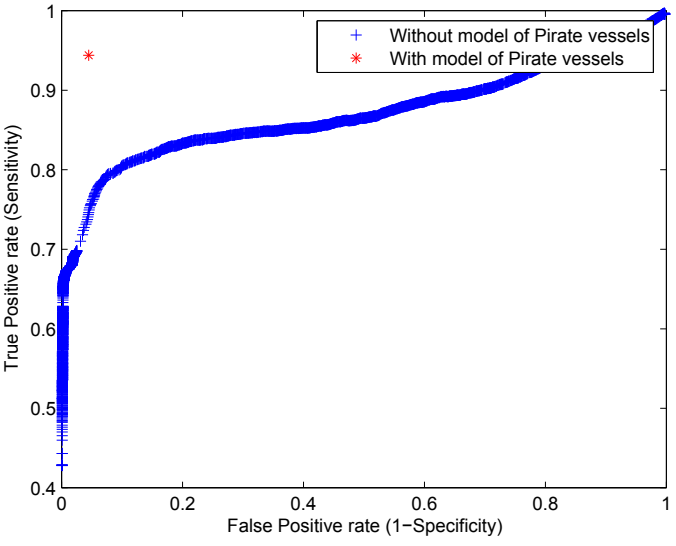


Fig. 6. ROC curve for the legitimate agent classifier (for a varied similarity threshold under which 'unknown/anomalous' classification output is produced). In addition, sensitivity vs. specificity relation for the agent categorizer classifier is also depicted (single point only as there is no variable parameter affecting the relation).

for learning zones. To learn agent traces, they simply compare a new trajectory with all routes that are already stored in the database using a simple distance measure. The limitation of this method is that they only use spatial information and temporal information is not well represented. In [2], the authors use a system based on a 2-layer hierarchical version of fuzzy K-means clustering. They first cluster similar agent trajectories into the same cluster according to their spatial information. Each object in a spatial cluster is then clustered according to temporal information. In addition, there are methods for detecting anomalies by direct comparison of behaviors without learning agent's motion patterns [11].

The specific application of agent motion patterns modeling on vessel agents has been studied in past few years, with existing papers mostly focused on anomaly detection. In [12], the authors use the framework of adaptive kernel estimation and hidden Markov models for the purpose of anomaly detection. In [13] and [14], the authors use a neural network trained on space-discretized AIS trajectory data to learn behavior of normal agents and then to detect anomalies as well as to predict future vessel position and velocity.

6 Conclusion

We have proposed a method for learning motion patterns of mobile agents. The combination of spatial modeling using GMM (level 1) and temporal modeling using HMM

(level 2) provides expressive framework for capturing agent behavior. We provide variants both for both the case where learning data are available only for the legitimate agents and for the case where data are available for all categories of agents. In the former case, the mechanism works as an anomaly detector – agent traces which are not similar enough to any of the legitimate models are classified as anomalous and thus likely illegitimate. In the latter case, the mechanism works as a standard classifier. We have evaluated the method in the maritime domain using trajectory data of four different categories of ships.

The future work will proceed along several possible directions. On the technical side, more research into the operation of the individual modeling algorithms at both levels and their mutual interplay is needed, including experimentation with alternative frameworks for spatio-temporal pattern representation. On the modeling side, the introduction of more background and context information should help to further improve the accuracy of the produced models.

Acknowledgements

The work presented is supported by the Office for Naval Research project no. N00014-09-1-0537 and by the Czech Ministry of Education, Youth and Sports under Research Programme no. MSM6840770038: Decision Making and Control for Manufacturing III.

References

1. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. In: BMVC '95: Proceedings of the 6th British Conference on Machine Vision, vol. 2, pp. 583–592. BMVA Press, Surrey (1995)
2. Hu, W., Xiao, X., Fu, Z., Xie, D., Tan, T., Maybank, S.: A system for learning statistical motion patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(9), 1450–1464 (2006)
3. Carmel, D., Markovitch, S.: Learning models of intelligent agents. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, Portland, Oregon, pp. 62–67 (1996)
4. Sukthankar, G., Sycara, K.: Policy recognition for multi-player tactical scenarios. In: AAMAS '07: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1–8. ACM, New York (2007)
5. Jakob, M., Vaněk, O., Urban, Š., Benda, P., Pěchouček, M.: AgentC: Agent-based testbed for adversarial modeling and reasoning in the maritime domain. In: Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010) (May 2010)
6. Jakob, M., Vaněk, O., Urban, Š., Benda, P., Pěchouček, M.: Employing agents to improve the security of international maritime transport. In: Proceedings of the 6th workshop on Agents in Traffic and Transportation (ATT2010) (May 2010)
7. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society, Series B 39(1), 1–38 (1977)
8. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. The Annals of Mathematical Statistics 41(1), 164–171 (1970)

9. IMB Piracy Reporting Centre (2009), http://www.icc-ccs.org/index.php?option=com_content&view=article&id=30
10. Makris, D., Ellis, T.: Learning semantic scene models from observing activity in visual surveillance. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 35(3), 397–408 (2005)
11. Computer, H.Z., Zhong, H.: Detecting unusual activity in video (2004)
12. Ristic, B., La Scala, B., Morelande, M., Gordon, N.: Statistical analysis of motion patterns in AIS data: Anomaly detection and motion prediction. In: 11th International Conference on Information Fusion, July 3-30, 2008, pp. 1–7 (2008)
13. Bomberger, N., Rhodes, B., Seibert, M., Waxman, A.: Associative learning of vessel motion patterns for maritime situation awareness. In: 9th International Conference on Information Fusion, July 2006, pp. 1–8 (2006)
14. Rhodes, B., Bomberger, N., Zandipour, M.: Probabilistic associative learning of vessel motion patterns at multiple spatial scales for maritime situation awareness. In: 10th International Conference on Information Fusion, July 2007, pp. 1–8 (2007)

Real-Time Sensory Pattern Mining for Autonomous Agents

Pedro Sequeira[1] and Cláudia Antunes[2]

[1] INESC-ID / IST, Av. Prof. Dr. Cavaco Silva, 2744-016 Porto Salvo, Portugal
pedro.sequeira@gaips.inesc-id.pt
[2] Instituto Superior Técnico, Av. Rovisco Pais, 1049-001 Lisboa, Portugal
claudia.antunes@ist.utl.pt

Abstract. Autonomous agents are systems situated in dynamic environments. They pursue goals and satisfy their needs by responding to external events from the environment. In these unpredictable conditions, the agents' adaptive skills are a key factor for their success. Based on previous interactions with its environment, an agent must learn new knowledge about it, and use that information to guide its behavior throughout time. In order to build more believable agents, we need to provide them with structures that represent that knowledge, and mechanisms that update them overtime to reflect the agents' experience. Pattern mining, a subfield of data mining, is a knowledge discovery technique which aims to extract previously unknown associations and causal structures from existing data sources. In this paper we propose the use of pattern mining techniques in autonomous agents to allow the extraction of sensory patterns from the agent's perceptions in real-time. We extend some structures used in pattern mining and employ a statistical test to allow an agent of discovering useful information about the environment while exploring it.

Keywords: Autonomous agents, adaptation, learning, pattern mining, knowledge discovery.

1 Introduction

There are several definitions of autonomous agents, and several attempts to define the requirements that a designer must meet when building those agents [1]. Nevertheless, there a few characteristics which are common to those definitions of agents, specifically that they are systems situated in dynamic environments, which have and actively pursue some goals and satisfy their needs by autonomously responding to external events from that environment [1] [2]. The characteristics of the environment make restrictions on the sensations perceived, and shape the actions performed, the knowledge constructed, and the decisions that are taken by the agent at each moment. Because those environments can be dynamic and unpredictable, the agent must have some mechanisms to distinguish the features that are perceived from it, focusing its attention in those that seem more promising to achieve its goals, and ignoring others that do not [1]. This ability of adapting to and learning new knowledge from the environment while interacting with it in real-time defines restrictions and requirements when building autonomous agents

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 71–83, 2010.

with such capabilities [2]. Namely, we need to provide them with some structures that represent the acquired knowledge about the environment, and mechanisms that update these representations overtime to reflect the agent's interaction experience.

Data mining encloses a set of techniques to extract previously unknown and possible useful information from data [3]. One of such techniques is transactional pattern mining, which extracts frequent associations and causal structures among sets of objects or items from several transactions.

In this paper we propose the use of pattern mining techniques within the autonomous agents paradigm in order to provide those agents with the ability of discovering associative patterns in their perceptions while they interact with their environment, i.e., in real-time. These patterns constitute the agents' knowledge about its world taken from the regularities that are perceived from its experience. Later on they can be useful for the agent to form concepts about the environment and by setting expectations about future events. To achieve that we extended some structures used in pattern mining to represent the discovered knowledge. We also adopted a statistical test to detect significant sensory information for the agent to discover useful facts about its world, and came up with some heuristics and algorithms to update and maintain the knowledge structures in real-time, while the agent explores it.

This paper is organized as follows: the next section gives an overview of the general idea and motivation behind the work presented, and define the research problem we are trying to solve. In the following section we present several extensions to related work as possible solutions to the problem. In section 5 we present the tests used to validate the proposed solutions and a comparative analysis relating them. Finally we draw some conclusions and possible future extensions for the work.

2 The Problem

In this section we describe the background motivation and research problem behind the current work, introduce the pattern mining problem and explain how the analogy between these problems and the autonomous agents' paradigm can be made. Next we characterize the problem to be solved and define a set of requirements to be satisfied.

2.1 Background

In a previous work [4], Sequeira *et al.* presented *SOTAI* (*Smart ObjecT-Agent Interaction*) framework to help autonomous agents identify the set of possible interactions with unknown objects of the environment, based on previous experiences with other objects. The approach is based on the notion of Gibson's *affordances* [5] which can be defined as the interaction opportunities which are transmitted by the objects to the agents taking into account their interaction capabilities. Within the *SOTAI* framework, an agent is provided with a set of sensors from which it perceives its environment. At each moment one sensor can contain a set of sensations which are modeled as symbolic qualities or features of the perceived objects from the environment. For example, if an agent interacts with an orange then it will receive the symbol *orange* in the *color* sensor, *spherical* in the *shape* sensor, the symbol *soft* to the *touch-texture* sensor, etc.

The framework is based in Cohen *et al.*'s *block-building* approach [6]. It allows the construction of small pieces of information called *Base Fluents*, which are pairs of *Sensations* that occur frequent and simultaneously in the agent's sensors. Using a Chi-square [7] statistical test, one can determine whether two monitored *Sensations* are correlated, and also their association strength by calculating their Phi-coefficient [7]. If an association is detected, *Base Fluents* are created and later tested in pairs to determine new associations. The learning process occurs in real-time while the agent interacts with its environment, building new and larger blocks of information (*Fluents*) from smaller structures. More implementation details can be found in [4].

To test the knowledge generation mechanisms, a test application (*Sotai-Tester*) was created where an agent explores and interacts with a simulated environment composed of several objects in a random manner. It selects at each decision moment an object to interact with and receives sensations from the environment. For example, when it *sees* an object, the agent takes its *shape* and *color* features from the object's internal description, when it *touches* it receives the object's tactile information, when it *eats* it takes flavor and energy properties, etc.

2.2 The Pattern Mining Analogy

Pattern mining is a technique that is used to discover frequent patterns, associations, correlations, or causal structures among sets of items or objects [8]. In *transactional pattern mining*, a transaction database (DB) is a set of transactions, each of which containing a set of items describing objects or events that occur simultaneously, at a given point in time. The goal of the pattern mining algorithms is to discover all the maximal sets of items that occur together in a significant number of transactions in the DB. *Itemsets* are considered *patterns* if the number of transactions in the DB where they occur is greater than a minimal threshold value, the pre-established *support*.

Considering the characteristics of the agents being modeled within the *SOTAI* framework, one can envisage a way of applying some of these pattern mining algorithms within an autonomous agent paradigm. In the case of such agents, at each time every sensor will have a certain stimulus represented by some symbol. In the knowledge-discovery language these symbols constitute categorical nominal attribute data or alphabets as they describe discrete values with no ordering between them. Moreover, if we consider the symbols as being items, then at each time the set of all the stimuli in the agent's sensors (it's current perception state) can be considered as a transaction. In this context, a DB can be defined as the whole record of perception states of the agent during all of its execution time.

2.3 Problem Definition

Having described the perception mechanism of the agent that we are modeling in the context of pattern mining (as a set of items and transactions), we can define the problem of discovering knowledge from the agent's interactions with its environment as a problem of mining patterns from sets of transactions from a database.

More specifically, we can better define the problem as finding *synchronous sensory patterns*: sets of stimuli that occur frequently and simultaneously within the agent's sensors that reveal some regularities of its environment. Because this discovering process directly depends on the perceptions made, the number and set of patterns discovered will depend on each agent's interaction experience even if the environment and its conditions (the objects and its features) are the same. From this definition, the mining algorithms to apply have to follow some requirements:

- Discover sensory patterns in real-time as the agent interacts with the environment;
- Discover the maximum amount of sensory patterns that indicate useful information to be used by the agent at which it should focus its attention;
- Maintain within the knowledge structure the maximum number of patterns so that they can be easily accessed and used by the agent to make decisions;
- Use the minimal amount of resources (both storage and time consumption).

3 Sensory Pattern Mining for Autonomous Agents

In this section we present several approaches to the problem defined in the previous section. We start by a quick overview of the *FP-Growth* algorithm and envisage possible modifications of the algorithm and its structures so that it can be used in mining patterns in real time. Finally we propose the use of the Jaccard index statistic as a way to retrieve more and meaningful patterns from the sensory data.

3.1 Transactional Pattern Mining

As described above, at each instant the agent's perception state describes a set of stimuli that can be defined as a transaction which can be provided to pattern mining algorithms. The best well-known algorithm within the area of transactional pattern mining is the Apriori algorithm introduced by Agrawal *et al.* [8]. It iteratively generates the set of candidate patterns (*itemsets*) of some length k from the set of frequent-patterns of length $k-1$. If the candidate is frequent, i.e. its occurrence is greater than the minimum support threshold, then it is considered a pattern, and new candidates are generated from this set to be tested in the next step. However, and despite the simplicity of the algorithm, its candidate generation and test philosophy impairs its efficiency, and makes difficult its extension to deal with continuous flows of data and real-time environments.

More recently, Han *et al.* proposed an algorithm to surpass some of the problems presented by the Apriori algorithm, namely the necessity of having to generate and test a huge number of candidate sets and execute multiple scans over the entire DB in order to discover the patterns. As such, Han *et al.* developed *FP-Growth*, an algorithm that builds up a compact structure (*FP-Tree*) from the data in a way that avoids scanning the DB multiple times [9]. The algorithm works in three steps: first, it scans the DB once in order to identify frequent items and identifies the best ordering among them; the frequency-descending order among items is then used to reorder each transaction,

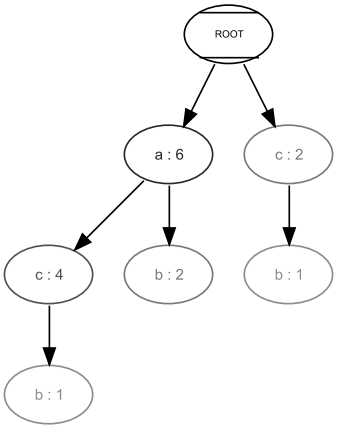


Fig. 1. The *FP-Tree* (left) resulting from the application of the *FP-Growth* algorithm (*min. support*=0.4) over the set of transactions (right). Lighter nodes represent smaller support values.

using the *FP-Tree* to compact all the transactions in the DB; finally, it goes through the tree in order to identify all the patterns in the database. Fig. 1 illustrates the result of applying the *FP-Growth* algorithm (left) to a set of transactions in a DB (right). The algorithm discovers a, b, c and ac as patterns (*min. support* = 0.4).

In light of the problem described earlier and taking into account the requirements defined in section 2.3, the *FP-Growth* algorithm presents some drawbacks:

- First by requiring a first scanning over the whole set of transactions from the DB to order items by their occurrence frequency, which is impossible to realize in the autonomous agent's case because we want the mining process to be made online;
- The main objective of the algorithm is to determine all the frequent *itemsets*. Due to the inherent compactness of the *FP-Tree* structure, it is sometimes difficult to determine whether specific combinations of items are considered patterns, having to scan the tree to determine their support;
- Finally, as another consequence of the compactness of the *FP-Tree*, it is difficult to verify if a specific *itemset* is a pattern in real-time. Indeed it would be beneficial for the agent to have an easy and rapid access to all the patterns discovered so far in order to use them to make better evaluations of the current state and also take better decisions upon it.

3.2 Real-Time Pattern Mining

One of the requirements stated in section 2.3 was that the algorithm to be developed must be executed in real-time, while the agent interacts with its environment, taking into consideration the past sensory experiences. As such, the first attempt to solve this issue was to try to build an *FP-Tree* structure using the same algorithm before mentioned, but doing it in only one scan over the set of transactions of the DB. To do that we discarded the first scan used to determine the order of the items according to their frequency. Because this order is only a heuristic to minimize the number of nodes generated (not

guaranteeing that the generated tree is the smallest one possible), we established a fixed order to be used by the algorithm, namely the alphabetic order of the symbols representing the agent's perceptions. Naturally, other kinds of ordering heuristics could be used, namely the ones based on domain knowledge about the environment, for example by determining what stimulus were more likely to be present in the agent's sensors throughout time, estimating the ordering among items. Nevertheless, because the objective was to provide a biased-free mechanism for the agents to explore the environment starting from (almost) zero information about it, and so we used the alphabetical order.

This change of ordering can have an impact on the time used to build the patterns tree, but does not solve the requirement of having the maximum number of known patterns readily accessible to the agent. To solve that problem we changed the way the transactions are inserted in the tree in such a way that now each node represents a specific *itemset* and its count value represents the frequency of that *itemset* so far. The algorithm thus generates all the possible sub-combinations of *itemsets* whenever a transaction is inserted. For example, if we inserted the transaction abc in a new tree (Fig. 2), the nodes a, $a \rightarrow b$, b, $a \rightarrow c$, $a \rightarrow b \rightarrow c$ and $b \rightarrow c$ are created with a count value of 1. We call this set of nodes representing all the sub-combinations of an *itemset* its *Dependency Tree* (DT). By making use of the anti-monotone Apriori heuristic [8] we can state that for an *itemset* to be frequent then every node in its DT must also be frequent. The idea behind this way of building the patterns tree is to have the maximum number of frequent patterns stored in the tree's structure without the need of having to search the whole tree for sub-combinations of *itemsets* and determine whether they are frequent.

Because the number of possible sub-combinations of an *itemset* can be very large, storing all the combinations for every possible transaction seems a very unfeasible and resource-consuming task. To tackle with that problem we defined a heuristic to prune those nodes that do not seem promising in becoming a pattern. This heuristic prunes a node (deletes it from the tree) whenever it's considered as an infrequent node, i.e., when its support (count value) drops below a certain threshold of the pre-established minimum support. In the context of this work we defined this limit as 0.75 times the minimum support[1]. This pruning heuristic is applied when determining the frequent patterns (nodes) to reduce the tree's size overtime. In Fig. 3 we can see the result of applying this new algorithm to the previous example.

3.3 The Jaccard Index

In our opinion, one of the problems that these pattern mining algorithms suffer lies within what they consider to be a frequent pattern. This judgment is based on the statistic of the frequency of an *itemset* in relation to the total number of transactions recorded. In the context of the current work however, sometimes there are some sets of stimuli which frequency is low, but that appear to occur always simultaneously. Looking at the previous example DB, we may notice that most of the times when item b appears in a

[1] Through experimentation, a high percentage value should be used as threshold to determine infrequent nodes during the pruning procedure. As such, 0.75 proved to be a value which pruned several nodes without damaging the overall patterns discovered.

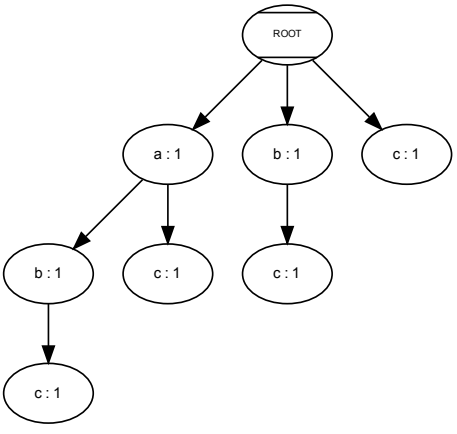


Fig. 2. The patterns tree after inserting transaction abc. It also represents the *Dependency Tree* (DT) of the node $a \rightarrow b \rightarrow c$.

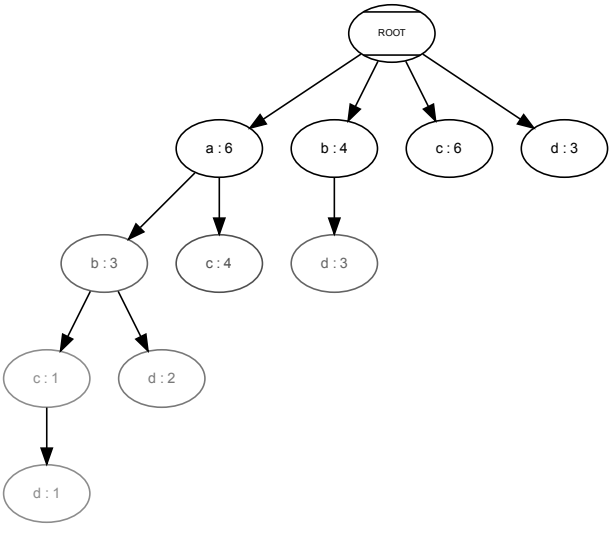


Fig. 3. The tree resulting from the application of the ”all-combinations” algorithm (transaction set and support is the same as the above example)

transaction, a and d also occur in that transaction. Because the support of the *itemsets* ab, bd and abd is very low, frequency-based algorithms cannot detect them as being patterns. This makes that a lot of useful information about the environment's regularities is being discarded by the agent throughout time.

Due to that fact, we decided to look for a statistic to determine the correlation level between nominal variables (as is the case of the ones here described). Considering the nature and structure of the patterns trees being built, we decided to apply the Jaccard index statistic [10] to determine the frequent patterns. This statistic allows us to determine the level of correlation between several variables as a function of the frequency of their intersection over their union. In the case of two variables A and B (non independent) the index can be expressed using the following formula [7]:

$$Jacc(AB) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{1}$$

The index value varies from 0 to 1, so values near 0 indicate a weak correlation between the variables while values near 1 indicate a strong correlation between them. This statistic can be used as a useful heuristic to determine sensory patterns in agents perceptions by allowing us to characterize the level of correlation between the stimuli as a measure of the deviation between their co-occurrence and the sum of all of its occurrences, which at the same time ignores the total number of perceptions made so far. By extending equation (1) to n variables we can informally write the Jaccard index by equation (2).

$$\textit{Jacc idx} = \frac{\text{no. of co-occurrences of the variables}}{\text{no. of co-occurrences} + \text{no. of non-simultaneous occurrences}} \tag{2}$$

If we take the structure of the patterns tree being developed here, we can determine the Jaccard index value of some *itemset* by dividing the count value of the node which represents the *itemset* in the tree (which precisely corresponds to the number of co-occurrences between the items) by the weighted sum of the count values of all the nodes within its DT. The weight determines the sign of the nodes' count value according to the parity of their depth in the tree ($-count$ if odd depth, $+count$ if even).

In this new context, the pruning heuristic is applied right after the insertion of a new transaction, to all the nodes within the DT of the node representing the transaction in the tree, because these were the nodes which count values were updated (for example, if transaction abc is inserted, the nodes belonging to the DT of the node $a \rightarrow b \rightarrow c$ are considered). Following the example DB of the previous sections, the application of this new approach results in the tree depicted in Fig. 4. As we can see, for a minimum Jaccard index of 0.4, the algorithm finds the same patterns discovered by the previous algorithms plus the *itemsets* ab and bd. *Itemset* abd still does not have enough support to be considered a pattern under the Jaccard index statistic, but it does have minimum support for not being removed under the pruning heuristic.

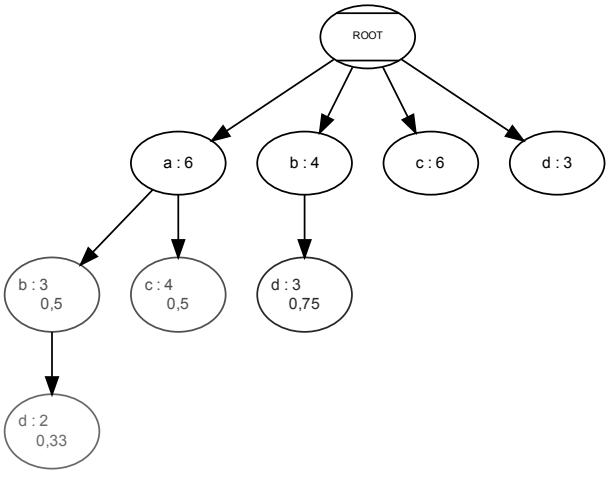


Fig. 4. The tree resulting from the application of the ”jaccard-index” algorithm (transaction set and support is the same as the above example). Jaccard index values are represented in the nodes below their support.

4 Experimental Results

In this section we describe the tests carried out over the proposed algorithms to determine their validity and the usefulness of the generated patterns, by presenting the main results and making a comparative analysis between the algorithms performance.

4.1 Test Conditions

To test the efficiency and quantity of patterns generated by the proposed algorithms we decided to test them against the *FP-Growth* [9] algorithm and the *SOTAI* framework algorithm [4], both described earlier. Of course *FP-Growth* is an algorithm which purpose is somewhat different from the type of pattern mining we are aiming at here. Nevertheless, because it served as a base algorithm for the proposed extensions, it provides reference values that we can use when comparing the quantity of patterns generated and the efficiency of the mining processes. In relation to the *SOTAI* framework, we adapted the generated *Fluents* as sensory patterns by considering the set of *Sensations* they represent as *itemsets*. The set of algorithms we ended up testing is the following (short name in brackets):

- *SOTAI* framework knowledge-generation algorithm (***Sotai***);
- *FP-Growth* algorithm (***FP-Growth***);
- *FP-Growth* with alphabetic item order, one scan over the DB (***Alpha-FP-Growth***);
- Algorithm considering all sub-combinations in every transaction with pruning heuristic (***All-Comb***);
- Same algorithm as All-Comb but using the Jaccard index statistic as a heuristic to determine patterns (***Jacc-Index***).

Because we wanted to test the algorithms within an autonomous agent architecture, in a context where the agent continuously interacted with its environment and which perceived sensations were in the form of symbols describing categorical nominal attributes of the objects, we chose *Sotai-Tester* [4] application to generate the transaction data to be input to the algorithms. *Sotai-Tester* agent, as explained before, has a random behavior and at each decision time it chooses an object from the environment to interact with. After that interaction is over, it chooses another object and so on. *Sotai-Tester* environment has a total of 8 objects that the agent can interact with, and the agent has a total of 13 sensors to perceive the environment. There are a total of 70 possible individual symbols to describe the perceptions of the agent. Because some of these symbols can be present at the same time within the agent's sensors, we considered a combination of two or more symbols as a single sensation (for example, if the *color* sensor has *yellow* and *red* symbols at the same update cycle, sensation *color-yellow-red* is considered as being an item). As such, at each update cycle of the agent's process, we recorded in a simple text file all the sensations that were present in the agent's sensors at that time. Each line of the text file thus represents a transaction to be processed by the pattern mining algorithm.

4.2 Metrics

The goal of the tests is essentially to analyze the algorithms' performance and the number and quality of the patterns it finds, i.e., the knowledge generated by the agent from perceiving its environment while interacting with it. As such, the following metrics were adopted and measured for each algorithm at each test execution:

- Time used to build the knowledge structure (tree), reading each transaction one-by-one from the previously generated text files, measured in CPU time (***Build-Time***);
- The number of nodes used to build the tree in order to evaluate the algorithms in terms of memory requirements (***Nodes-Number***);
- Time used to retrieve from the tree every possible pattern according to the minimal threshold established, measure in CPU time (***Pattern-Time***);
- Total number of patterns found. One-item-length patterns were ignored because we are interested in finding correlations in the sensory data (***Pattern-Number***).

Using the *Sotai-Tester* application described earlier we generated a total of 10 text files containing 10K transactions (average length = 5), each representing an update cycle of the agent. To see the relationship between the limit values established to discover patterns and the algorithms' performance, we iterated the minimum support threshold over 10 possible values from 0.1 to 1. In the case of the *Jacc-Index* algorithm, this threshold corresponds to a minimum value for the Jaccard's index associated with a node. In relation to the *Sotai* algorithm, the threshold is the minimum association strength (Phi-coefficient) between the generated structures. After the execution of all the tests, we averaged the values of each metric for each algorithm and each support value, removing mild outliers through quartile estimation.

4.3 Results and Comparative Analysis

The results of the tests described earlier are depicted in Fig. 5. By looking at the algorithms' behavior during the tests we are able to make the following observations about them:

SOTAI framework's knowledge-discovery algorithm didn't perform well at any of the analyzed parameters. It takes too much time to discover associations from the sensory data, maintaining unnecessary information of statistical tests, and discovering few patterns overtime due to its *block-building* approach explained earlier.

Although we introduced *FP-Growth* during the tests only to establish base values for the measures, we found that the algorithm performed very well within its "family". As we expected, by having the transactions sorted by item frequency before its introduction in the tree, less nodes are needed to build the tree and less time is required to search for the patterns than the alphabetical-order algorithm. By relying on the *itemsets* frequency, *FP-Growth* discovered, as expected, less patterns than the statistic-based association mining algorithms (*Sotai* and *Jacc-Index*). We can also observe that storing all the subcombinations in memory isn't that useful when mining for frequent patterns: it creates more nodes and requires pruning over the entire tree to find patterns. As a conclusion, none of the *FP-Growth*-based algorithms represents a real solution for the problem being solved.

The first thing we can say about the proposed Jaccard-index-based algorithm is that if we set the minimum index threshold too low, we end up having too many nodes, too many *itemsets* considered as patterns, and also spend too much time scanning the tree

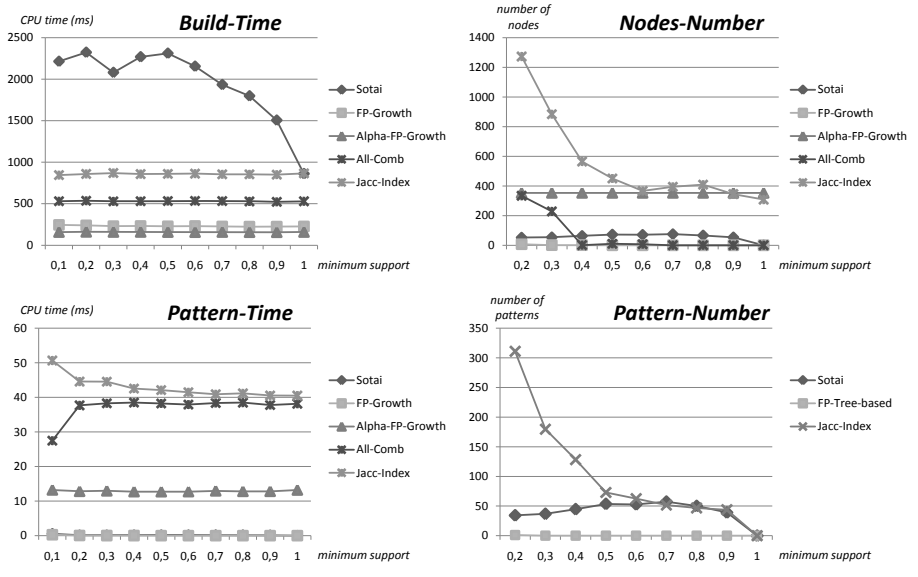


Fig. 5. The graphics containing the results of the tests performed over all the proposed sensory pattern mining algorithms

for those patterns. A compromise must be made between performance and usefulness of the discovered patterns as to consider how strong an association between stimuli must be for it to be considered a regularity of the environment, a pattern. As expected, it takes more time to build the tree and search for patterns than the *FP-Growth*-based algorithms because it needs to calculate the statistic to determine patterns. However, in this case, the structure of the patterns tree, by storing all the sub-combinations of *itemsets*, enhances its mining performance by maintaining in memory the nodes that are necessary for the index calculation (its Dependency Tree). We believe that the algorithm's performance shows that it can be a solution to our problem: it generates patterns of sensory data from the agent's perceptions throughout time, based on the statistical significance of the correlation between the stimuli they represent; it also performs this task in a reasonable time to be used by the agent while it interacts with its environment, i.e., in real-time (less than 1 second to process 10K update cycles).

5 Conclusions and Future Work

Autonomous agents are systems inhabiting dynamic and unpredictable environments which they perceive and react to events accordingly. To survive, the agent must learn new facts from the world and perceive its regularities to make better decisions on how to act upon it. In this paper we showed that by modifying some pattern mining algorithms we can provide autonomous agents with mechanisms to discover useful information about its environment while interacting with it in real-time. If we see the agents' perceptions as transactions of a DB, then we can apply those algorithms to discover sensory patterns from the environment. We proposed a new structure to store the sensory information perceived by the agent, which is constructed in a way that facilitates the retrieval of the sensory patterns. We also proposed a new heuristic to discover frequent patterns by using the Jaccard index statistic, which is sensible to some regularities of the environment that are not frequent, but denote particular cases of correlations within the perceptions.

We believe that the use of data mining techniques will provide autonomous agents with capabilities of discovering patterns relating their activities while interacting with the environment. For now, the proposed algorithms discover synchronous sensory patterns, i.e., sets of stimuli that co-occur frequently. In the future we want to extend them for the discovery of *asynchronous patterns*, i.e., sets of stimuli that frequently occur one after the other, revealing causal relations between them. To achieve that, we can adapt algorithms from the sequential pattern mining area [11]. We would also like to test the algorithms in different contexts, testing the pattern mining mechanisms in richer scenarios, possibly involving virtual environments and synthetic characters. Finally another approach would be to test this solution within a multi-agent application. Because different agents interact with the environment in different manners at particular instants of time, they will perceive the world in a singular manner and as such, they will create different sensory patterns throughout time. It would be interesting to check commonalities and differences between the patterns created, and provide them with communication capabilities so that overall social knowledge could be created to reflect the experiences of particular groups.

Acknowledgments

This paper was supported by a scholarship (SFRH / BD / 38681 / 2007) granted by the Fundação para a Ciência e Tecnologia. The authors are solely responsible for the content of this publication. It does not represent the opinion of the Fundação para a Ciência e Tecnologia, which is not responsible for any use that might be made of data appearing therein.

References

1. Franklin, S., Graesser, A.: Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. In: Jennings, N.R., Wooldridge, M.J., Müller, J.P. (eds.) ECAI-WS 1996 and ATAL 1996. LNCS, vol. 1193, pp. 21–36. Springer, Heidelberg (1997)
2. Maes, P.: Modeling adaptive autonomous agents. Artificial life 1(1-2), 135–162 (1994)
3. Frawley, W., Piatetsky-Shapiro, G., Matheus, C.: Knowledge discovery in databases: An overview. AI Magazine 13(3), 57–70 (1992)
4. Sequeira, P., Vala, M., Paiva, A.: What can i do with this?: finding possible interactions between characters and objects. In: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '07, pp. 5:1–5:7. ACM, New York (2007)
5. Gibson, J.: The ecological approach to visual perception. Houghton Mifflin (1979)
6. Cohen, P., Atkin, M., Oates, T., Beal, C.: Neo: Learning conceptual knowledge by sensorimotor interaction with an environment. In: Proceedings of the First International Conference on Autonomous Agents, pp. 170–177 (1997)
7. Warrens, M.J.: Similarity coefficients for binary data: properties of coefficients, coefficient matrices, multi-way metrics and multivariate coefficients. Doctoral thesis, Leiden University (2008)
8. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, Citeseer, vol. 1215, pp. 487–499 (1994)
9. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery 8(1), 53–87 (2004)
10. Jaccard, P.: The distribution of the flora in the alpine zone. New Phytologist (1912)
11. Antunes, C., Oliveira, A.: Sequential pattern mining algorithms: Trade-offs between speed and memory. In: 2nd Workshop on Mining Graphs, Trees and Seq, Citeseer (2004)

Part III

Data Mining in Agents

Analyzing Agent-Based Simulations of Inter-organizational Networks

Dominik Schmitz, Thomas Arzdorf, Matthias Jarke, and Gerhard Lakemeyer

RWTH Aachen University, Informatik 5, Ahornstr. 55, 52056 Aachen, Germany
{schmitz,arzdorf,jarke}@dbis.rwth-aachen.de,
gerhard@cs.rwth-aachen.de

Abstract. Modeling and simulation is intended to support the understanding and governance of inter-organizational networks. But the analysis of the outcome of such agent-based simulations requires advanced support. Due to the complexity of emergent behavior, the modeler must firstly decide about the validity of the simulation model. Secondly, the runs must be interpreted in order to receive support on the right managerial choices. In this paper we give a list of questions addressing both issues, validation and support, for the field of inter-organizational networks and investigate four different kinds of analysis – time series analysis, association rule mining, clustering, and social network analysis – in regard to their potential for providing support.

1 Introduction

Inter-organizational networks are a flexible form of organizing the cooperation between enterprises [20]. Examples are the "entrepreneurship networks" that have evolved around large technical universities such as MIT and RWTH Aachen University [13]. In comparison to hierarchies, networks replace the stable, hierarchical structure by more flexible trust relationships. Sociological research has also shown that distrust is equally important in particular to avoid Mafiose-like structures [7]. Eventually, confidence in the network as a whole is needed as another ingredient. From these insights, Gans et al. [9] have derived the trust-confidence-distrust model of success and failure of such networks (see Fig. 1). It does not only explain that a balanced mix of the three trust ingredients is needed, but also hints at means of governance, in particular network rules, that allow, for example, to foster the growth of trust relationships or to fight tranquility. The overall aim is to keep such networks on a successful path by helping to decide about the right managerial choices [17]. Only the continuous adaptation of these measures ensures the liveliness, innovativeness, and flexibility that makes this form of organization advantageous over purely static and stable hierarchical structures as well as too flexible and volatile market relationships.

In order to address the inherently dynamic issues – trust only evolves over time – as well as the need for continuous adaptation – new measures need to be evaluated –, Gans et al. [9] suggest a method that integrates two perspectives: a strategic modeling perspective based on the agent- and goal-oriented formalism *i** [21] and a logic-based

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 87–102, 2010.

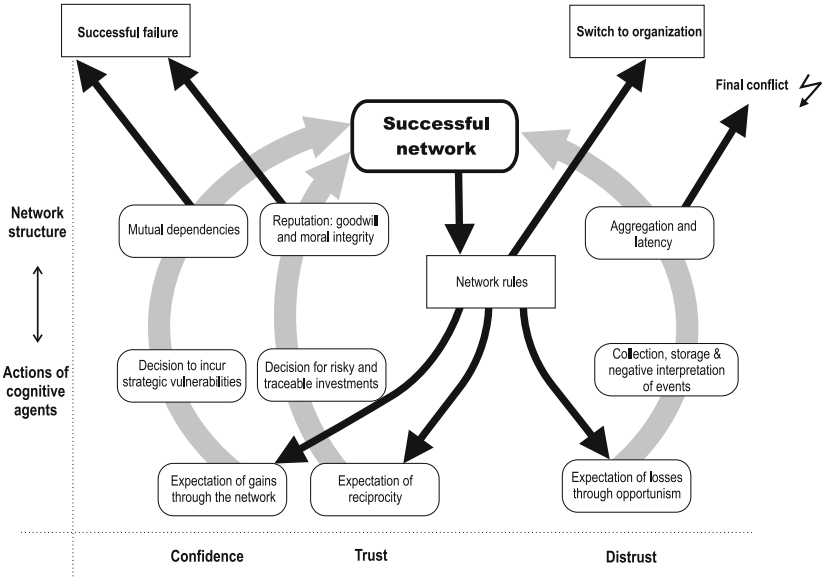


Fig. 1. Model for Success and Failure of Inter-Organizational Networks

simulation perspective using ConGolog [5]. The individual members' goals, abilities, and dependencies modeled in *i** are automatically transformed into (nondeterministic) ConGolog programs. Eventually, the SNet simulation environment adds basic agent facilities such as planning and communication.

A typical reason for applying such agent-based simulations – also here for the case of inter-organizational networks – is that a comprehensive understanding of the overall situation is not straight forward. Accordingly, the problem is decomposed into smaller, less complex components (agents). But obviously, we expect the interaction of many such agents to be non-trivial but helpful in regard to understanding the overall problem. Not surprisingly, the simulations themselves are then hard to analyze. Two major aspects of simulations intended to reflect real world behavior need to be considered. For one, we need to ensure that they correctly reflect the real world, this is validating them. For another, especially in complex multi stakeholder settings such as ours, support is needed to evaluate simulation runs and outcomes to advance the understanding of the investigated problem and support its governance. In this paper (based on [1]), we propose to investigate time series analysis, association rule mining, clustering, and social network analysis in regard to their usefulness for these purposes.

The paper is organized as follows. Section 2 introduces the foundations of modeling and simulating inter-organizational networks. Section 3 exhibits concrete validation as well as support questions. Then in Sect. 4, after discussing basic issues in regard to simulation data collection, the four kinds of analysis are investigated. We conclude with a discussion of related work as well as a summary and outlook on future work (Sect. 5).

2 Modeling and Simulating Inter-organizational Networks

2.1 Modeling with *i**

The *i** framework [21] is originally a language to support business process reengineering as well as requirements engineering. It includes the *strategic dependency (SD)* diagram for describing the network of relationships among actors and the *strategic rationale (SR)* diagram that we focus here for capturing the internal structure of an actor in terms of tasks, goals, resources, and softgoals. In previous work [9,10], we have extended *i** to cover additional operational details such as preconditions and effects of tasks and goals, sequence links, and a quantitative interpretation of the softgoal concept. These extensions alleviate the mapping to a simulation environment (ConGolog, see below) and thus to use *i** for modeling actors in agent-based simulations.

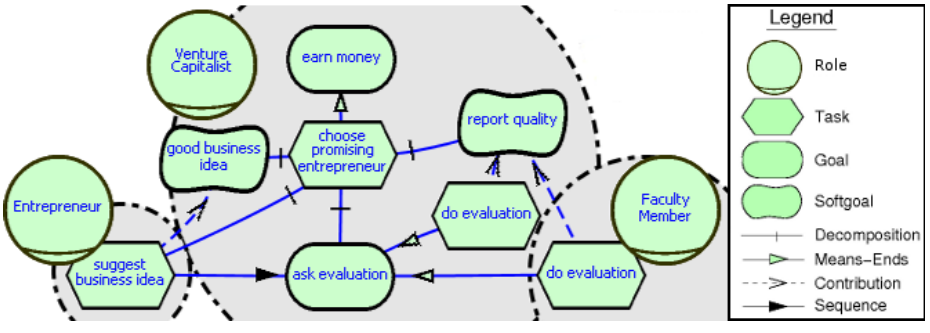


Fig. 2. Partial Strategic Rationale Diagram for a Venture Capitalist

Figure 2 shows an excerpt of the details of the "venture capitalist" participating in the entrepreneurship inter-organizational network. The venture capitalist's task "choose promising entrepreneur" as a means to "earn money" is decomposed to a subtask and a subgoal, ordered via a sequence link. The task "suggest business idea" is delegated to the "entrepreneur". "ask evaluation" is modeled as a goal to allow the modeled agent to choose between alternatives at runtime. In this example, the "venture capitalist" can either "do the evaluation" on its own or delegate this to a "faculty member". The softgoal "report quality" captures the quality of the assessment that might vary across different individuals that play the roles as well as across the two basic alternatives.

2.2 Simulation with ConGolog

ConGolog [5] is based on the situation calculus, a popular language for representing and reasoning about the preconditions and effects of actions [16]. It is a variant of first-order logic, enriched with special function and predicate symbols to describe and reason about dynamic domains. Relations and functions whose values vary from situation to situation are called *fluents*, and are denoted by predicate symbols taking a situation term as their last argument. Successor state axioms capture how fluents are affected

by actions. Furthermore, there is also a special predicate $Poss(a, s)$ used to state that action a is executable in situation s.

ConGolog is a language for specifying complex actions (high-level plans). For this purpose, normal imperative programming constructs like sequence, procedure, *if-then-else*, but also nondeterministic (e. g. *ndet*) and concurrency (e. g. *conc*) constructs are provided. ConGolog comes equipped with an interpreter (written in Prolog) which maps these plans into sequences of atomic actions assuming a description of the initial state of the world, the above mentioned action precondition axioms (*poss*), and successor state axioms. For details see [5].

The mapping from *i** we have proposed in [9,10] allows representing the behavior of agents by automatically creating a set of possibly nondeterministic procedures. For example, a complex task (e. g. "choose promising entrepreneur") is transformed into a procedure whereby the body is derived from the sub-elements, appropriately using sequential ([. . .]) or concurrency (*conc*) constructs. There are primitive actions preceding and following the body, so that the preconditions to and effects of this element can be reflected in the program. The transformation of goal elements and their fulfilling tasks is rather similar to the one of complex tasks, but the sub-elements are combined by the nondeterministic choice operator *ndet* to reflect the fact that one of these alternatives has to be chosen at runtime. Eventually, resources and softgoals are represented by fluents and precondition/effect elements are mapped to *poss* and *successor state* axioms, respectively.

2.3 The SNet Environment

The current SNet environment [10] supports the modeling in a role-based manner and provides an automatic transformation to ConGolog. The user can establish a concrete simulation by instantiating these roles and specifying initial settings (e. g. concrete trust relationships) and general characteristics (e. g. concerning risk attitude, trust orientation, or how much gain is provided as payment to delegation partners). Furthermore, the SNet environment equips each agent with a decision-theoretic deliberative planning component and communication facilities to cope with delegations and nondeterminism autonomously. The simulation user can influence an individual simulation run by triggering proactivities of agents (e. g. issuing that a "venture capitalist" wants to "earn (more) money"). This information can also be incorporated in a batch mode to run simulations without user intervention.

3 Validation and Support Questions

3.1 Validation Issues

As simulations are nowadays becoming the "third column of scientific investigation" (besides experiments and theory), the validity of the simulation model is gaining more and more importance. In particular in the field of control system development (CSD) [15], this is already widely accepted. Control system engineers use a detailed model of the controlled system to understand the control problem and only afterwards develop a suitable

controller. But before the model can be used, it has to be ensured that it correctly reflects the real world. Thus, expensive testbed experiments are run to validate and calibrate a model, in particular since simplifications of the precise mathematical models are needed for computational feasibility.

Unfortunately, validating social simulations is less easy, since the corresponding "real world" cannot simply be "connected" to a testbed. Table 1 gives a (non-exhaustive!) list of questions that we think are helpful in deciding about the validity of simulations of inter-organizational networks. Unsurprisingly, the focus is on core aspects mentioned in the introduction, in particular trust: whether the agents behave according to the intention of specified parameters (e. g. risk-averse, trust-oriented, etc.) (V1), whether trust is respected (V2), whether trust and cooperation experiences co-evolve (V3), and regarding the robustness against the specification of the initial setting. This is how relevant are the initial settings or do (minor) adjustments have undesirably large impacts (V4)? Similar to CSD, no behavior is taken for granted only because the model was designed to exhibit it. Just as the mathematical equations used to simulate a controlled system can turn out to be inadequate (e. g. too imprecise), a chosen trust model or cooperation protocol can be insufficient or have flaws that are not visible at a first glance.

Table 1. Questions to Validate Simulations of Inter-Organizational Networks

No.	Validation Questions
V1	Do members act in line with their characterizing parameters?
V2	Does trust have an influence on the choice of partners?
V3	Does the evolution of trust between partners correlate with cooperations?
V4	How relevant are initial settings?

3.2 Managerial Support Issues

Support for understanding and governance is in particular needed in regard to trust as well as the interplay between individual level and network level mediated via network rules (see Fig. 1). The simulation environment allows to try out different alternatives for governance before applying them to a real world setting. Relevant questions that could probably be answered via suitable analyses of simulation runs are summarized in Tab. 2.

Regarding individual behavior and concrete trust based interactions of agents (abbreviated by "I"), sets of actual cooperation partners can be analyzed in regard to whether they have similar characteristics (e. g. all are trust oriented) (I1), have a tendency to build strong ties, this is Mafiose-like structures emerge that favor insider deals within a small group over rational choices of partners (I2), or do not really fit together thereby indicating a lack of alternatives that needs to be addressed by opening up the network towards new members with redundant capabilities (I3). Also the attempt to detect free-riders that only work for their own profit not interested in the long-term success of the network is included in this part of the list (I4).

Table 2. Support Questions for Inter-Organizational Networks

No.	Support Questions
I1	Do trust-oriented partners find each other?
I2	Do Mafiose-like structures emerge in a network?
I3	Do partners cooperate despite missing trust (lack of alternatives)?
I4	Is it possible to detect free-riders?
N1	Is there a network rule that is ignored/circumvented by a majority of members?
N2	How is a new network rule adopted by the members?
M1	Any long-term positive effect of distrust-based monitoring?
S1	What are suitable measures for network (individual member, resp.) success?
S2	Do members that adhere to network rules benefit?
S3	How equally do the network members profit from the network?
S4	How does a modified set of network rules affect the overall network situation?

Regarding network rules and monitoring ("N" and "M", respectively), it is of interest to detect rules that are circumvented by a majority of members, indicating that the rule might no longer be necessary or has even become hindering (N1). Furthermore, network life-cycle investigations [17,19] suggest that the introduction of new rules to overcome shortcomings of a current network situation has a dynamic dimension of its own. The community of network members needs to commit to the new rules and adopt them. This might not happen immediately, but gradually, potentially depending on the role (and standing) of "early-adopters" (N2). Eventually, the role of distrust and monitoring needs a closer investigation. If distrust is really helpful, it must be possible to return from a high-level of distrust (resulting in expensive monitoring activities) to a moderate, even trustful cooperation (M1).

Eventually, in regard to success ("S") first of all, suitable criteria to measure success of individual members as well as the network as a whole have to be identified, for example, the number of (successful) cooperations, the overall level of trust (and distrust) in the network etc. (S1). These criteria can then be consulted to investigate whether it is really advantageous for a member to follow the rules that a network has given itself (S2). Similarly, the benefit for different members can be compared in order to identify inequalities, for example, only a small "elite" of members benefits from the network, that can cause the network to fall apart (S3). Certainly, the success criteria are also essential to decide about the suitability of changes to network rules or a need thereof, for example, to increase the overall level of trust in the network (S4).

4 Simulation Analysis

4.1 Data Collection

As a prerequisite for running more advanced analysis on agent-based simulations, the according simulation data must be accessible. In particular the user has to specify what data to collect. The foundation of our simulation approach in the situation calculus makes this easy. As introduced previously, relational and functional fluents represent the current state of the world. Thus, the user only has to specify the fluents that are to

be logged. Then each time a primitive action is executed the corresponding successor state axioms are checked. If a relevant fluent is affected, the execution engine creates the according output. A flat, XML-based logging format is flexible enough to feed any tool, in particular due to the applicability of standard query and transformation languages such as XPath and XSLT. They allow for selecting, updating, and aggregating data [11] even after a simulation has been run. Furthermore, the format is defined in a way that allows for continuous analyses in parallel to long running simulations. This enables the detection of interesting intermediate situations for which branching could be lucrative.

Figure 3 shows an excerpt of the recorded information regarding trust issues and cooperation details in our simple entrepreneurship setting for two points in time. Cooperations are in fact special since for them relevant information arises at various points in time. For example, the trust relationships at the time of planning might be of interest as well as which agent did not offer its participation or was not selected. Later on, the success (or cancellation) of the cooperation needs to be recorded as well as the (possibly execution-dependent) effect on trust and gain (if applicable).

```
<simulation_data> ...
  <time value="38">
    <cooperation agent="A" role="VC" activity="earn money" pid="ID53"...>
      <partner role="Entrepreneur" delegated="suggest business idea"
               val="B"/>
      <partner role="Faculty Member" delegated="do evaluation" val="C"/>
    </cooperation>
  </time>
  <time value="42">
    <cooperation_result from="A" to="B" role="Faculty Member"
                        task="do evaluation" pid="ID53" val="OK">
    <trust from="A" to="B" for="do evaluation" val="0.5"/>
  </time>...
</simulation_data>
```

Fig. 3. Excerpt of Logged Simulation Data

In the following for each of the four kinds of analysis, we introduce its basic features and existing tool support. Afterwards we discuss its applicability to the validation and support questions identified in Sect. 3.

4.2 Time Series Analysis

Time series analysis [11] concerns a sequence of values or events with usually fixed time intervals. Typical application examples are stock quotations or data from medical treatments. Series can then be investigated, for example, in regard to the occurrence of trends, cyclic, seasonal, or irregular movements or in regard to the similarity of the whole sequence or parts thereof.

Time Searcher 2 (http://www.cs.umd.edu/hcil/timesearcher/) from the University of Maryland is a freely available tool for interactive querying and exploration of time series data. It is able to cope with long time series and can present multi-variate time

series data via simultaneous plots. A "search box" allows for finding similar patterns throughout the data. The input format (TQD) is simply text-based containing various pieces of information such as variables to be investigated, the list of time points and of course the concrete data. Attribute information can be added separately to support filtering. From our log, the corresponding information can be generated automatically. For details see [1].

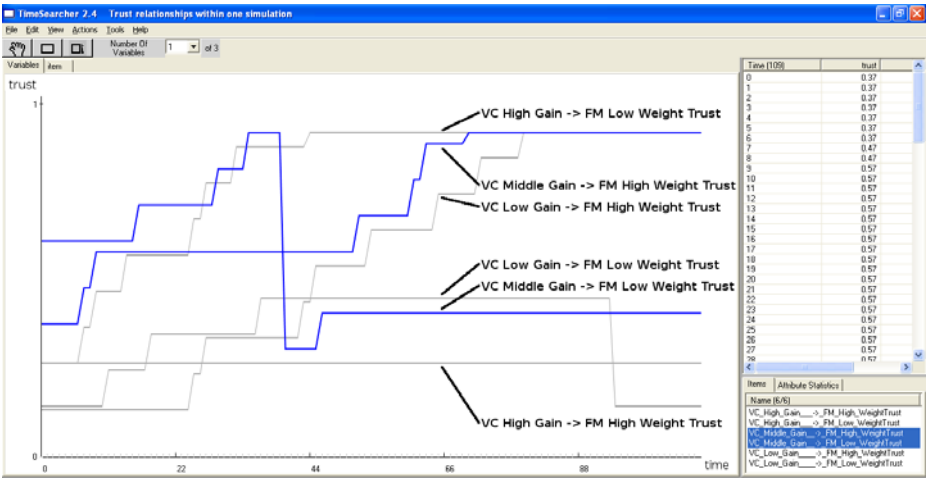


Fig. 4. Time Series Analysis: Trust for Six Delegations

Due to the focus on dynamic issues time series analysis has the broadest span of application in our setting. In order to decide whether agents adhere to their parameter settings (V1), e. g. risk averse behavior manifests itself by accepting mainly trusted collaboration partners, the relevant cooperation features (e. g. trust) need to be inspected over the whole period of simulation. For one, the actual behavior of similarly or differently parameterized agents can be compared to each other. Figure 4 shows exemplarily the evolution of six trust relationships[1] between instances of the "venture capitalist" and instances of the "faculty member" within the entrepreneurship setting over time. Three "venture capitalists" providing "high", "middle", or "low gain" need to get their evaluation reports done by either a "faculty member" that puts a "high weight on trust" or a "low weight", this is she is more (or less) gain oriented. After recording a successful cooperation, the trust value grows by a certain delta until the maximal level has been reached. Correspondingly, an observed cooperation failure causes the trust level to drop (faster than the increase in case of success). As to be expected, we can see that the "faculty member" with a "low weight on trust" prefers the "venture capitalist" that provides "high gain" whereas the "venture capitalists" that provide "low" or "middle gain" strengthen their cooperation with the more trust oriented "faculty member" over time.

[1] We consider trust a subjective probability thus ranging only between 0 and 1.

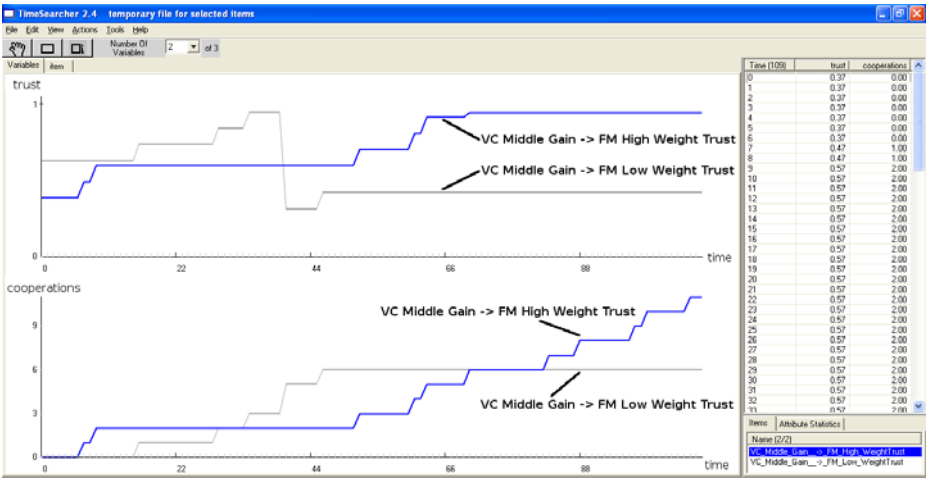


Fig. 5. Time Series Analysis: Trust vs. # Cooperations

For another, noticeable deviations between the time series for level of trust and number of cooperations can be used as indicators for inconsistencies. Figure 5 shows two curves for trust relationships in the upper part (both involving the "venture capitalist" that provides "middle gain" at the one side and then the two "faculty members" ("low weight trust" vs. "high weight trust") at the other) and the corresponding total number of (successful or failed) cooperations in the lower part. Usually, it is to be expected that related curves in the two graphs grow in accordance to each other as it is the case for the darker curves in the figure ("venture capitalist (middle gain)" cooperating with "faculty member (high weight for trust)"). A deviation can indicate a bug concerning the co-evolution (V3) or can identify a lack of cooperation alternatives (I3). To elaborate the latter, even though a rather low level of trust occurs repeatedly, the delegating agent continues to cooperate with this untrusted partner since there is simply no other option. Figure 5 shows that the "venture capitalist" has again (T=46) chosen the "faculty member" with a "low weight of trust" despite of severe negative experiences (T=40). The relevant situation needs then to be investigated further for the reasons. In this case the planning for the activity has taken place at a very short notice and thus the other "faculty member" simply was not available due to an already high work load. Furthermore due to her trust-oriented behavior, she did not consider to cancel any of her current activities.

The effect of different initial settings (V4), for example, in regard to trust, can certainly be investigated by comparing time series of simulation runs that only deviate in this regard. In this context it could also make sense to abstract from all details by considering suitably aggregated trust measures, e. g. combining the relationships at role level. This is instead of analyzing individual trust relationships between pairs of agents we compute one trust value that combines all trust relationships between, for example, all instances of "venture capitalist" and all instances of "faculty member". Similarly, a data set that aggregates all trust relationships within a model could be used to investigate the effect of a modified set of network rules (S4), in particular if it targets the

growth of trust relationships. Discontinuities in the evolution of trust over time could be considered, if network rules have been changed during a simulation run.

Time series are also helpful to identify criteria that indicate success (S1). After taking expert advice on which simulation run or network situation can indeed be classified as "successful", graphs of monitored characteristics can be consulted to identify a set of features (and particular values) that correctly and reliably characterizes success. Again validation of these findings is of importance. Given such criteria are available, the success of individual members can be compared. By grouping agents, for example, into ones that respect rules and ones which do not, it is possible to check whether conforming to rules is beneficial (S2). Similarly, it can be checked which agents benefit more from the network than others (S3).

By relating gain evolution and cooperation cancellations, it might even be possible to detect free-riders (I4). And by evaluating the evolution of trust after distrust-based monitoring has taken place, the chances for returning to a normal trustful cooperation relationship can be investigated (M1).

4.3 Association Rule Mining and Clustering

Association rule mining [11] tries to find frequent patterns in sets of data objects (originally in databases). Classical application field is a market basket analysis as any online dealer is performing nowadays to find out which products are most often sold together. The analysis has two steps. In the first step, all frequent itemsets above a prespecified threshold (minimal support) are computed. Then for each subset s of a frequent itemset l a rule is created $s \Rightarrow (l - s)$, but only if the confidence of the rule is above another threshold. Various algorithms exist to compute the rules efficiently.

Similarly, clustering groups (multi-dimensional) data objects without presuming any classification label by the "principle of maximizing the intraclass similarity and minimizing the interclass similarity" [11]. This facilitates the organization of observations. For example, online shops use cluster analysis to identify target groups for marketing.

There is consequently some kind of overlap with association rule mining, even regarding the tool. RapidMiner (http://www.rapidminer.com) is capable of executing a large set of analyses from the field of data mining including association rule mining and clustering. It expects two proprietarily formatted text files as input. The AML file contains the description of the attributes of the data objects (name, range of values, etc.) and the DAT file the actual values for the specified attributes. Accordingly, both kinds of analyses, association rule mining as well as clustering, are applicable to similar questions. To reduce resulting redundancy, for each question we only discuss the application of one of these analyses.

To validate whether agents act according to their parameterization (V1), the set of all cooperations can be investigated. The RapidMiner needs to be fed with the pairs of cooperation partners. The data set can even be enlarged by considering the decomposition structure within the foundational *i** model and including all partners that ultimately contribute to a proactivity of some agent. Due to the properties of association rules the latter setting subsumes the former. The according data is easily retrievable from our simulation data since for each proactivity a unique identifier (PID) is issued that is propagated throughout all (sub-)delegations. It is also possible to add further details in regard to the

result of the cooperation (success vs. failure) or a categorization of the trust relationship (high, middle, low). For example, for a trust-oriented partner association rules should indicate that in the majority of cooperations trusted partners are chosen. Thereby, such an analysis is also able to show that trust is in fact respected during choice of partners (V2).

Table 3. Association Rules regarding Cooperations

Venture Capitalist	⇒	Faculty Member	Confidence
agent providing low gain	⇒	agent with low trust weight	0.32
low gain	⇒	high trust weight	0.68
middle gain	⇒	low trust weight	0.46
middle gain	⇒	high trust weight	0.54
high gain	⇒	low trust weight	0.52
high gain	⇒	high trust weight	0.48

In regard to the validation of the initial setting (V4), Tab. 3 shows some results computed with RapidMiner again on the simple delegation relationship between "Venture Capitalist" and "Faculty Member". The configuration includes three agents playing the "Venture Capitalist" role differentiated by providing low, middle, and high gain to delegatees as well as two agents playing the "Faculty Member" role that vary in regard to how important trust (and accordingly gain) is for them (low vs. high trust weight). These analyses have revealed the dominating role of the initial trust relationship settings, at least for a simplified setting where overlapping jobs have not been admitted. Under these constraints the confidence value nearly identically reflects the initial values. Thus, for more interesting insights more complex simulations need to be run. In addition aggregated measures across several simulations might be needed to remedy the influence of the initial setting.

Whether similar agents find each other (I1) or cooperate only locally (I2) can be detected by clustering. Again cooperations are the foundations but additionally the relevant characteristics, e. g. trust orientation, need to be added. The emerging clusters can then be analyzed in regard to their regional size, this is how narrowly related, i. e. similar, are clustered items, as well as their pure quantitative size, this is how many items belong to one cluster. The first measure reflects the diversity of agents that cooperate and the second the flexibility in regard to cooperation partners and thus provide answers to the questions I1 and I2, respectively. Similarly, outliers in regard to clustering can indicate a lack of alternatives (I3). The detection of free-riders (I4) is less reliable. An idea would be that such an agent only chooses cooperations that ensure the highest gain, while cooperations with lower gains are cancelled more often. Clustering (or building association rules) in regard to these characteristics could at least indicate candidates, even though a free-rider can "hide" her behavior more intelligently.

The confirmation to network rules (N1) and their adoption (N2) can be addressed straight forwardly by looking for association rules that include a check on whether an agent utilizes a rule or not. Being considered a frequent item step is then already the first hurdle. If a network rule fails that hurdle, it might need to be abandoned. In order

to detect important stakeholders, during analysis one could watch out for association rules that consider members early involved in the adoption of a new rule that later on turns out to be successful.

Eventually, also success related issues can be addressed. For example, if some characteristic criteria have been determined by experts, a corresponding clustering according to these criteria can be carried out. If agents are clustered together that are not all successful, this clearly indicates that the chosen characteristics are not yet sufficient (S1). An association rule can also presume network rule compliant behavior and then check whether successful agents are on the majority (S2). Finally clustering according to (individual) success gives a fast overview of inequalities in the network (S3).

4.4 Social Network Analysis (SNA)

To address patterns more complex than sets (see association rules) and sequences, graph-based approaches are applied increasingly [11]. Most importantly networks that result from the investigation of various kinds of relationships between objects are of interest. Usually, a numerical value is given as a weight to the link between two graph nodes reflecting the strength of the relationship. The most basic form of analysis provided by such an approach is to visualize the network in order to reveal its induced internal structure. yFiles (http://www.yworks.com) and JUNG (http://jung.sourceforge.net) are two well-known tools that allow for such kind of analysis. They also provide additional algorithms, for example, to compute different types of centrality: degree centrality as a local measure of the connectedness of a node, closeness centrality in contrast as a global measure indicating the distance to any other node in the graph, and betweenness centrality relating the shortest path a particular node is involved in to the overall shortest path. The latter measure thereby is a strong indicator for the influence of that node. Other relevant kinds of analysis investigate the cohesion in a graph, in particular cliques, this is subsets of nodes where each node is connected to each other. With GraphML (http://graphml.graphdrawing.org), an agreed data format is available that alleviates integration.

Since the objects of our investigations are in itself networks, social network analysis obviously is of interest. As kind of relationship trust, distrust, or simply the number of cooperations can be considered. Such analyses have already been applied to statically analyze *i** models [14]. Accordingly, we can certainly apply them to starting, intermediate, or final situations. A match between a network in regard to trust with a network in regard to cooperation can provide an answer to whether similar agents find each other (I1). Outliers to such a matching can indicate at a lack of alternative partners as could a suitable centrality measure since the corresponding agent needs to be involved in most of the related cooperations (I3). A high density of cooperations between a limited set of members – also detectable via a suitable cohesion measure – hints at potentially Mafiose-like structures (I2).

To investigate the adoption of new rules (N2), connections can be drawn that reflect precedence when accepting a new rule. Betweenness centrality can then indicate important stakeholders that have influenced success (or failure) of rule adoption. Also

whether the cooperations in an inter-organizational network are equitably distributed across the whole network can quite easily be derived from a visual representation as well as the computation of cohesion such as cliques and thus become a part of the set of success criteria (S1).

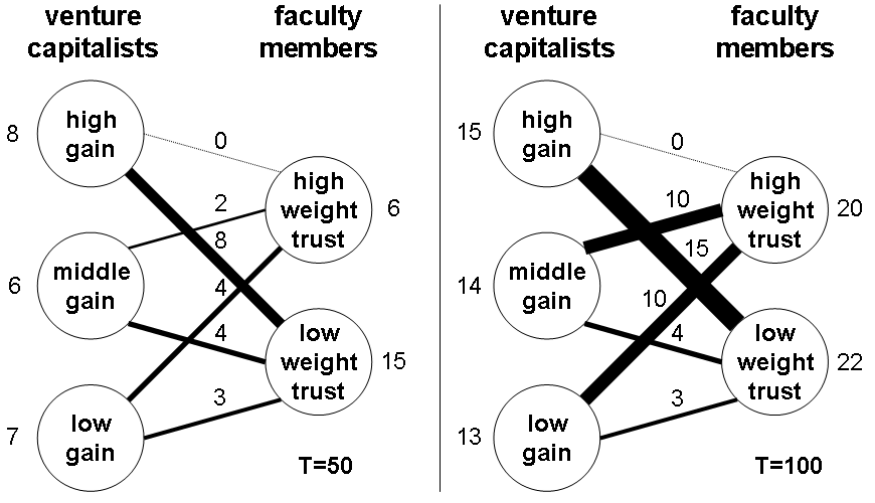


Fig. 6. Network of Cooperations for Two Different Points in Time

Even more valuable seems the extension towards dynamic social network analysis [3] simply due to the fact that we run simulations. How the actually exhibited cooperation network evolves over time in particular in response to network rule changes (S4) is of great interest. While unfortunately, a connection to existing tool support at the chair [2] has not yet been established, for small examples such as the entrepreneurship example investigated here, a manual capture of the evolution of the relationships is possible. Figure 6 shows the network of cooperations between different "venture capitalists" (providing high, middle, or low gain) and different "faculty members" (high weight for trust vs. low weight for trust) for two different points in time within a single simulation run (T=50 and T=100). The thickness of the connecting lines indicates the number of successful cooperations (also given as exact numbers) whereas the numbers at the different agents denote the total number of cooperations the agent is involved in. The comparison reveals that despite advantages for the "faculty member" with "low trust weight" in regard to the initial trust settings, the "venture capitalists" that provide only "middle" and "low gain" experienced the unreliability of this partner and thus switched to the alternative partner (with "high trust weight"). On the other hand, the "venture capitalist" providing "high gain" does not experience any problems since the "faculty member" with "low weight trust" is preferring her due to the high payment.

It is important to remark that when analyzing network structures it is necessary to take into account which structures are induced by the modeling in particular by the role-based relationships defined in *i**. For the example shown in Fig. 6, we do not see

but also cannot expect cooperations between "venture capitalists" or between "faculty members" simply because the *i** model (in Fig. 2) does not introduce such a relationship. The situation becomes more complex in a more sophisticated setting where an agent is involved in several roles, for example, a successful "entrepreneur" that starts to work also as a "business angel" for new entrepreneurs. An analysis focusing agents (instead of particular roles) could then reveal how the "entrepreneur" utilizes existing relationships for the new activities. This has to be investigated more deeply in future work.

5 Discussion

While agent-based simulation tools such as Repast Simphony (http://repast. sourceforge.net) nowadays provide a rather large set of interfaces towards advanced analysis means (e. g. WEKA, Matlab, JUNG), for logic-based simulations the situation is less comfortable. In particular the usage of ConGolog is mainly in the field of agent programming rather than simulations, thus debugging and program execution visualization [5] have been focussed. For *i** models on the other hand, a lot of analysis means is available [8, 12]. But they mostly focus static issues or are founded in formalisms with severe restrictions in regard to dynamic considerations such as the number of instances that can be considered in model checking-based approaches [18].

In this paper we have presented basic ideas on how to apply more sophisticated analysis means to ConGolog-based simulations. Major steps towards implementation have already been taken, but more complex settings have not yet been analyzed. As the summary in Tab. 4 reveals, we were able to identify helpful analysis for each of the raised questions. But as it can also be seen (and possibly also had to be expected) the mapping is not simply one to one. Accordingly, the need arises to compare, cross-validate, and/or combine the results of several analyses depending on whether they address different aspects of a question or not.

Table 4. Matching Analyses to Questions

	Validation				**Individual**				**NWR**			**Success**			
	V1	**V2**	**V3**	**V4**	**I1**	**I2**	**I3**	**I4**	**N1**	**N2**	**M1**	**S1**	**S2**	**S3**	**S4**
Time series analysis	x		x	x			x	x			x	x	x	x	x
Assoc. rules/Clustering	x	x		x	x	x	x	x	x	x		x	x	x	
Social network analysis					x	x	x			x		x			x

For future work, we thus do not only have to run larger experiments. If some question has been identified to be of general interest when investigating inter-organizational networks, the relevant analyses should be bundled and automated as a tool functionality, including as much integration information, e. g. from the common formalism base in *i** and ConGolog, as possible. Furthermore, we can distinguish two different settings where such analyses make sense. At early analysis stages – this is where the interesting features of the subject of investigation are not yet fully understood (as it is the case here) – mostly qualitative analyses are relevant. These then aim at growing an understanding

of the working mechanisms and driving forces within a setting. For example, does an expected subnetwork really come to existence? Or the more open question what kind of subnetworks could possibly emerge? If at a later point in time, the analyses and accompanying tool is used to provide online support to the management of a real-world network, we can expect a shift towards more quantitative analyses that better serve as input to management decisions (similar to [4]). Accordingly, we then need to adapt the basic analyses that are provided as well as the treatment of simulation settings and runs in order to compute such "hard" metrics.

In addition, the usefulness of the analyses for other applications of ConGolog could be investigated. While the analyses currently presume some specific features to be part of the world formalization, for example, a discrete model of time and the explicit mention of which agent is executing a primitive action, the basic principle of observing the evolution of fluents is applicable to any ConGolog setting. Thereby, it could be possible to watch out for applications of these analyses to the control of robots, for example, in the RoboCup competition (http://www.robocup.org). A combined investigation of real world sensor data and the agent"s internal world model could support existing approaches (e. g. [6]) in detecting recurring behavior of opponents to develop response strategies, e. g. counterattacks.

Acknowledgment. This work was partially supported by the Deutsche Forschungsgemeinschaft within its Graduate School 643 and the German Excellence Initiative within the IMP Boost OBIP project.

References

1. Arzdorf, T.: Analysis support for agent-based simulations of inter-organizational networks (in German). Master's thesis, RWTH Aachen University (2008)
2. Cao, Y., Klamma, R., Spaniol, M., Leng, Y.: A toolkit to support dynamic social network visualization. In: Qiu, G., Leung, C., Xue, X.-Y., Laurini, R. (eds.) VISUAL 2007. LNCS, vol. 4781, pp. 512–523. Springer, Heidelberg (2007)
3. Carley, K.M.: Dynamic network analysis. In: Carley, K., Breiger, R., Pattison, P. (eds.) Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers, pp. 133–145. Comittee on Human Factors, National Research Council (2003)
4. Chaturvedi, A.R., Mehta, S.R.: Simulations in economics and management. Communications of the ACM 42(3), 60 (1999)
5. de Giacomo, G., Lespérance, Y., Levesque, H.J.: ConGolog, a concurrent programming language based on the situation calculus: language and implementation. Artificial Intelligence 121(1-2), 109–169 (2000)
6. Dylla, F., Ferrein, A., Lakemeyer, G., Murray, J., Obst, O., Röfer, T., Schiffer, S., Stolzenburg, F., Visser, U., Wagner, T.: Approaching a formal soccer theory from behaviour specifications in robotic soccer. In: Dabnichki, P., Baca, A. (eds.) Computers in Sport, pp. 161–185. WIT Press, Southampton (2008)
7. Ellrich, L., Funken, C., Meister, M.: Kultiviertes Misstrauen. Bausteine zu einer Soziologie strategischer Netzwerke. Sociologia Internationalis 39(2), 23–66 (2001)
8. Fuxman, A., Liu, L., Pistore, M., Roveri, M., Mylopoulos, J.: Specifying and analyzing early requirements in Tropos. Requirements Engineering Journal 9(2), 132–150 (2004)

9. Gans, G., Jarke, M., Kethers, S., Lakemeyer, G.: Continuous requirements management for organization networks: A (dis)trust-based approach. Requirements Engineering Journal 8(1), 4–22 (2003)
10. Gans, G., Jarke, M., Lakemeyer, G., Schmitz, D.: Deliberation in a metadata-based modeling and simulation environment for inter-organizational networks. Information Systems 30(7), 587–607 (2005)
11. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
12. Horkoff, J., Yu, E.S.K.: Qualitative, interactive, backward analysis of i* models. In: 3rd i* Workshop, CEUR Workshop Proceedings, Recife/Brazil, February 11-12, vol. 322, pp. 43–46. CEUR-WS.org (2008)
13. Jarke, M., Klamma, R., Marock, J.: Gründerausbildung und Gründernetze im Umfeld technischer Hochschulen: ein wirtschaftsinformatischer Versuch. In: Zu den Wirkungen des regionalen Kontexts auf Unternehmensgründungen, pp. 115–154. EUL-Verlag (2003)
14. Klamma, R., Spaniol, M., Denev, D.: Paladin: A pattern based approach to knowledge discovery in digital social networks. In: I-KNOW, Graz/Austria, September 6-8, pp. 457–464. Springer, Heidelberg (2006)
15. Lunze, J.: Automatisierungstechnik. Oldenbourg (2003)
16. McCarthy, J.: Situations, actions and causal laws. Technical report, Stanford (1963); Reprinted in Minsky, M. (ed.): Semantic Information Processing. MIT Press, Cambridge (1968)
17. Provan, K.G., Kenis, P.: Modes of network governance and implications for network management. Research Colloquium Tilburg University (2005)
18. Schmitz, D., Lakemeyer, G., Jarke, M.: Comparing three formal analysis approaches of the Tropos family. In: Kolp, M., Henderson-Sellers, B., Mouratidis, H., Garcia, A., Ghose, A.K., Bresciani, P. (eds.) AOIS 2006. LNCS (LNAI), vol. 4898, pp. 164–182. Springer, Heidelberg (2008)
19. Straßheim, H.: Power in intercommunal knowledge networks. Discussion Paper SP III 2004-104. Wissenschaftszentrum Berlin für Sozialforschung (2004)
20. Weyer, J.: Soziale Netzwerke. Oldenbourg Verlag, München (2000)
21. Yu, E.: Modelling Strategic Relationships for Process Reengineering. PhD thesis, University of Toronto (1995)

Clustering in a Multi-Agent Data Mining Environment

Santhana Chaimontree, Katie Atkinson, and Frans Coenen

Department of Computer Science
University of Liverpool, UK
{S.Chaimontree,katie,Coenen}@liverpool.ac.uk

Abstract. A Multi-Agent based approach to clustering using a generic Multi-Agent Data Mining (MADM) framework is described. The process use a collection of agents, running several different clustering algorithms, to determine a "best" cluster configuration. The issue of determining the most appropriate configuration is a challenging one, and is addressed in this paper by considering two metrics, total Within Group Average Distance (WGAD) to determine cluster cohesion, and total Between Group Distance (BGD) to determine separation. The proposed process is implemented using the MASminer MADM framework which is also introduced in this paper. Both the clustering technique and MASminer are evaluated. Comparison of the two "best fit" measures indicates that WGAD can be argued to be the most appropriate metric.

Keywords: Agent-Based Clustering, Multi Agent Data Mining.

1 Introduction

In this paper we describe a multi-agent based approach to clustering, that harnesses the processing power of a collection of "clustering" agents, to produce a "best" set of clusters given a particular clustering problem. The motivation for the work is that there is no clear, general purpose, best clustering algorithm suited to all data. It is suggested that a Multi-Agent System (MAS) based approach provides a good solution to the generic problem of finding a best set of clusters. The approach allows for a collection of clustering agents to collaborate to produce a "best" cluster configuration. This entails a number of research challenges. The first is the operation of the desired MAS based clustering, i.e. the nature of the MAS environment, and the coordination and communication mechanisms to be used. The second is how to define what is meant by a "best" cluster configuration, and then use this definition within a MAS framework.

To address the first issue the research team have developed a Multi-Agent Data Mining (MADM) environment, called *MASminer*. MASminer is still under development and is designed to support generic forms of data mining, however a demonstration version is in operation. MASminer has been built on earlier work directed at generic MADM, such as EMADS [1]; but is distinctive in that it uses an ontology based communication mechanism, *OntoDM* as opposed to the "hard coded" *utterances* required by systems such as EMADS. A full description of the MASminer architecture is included in this paper.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 103–114, 2010.

The second issue is more difficult to address. The accuracy of a set of clusters can of course be evaluated by comparing the derived results with a set of known results as in the case of supervised learning. However, this requires provision of pre-labelled data so that a *training* set can be presented to the clustering system, from which labelled clusters can be generated, which can then be evaluated using a *test* set. The pre-labelling of data entails an undesirable overhead, unsupervised learning is therefore more desirable. However, in this case suitable measures must be adopted to determine the appropriateness of a generated cluster configuration. Two measures are considered and compared in this paper, total Within Group Average Distance (WGAD) and total Between Group Distance (BGD). The main contributions of this paper are thus as follows:

- A generic approach to multi-agent based clustering to identify a best set of clusters using a collection of "clustering agents".
- An overview of MASminer, a generic MADM environment that uses a "bespoke" data mining ontology, OntoDM.
- A comparison of two measures, WGAD and BGD, to identify the most appropriate set of clusters for a given clustering problem.

The rest of this paper is organised as follows. A review of previous work conducted in the field of generic MADM and multi-agent based clustering is presented in Section 2. The WGAD and BGD measures are presented in Section 3. The MASminer environment, together with an overview of OntoDM, is introduced in Section 4; and the multi-agent based clustering process in Section 5. The use of the WGAD and BGD measures, are evaluated in Section 6. Section 7 then presents some conclusions.

2 Previous Work

There are a number of reported MADM systems in the literature. These are mostly directed at specific data mining tasks or applications. One example is that of Baazaoui Zghal et al. [2] who developed a MADM directed at geographic data. The objective was to generate a Knowledge Base (KB), using data mining processes, and then to use this KB to support decision making by end users.

A number of agent-based approaches directed at clustering have also been reported in the literature. The earliest reported systems are PADMA [3] and PAPYRUS [4]. The aim of these systems is to achieve the integration of knowledge discovered from different sites with a minimum amount of network communication and a maximum amount of local computation. PADMA is used to generate hierarchical clusters in the context of document categorisation. Agents are employed for local data accessing and analysis. All *local clusters* are collected at the central site to generate the *global clusters*. PAPYRUS [4] is a clustering MAS where both data and results can be moved between agents according to given MAS strategies.

A more recent MAS approach to clustering is the KDEC scheme proposed by Klusch et al. [5]. KDEC is a distributed density-based clustering algorithm. Reed et al. [6] proposed a MAS for the distributed clustering of text documents which assigns new, incoming, documents to clusters. The objective was to improve the accuracy and the relevancy of information retrieval processes. Cen et al. [7] describe an eCommerce

application of clustering founded on MADM. Cen et al. used the well know Apriori algorithm to analyse the interests of eCommerce WWW site users with respect to the characteristic of the website and the time spent at the website. A clustering mechanism was then used to group users according to this analysis. To the best knowledge of the authors the most recently reported MAS clustering system is JABAT [8]. JABAT is a MAS for both distributed and non-distributed clustering (based on the K-means algorithm). JABAT is of note in the context of this paper because it also uses ontologies to define the vocabularies and semantics for the content of message exchange among agents. The distinguishing feature between these systems and the system described in this paper is that they are tied to a particular clustering technique.

There are very few reported examples of generic MADM systems. EMADS (the Extendible Multi-Agent Data Mining Framework) [1] is one example. The objective of EMADS was to provide an easily extendable framework which can integrate new DM techniques and data sources in a distributed infrastructure and collaborative environment. From the literature EMADS has been demonstrated using two data mining scenarios: distributed ARM and Classification. The best classifier fitted to a particular data set is identified in the second scenario. A fixed protocol is used in EMADS to facilitate shared agent understanding, whereas MASminer uses a more accessible ontology based approach in order to achieve generic MADM.

3 Quality Measures

Different clustering algorithms provide different cluster results depending on the characteristics of the data set and the input parameters for defining groups (clusters). The validity of a generated cluster configuration can be evaluated in a number of ways. Two of the most popular are *cohesion* and *separation* [9]. Cohesion is used to measure the compactness ("tightness") of clusters. Separation is a measures of the distinctiveness of a cluster with respect to other clusters.

There are a number of techniques where by cohesion and separation can be calculated, these include *density-based*, *graph-based* and *prototype-based* approaches. For example, using the graph-based approach the cohesion of clusters is defined as the sum of the weights of the links among points in the cluster. Separation is then measured by computing the sum of the weights of the links from points in one cluster to points in other clusters. In the work described here the prototype- (centroid- or medoid-) based approach has been adopted because it has a lower computational overhead in the context of the proposed approach. Using the prototype-based approach the cohesion of a cluster is measured in terms of the sum of the weights of the links from the prototype to points in the cluster. The separation between two clusters is then measured in terms of the sum of the weights of the links among the prototypes of two clusters. This is illustrated in Figure 1 where an asterisk ($*$) represents a centroid of a cluster.

More specifically, in this paper, total Within Group Average Distance (WGAD) is used to determine cohesion [10], and total Between Group Distance (BGD) to determine separation. WGAD is the sum of the average distance of a cluster centroid, c_i, to each data point (x) in the cluster. The lower the WGAD the greater the compactness

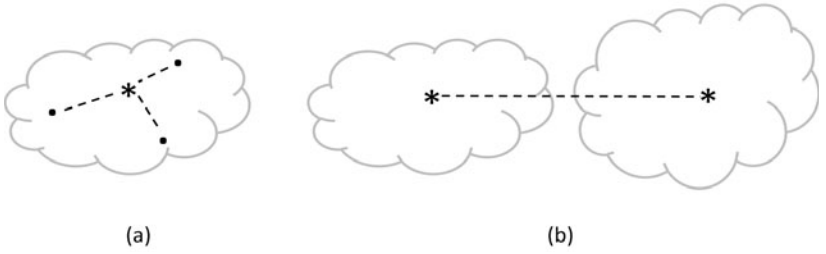


Fig. 1. (a) Prototype-based Cohesion (b) Prototype-based Separation

(cohesiveness) of the cluster. The total WGAD of a given cluster configuration is defined as:

$$Total\ WGAD = \sum_{i=1}^{i=K} \frac{\sum_{j=1}^{j=m_i} dist(x_j, c_i)}{m_i} \quad (1)$$

where K is the number of clusters and m_i is the number of data points in cluster i.

BGD is then the sum of the distance of each cluster centroid, c_i, to the overall centroid, c. The higher the total BGD of a cluster configuration the greater the separated of the clusters from one another. The total BGD of a given cluster configuration is defined as:

$$Total\ BGD = \sum_{i=1}^{i=K} dist(c_i, c) \quad (2)$$

Thus, to identify a "best" cluster configuration, we wish to minimise the WGAD and maximise the BGD to achieve a best degree of cohesion and separation.

4 MADM Framework

MASminer is designed to be a generic MADM environment. As such it comprises five categories of agent: (i) User Agents, (ii) Task Agents, (iii) Data Mining (Clustering) Agents, (iv) Data Agents, and (v) Validation Agents. User Agents are the interface between end users, who wish to conduct some data miming activity, and the MASminer environment. There is typically one User Agent per MASminer end user. Task Agents are agents that are spawned by User Agents in response to an end user data mining request; they exist until the data mining task they are directed to coordinate, whatever this might be, is completed. Task Agents are responsible for coordinating the response to an end user data mining request. They do this by interacting with existing agents within the MASminer environment, they typically do not generate a solution themselves. MASminer supports a number of different types of Task Agent (work is in progress to increase the number of such agents). In the context of this paper a clustering task agent will be spawned. Data Mining Agents are typically equipped with data mining algorithms of varying kinds, clustering algorithms with respect to the scenario

under consideration. Data Agents "own" data, or more specifically act as communication conduits to and from data sources. Validation Agents are a special type of agent that performs validation operations on data mining results. In the case of the clustering scenario this will be to identify the most appropriate cluster configuration (in terms of the WGAD and BGD metrics described above) from a collection of such configurations.

To the above list of agents we can also add some house keeping agents. MASminer has been implemented using the Java Agent Development Environment (JADE) which comes with a number of house keeping agents, namely: the AMS (Agent Management System) Agent and the DF (Directory Facilitator) Agent. The first is used to control and manage the lifecycle of other agents in the platform, the second provides a lookup service to allow agents to register their services. This lookup service allows a Task Agent to identify the appropriate Data Mining, Data and Validation Agents required to complete a given data mining tasks.

A typical MASminer agent configuration is given in Figure 2. The figure includes a User Agent, a Task Agent, several Data Mining Agents, a Data Agent and some house keeping agents. The directed arcs indicate communication between agents. Note that communication can be bidirectional or unidirectional. The MASminer agent configuration given in Figure 2 actually describes the clustering scenario of interest in this paper. The figure will be returned to in Section 5 where the MASminer clustering procedure is described in detail.

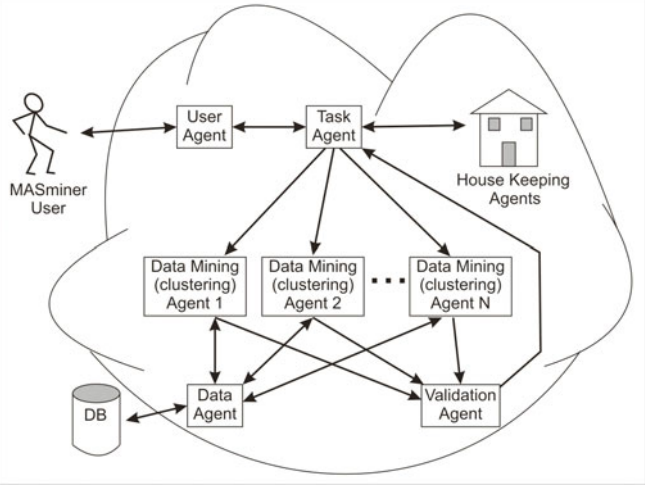


Fig. 2. An example MASminer Agent Configuration

Within MASminer communication is defined by a "be-spoke" data mining ontology, OntoDM. Intra agent messaging is conducted using a sequence of communicatives. A sample set of MASminer communicatives, directed at clustering operations, is presented in BNF form in Table 1. In the table the ... notation is used to indicate that there are further alternatives not included in the table for ease of understanding. From

Table 1 it can be seen that, at a high level, an OntoDM *utterances* comprises one of the following:

1. A **data informative**, indicating a data set, coupled with a **data mining request** to perform a specific data mining task with respect to that data; typically sent by a Task Agent to a Data Mining Agent.
2. A **validation request**, comprising information regarding the nature of the validation and the data on which the validation is to be performed; typically sent by a Data Mining Agent to a Validation Agent.
3. A **data request** typically sent by a Data Mining Agent to a Data Agent.
4. A **data informative**, the response from a Data Agent to a successful data request from a Dara Mining Agent.
5. A **results informative** used to return results to the originating Task Agent.
6. Some form of **error informative** used to handle situations where agents are unable to respond to an utterance because of some technical miss-function.

5 Multi-Agent Clustering

MASminer is designed to act as a platform to support generic data mining. The data mining is implemented and coordinated according to the nature of the communications between agents. The process commences with the end user instructing their User Agent to perform a specific data mining task. This instruction is facilitated by a GUI included in the User Agent. The User Agent then spawns a specific Task Agent as directed by the end user's instruction. MASminer is facilitated with a number of kinds of generic Task Agent. The generated Task Agent then interacts with the house keeping agents to identify those MASminer agents that may best contribute to the resolution of the given data mining task. This generic process is illustrated in Figure 2 where the directed arcs indicate communications between agents.

With respect to data clustering, once the Task Agent has identified appropriate clustering Data Mining Agents (there are a sequence of N of these in Figure 2) the Task Agent requests the identified clustering agents to conduct the desired clustering with respect to the indicated data. Each clustering agent then communicates with the appropriate Data Agent (The Data Agent that has access to the indicated data source), as a consequence the Data Agent passes the data back to the clustering agent which then generates a set of clusters according to their specific clustering algorithm. The generated results are then passed to the Validating Agent, which determines the most appropriate set of clusters, according to some metric (experiments using WGAD and BGD are reported in this paper), and returns the "best" configuration to the Task Agent. The Task Agent then returns the result to the user, via the User Agent, after which the process is ended and the Task agent ceases to exist.

This process is illustrated in Table 2 in terms of the agent communicatives presented in Table 1 used at each stage. Note that the Table does not include communication with the house keeping agents or communication from the User Agent to the Task Agent as neither is conducted using OntoDM. Communication with house keeping agents is as

Table 1. Segment of message content for Clustering

* Top level utterances *\	
$< UTTERANCE >$	::= $< DATABASE >< TASKTYPE > \mid < VALIDATION > \mid$ $\mid < DATA\_REQ > \mid < DATA > \mid < RESULTS > \mid < ERROR >$;
* Data Informative *\	
$< DATABASE >$	::= "Database" URL| "Database" URL $< DATA\_PARAMS >$;
$< DATA\_PARAMS >$	::= "All" $\mid < FROM\_REC\_NO >< TO\_REC\_NO > \mid$ $< FROM\_ATT\_NO >< TO\_ATT\_NO > \mid$ $< FROM\_REC\_NO >< TO\_REC\_NO > \mid$ $< FROM\_ATT\_NO >< TO\_ATT\_NO >$;
* Data Mining Request *\	
$< TASKTYPE >$	::= $< CLUSTERING > \mid \ldots$;
$< CLUSTERING >$	::= $< NAME\_OF\_ALGO > \mid$ $< NAME\_OF\_ALGO >< CLUSTERING\_PARAM >$;
$< NAME\_OF\_ALGO >$	::= "K-means"|"KNN"|"DBSCAN" ;
$< CLUSTERING\_PARAM >$	::= int | double | double int ;
* Validation Request *\	
$< VALIDATION >$	::= $< UNSUPERVISED\_VALIDATION > \mid \ldots$;
$< UNSUPERVISED\_$ $VALIDATION >$	::= $< APPROACH\_NAME > \mid$ $< APPROACH\_NAME >< VALIDATION\_PARAM >$;
$< APPROACH\_NAME >$	::= "WGAD" | "BGD" | ... ;
$< VALIDATION\_PARAM >$	::= $< RESULTS > \mid \ldots$;
* Data Request or Informative *\	
$< DATA\_REQ >$	::= $< DATA\_PARAMS >$;
$< DATA >$	::= arrray_of_data ;
* Results Informative *\	
$< RESULTS >$	::= $< CLUSTER\_LIST\_OF\_LISTS >< ACCURACY > \mid \ldots$;
$< CLUSTER\_LIST\_OF\_LISTS >$	::= "["$< CLUSTERLIST >$"]"|"[ ]" ;
$< CLUSTERLIST >$	::= "["$< CLUSTER >$"]"| ; "["$< CLUSTER >$"]"$< CLUSTERLIST >$;
$< CLUSTER >$	::= $< RECNO > \mid < RECNO >< CLUSTER >$;
$< RECNO >$	::= int ;
* Error Informative *\	
$< ERROR >$	::= "null" ;
* Miscelaneous *\	
$< FROM\_REC\_NO >$	::= int ;
$< TO\_REC\_NO >$	::= int ;
$< FROM\_ATT\_NO >$	::= int ;
$< TO\_ATT\_NO >$	::= int ;
$< ACCURACY >$	::= double ;

dictated by JADE. Communication from the User Agent to the Task Agent is integral to the Task Agent spawning process.

The entire scenario has been implemented using the current JADE implementation of MASminer. Three clustering agents were included, each with a distinct clustering algorithm: (i) K-means, (ii) KNN and (iii) DBSCAN.

Table 2. Clustering Process in Terms of the Agent Communications using OntoDM

Step	From Afent	To Agent	Message Content
1	Task Agent	Clustering Agent	"Database" URL "All" $< NAME\_OF\_ALGO >$ $< CLUSTERING\_PARAM >$
2	Clustering Agent	Data Agent	"All"
3	Data Agent	Clustering Agent	array_of_data
4	Clustering Agent	Validation Agent	$< APPROACH\_NAME >$ $< CLUSTER\_LIST\_OF\_LISTS >$
5	Validation Agent	Task Agent	$< CLUSTER\_LIST\_OF\_LISTS >$ $< ACCURACY >$
6	Task Agent	User Agent	$< CLUSTER\_LIST\_OF\_LISTS >$ $< ACCURACY >$

6 Evaluation

Evaluation of the multi-agent based clustering process, and comparison of the WGAD and BGD measures, was conducted using a sequence of pre-labelled ("classification") data sets from the UCI data repository [11]. The datasets used are all pre-labelled with class values, thus the results produced using MASminer can be compared with the known cluster configuration. Note that MASminer makes no use of these class labels, they are only used here to evaluate the outcomes.

With respect to the K-means algorithm, so that the number of desired clusters may be specified, we have used the number of classes given in the UCI repository (KNN and DBSCAN determine their own most appropriate number of clusters). The parameter used for KNN was a threshold, *t*, used to determine the nearest neighbour. The parameters used for DBSCAN were a minimum size *(minPts)* and density threshold *(ϵ)*. The specific values used was dependent on the nature of the data set. For each data set a pair of experiments was conducted using the WGAD and BGD best fit measures respectively.

The results, using Kmeans, KNN and DBSCAN, are reported in Tables 3 to 5 respectively. Each row in each table includes the number of records in the data set and the number of classes generated. For every case the WGAD and BGD values are reported. Note that in Table 4 a separation (BGD) of 0.00 is recorded for the $Lenses$ data set because KNN allocates all records to a single cluster! Across the data sets the the best (minimum) cohesion (WGAD) value is recorded as 1.17 using KNN on the $Lenses$ data set. The best (maximum) separation (BGD) value is recorded as 96.64 using KNN applied to the $PimaIndiansDiabetes$ data set.

Table 6 gives a comparison of the best clustering configurations when selection is made by; (i) minimising the WGAD, or (ii) maximising the BGD. Columns 4, 5 and 6 give the number of classes (column 4) produced with respect to the minimum WGAD value (column 5) which was obtained with respect to the given clustering algorithm (column 6). Similarly , columns 7, 8 and 9 give the number of classes (column 7) produced with respect to the minimum WGAD value (column 8) which was obtained with respect to the given clustering algorithm (column 9). It is interesting to note, from

Table 3. Muti-Agent Based Clustering Results using K-means algorithm

K-means					
Num.	Data Set	Num Records	Num Classes	WGAD	BGD
1	Lenses	24	3	2.41	1.41
2	Iris Plants	150	3	1.95	3.62
3	Zoo	101	7	7.32	5.99
4	Wine	178	3	21.14	24.81
5	Heart	270	2	10.71	10.78
6	Ecoli	336	8	1.67	1.07
7	Blood Tranfusion	748	2	30.62	31.96
8	Pima Indians Diabetes	768	2	44.06	21.18
9	Yeast	1484	10	1.67	0.75
10	Car	1782	4	8.52	3.31

Table 4. Muti-Agent Based Clustering Results using KNN algorithm

KNN					
Num.	Data Set	Num Records	Num Classes	WGAD	BGD
1	Lenses	24	1	1.17	0.00
2	Iris Plants	150	4	2.34	4.26
3	Zoo	101	9	6.76	9.88
4	Wine	178	3	22.97	41.80
5	Heart	270	3	13.84	19.09
6	Ecoli	336	13	1.47	1.78
7	Blood Tranfusion	748	2	31.08	35.03
8	Pima Indians Diabetes	768	3	38.93	96.64
9	Yeast	1484	9	1.46	1.62
10	Car	1782	5	10.40	3.76

Table 5. Muti-Agent Based Clustering Results using DBSCAN algorithm

DBSCAN					
Num.	Data Set	Num Records	Num Classes	WGAD	BGD
1	Lenses	24	2	1.65	0.94
2	Iris Plants	150	3	3.22	2.16
3	Zoo	101	7	6.69	6.80
4	Wine	178	4	10.63	16.75
5	Heart	270	4	4.56	15.14
6	Ecoli	336	10	2.41	1.16
7	Blood Tranfusion	748	5	33.93	45.05
8	Pima Indians Diabetes	768	9	53.26	40.19
9	Yeast	1484	9	2.11	0.57
10	Car	1782	5	6.69	3.48

Table 6. A comparison of WGAD and BGD measures

Num.	Data Set	Num Records	Num Classes	WGAD	Best clustering algo.	Num Classes	BGD	Best clustering algo.
1	Lenses	24	1	1.17	KNN	3	1.41	K-means
2	Iris Plants	150	3	1.95	K-means	4	4.26	KNN
3	Zoo	101	7	6.69	DBSCAN	9	9.88	KNN
4	Wine	178	4	10.63	DBSCAN	3	41.80	KNN
5	Heart	270	4	4.56	DBSCAN	3	19.09	KNN
6	Ecoli	336	13	1.47	KNN	13	1.78	KNN
7	Blood Tranfusion	748	2	30.62	K-means	5	45.05	DBSCAN
8	Pima Indians Diabetes	768	3	38.93	KNN	3	96.64	KNN
9	Yeast	1484	9	1.46	KNN	9	1.62	KNN
10	Car	1782	5	6.69	DBSCAN	5	3.76	KNN

the Table, that there is little agreement (except in the case of the $Yeast$ and the $Pima$ data sets) between the two metrics or the most appropriate algorithm.

Table 7 gives the expected accuracy, using the three different algorithms and the same parameters as above, given the assumed "ground truth" configuration indicated by the class labels associated with the records. The accuracy is calculated as follows:

$$Accuracy = \frac{\sum_{i=1}^{i=K} C_i}{m} \tag{3}$$

Where K is the number of clusters, m is the number of records and C_i is the size (number of records) of the majority class for cluster i. Not that the accuracy produced is not dependent on the number of clusters. Thus if a given input data set is known to have three classes (x, y and z) and MASminer produces four clusters, the first two of which contain only examples of class x, the third contains only examples of class y and the fourth contains only examples of class z; then an accuracy value of 100% will be

Table 7. Comparison of the result accuracy provided by K-means, KNN, and DBSCAN algorithms

Num.	Data Set	Num Records	Num Classes	K-means Accuracy	KNN Accuracy	DBSCAN Accuracy	
1	Lenses	24	3	0.62	0.62	0.62	K-means,KNN,DBSCAN
2	Iris	150	3	0.89	0.84	0.67	K-means
3	Zoo	101	7	0.78	0.83	0.85	DBSCAN
4	Wine	178	3	0.54	0.41	0.16	K-means
5	Heart	270	2	0.62	0.67	0.09	KNN
6	Ecoli	336	8	0.83	0.71	0.87	DBSCAN
7	Blood	748	2	0.76	0.76	0.25	K-means, KNN
8	Pima	768	2	0.65	0.65	0.12	K-means,KNN
9	Yeast	1484	10	0.53	0.34	0.64	DBSCAN
10	Car	1782	5	0.70	0.70	0.70	K-means,KNN,DBSCAN

returned. Of course the "ground truth" accuracy presented in Table 7 may not represent the "best" cluster configuration, however the table suffices as a guide to what may be the most appropriate configuration.

By cross referencing between Table 6 and Table 7 it can be seen that with respect to the $Lenses$, $Pima$ and Car data sets both metrics identify the most appropriate clustering configuration (although not necessarily using the same algorithm). In three cases ($Wine$, $Ecoli$ and $Yeast$) neither approach finds the most appropriate cluster configuration. From the remaining four data sets WGAD operates most successfully in three of the four cases. An argument can therefore be made in favour of WGAD.

7 Conclusions

This paper has described a multi-agent based approach to clustering. The motivation was two fold. The first was to provide a solution to the "best cluster configuration" problem, a challenging problem that the MASminer solution proposed in this paper goes someway to address. The second was to illustrate the operation of MASminer, a generic MADM that is currently under development, although a demonstration system is in operation (as illustrated in the foregoing). To generate a best cluster configuration appropriate metrics are required. In this paper the authors have also reported on the use of two suggested measures, a cohesion measure (WGAD) and a separation measure (BGD). The experimental results indicate that an argument can be made in favour of WGAD.

Other than extending the functionality of MASminer, and the associated OntoDM ontology, the authors are currently investigating the use of alternative "best fit" clustering measures and combinations of such measures. The authors are also conducting further experiments using a wider range of data sets. However, the authors are greatly encouraged by the research results obtained to date and reported in this paper.

References

1. Albashiri, K., Coenen, F., Leng, P.: Emads: An extendible multi-agent data miner. Journal of Knowledge Based Systems 22(7), 523–528 (2009)
2. Baazaoui Zghal, H., Faiz, S., Ben Ghezala, H.: A framework for data mining based multi-agent: An application to spatial data. In: Proceedings - WEC'05: 3rd World Enformatika Conference, vol. 5, pp. 22–26 (2005)
3. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent based architecture. In: Proceedings the Third International Conference on the Knowledge Discovery and Data Mining, pp. 211–214. AAAI Press, Menlo Park (1997)
4. Bailey, S., Grossman, R., Sivakumar, H., Turinsky, A.: Papyrus: A system for data mining over local and wide area clusters and super-clusters. In: Proceedings of Supercomputing. IEEE, Los Alamitos (1999)
5. Klusch, M., Lodi, S., Moro, G.: Agent-based distributed data mining: The kdec scheme. In: Klusch, M., Bergamaschi, S., Edwards, P., Petta, P. (eds.) Intelligent Information Agents. LNCS (LNAI), vol. 2586, pp. 104–122. Springer, Heidelberg (2003)

6. Reed, J.W., Potok, T.E., Patton, R.M.: A multi-agent system for distributed cluster analysis. In: Proceedings of Third International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'04) W16L Workshop - 26th International Conference on Software Engineering, Edinburgh, Scotland, UK, pp. 152–155. IEE, New York (2004)
7. Cen, Q., Zhao, J., Zhu, X.: The data mining system based on multi-agent under the circumstance of e-commerce. In: Proceedings - Third International Conference on Natural Computation, ICNC 2007, vol. 3, pp. 34–38 (2007)
8. Czarnowski, I., Jędrzejowicz, P.: Agent-based non-distributed and distributed clustering. In: Perner, P. (ed.) Machine Learning and Data Mining in Pattern Recognition. LNCS (LNAI), vol. 5632, pp. 347–360. Springer, Heidelberg (2009)
9. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2005)
10. Rao, M.R.: Cluster analysis and mathematical programming. Journal of the American Statistical Association 66(335), 622–626 (1971)
11. Asuncion, A., Newman, D.: UCI machine learning repository (2007)

Time-Based Reward Shaping in Real-Time Strategy Games

Martin Midtgaard, Lars Vinther, Jeppe R. Christiansen,
Allan M. Christensen, and Yifeng Zeng

Aalborg University, Denmark
{initram,larskv,jepperc,allanmc,yfzeng}@cs.aau.dk

Abstract. Real-Time Strategy (RTS) is a challenging domain for AI, since it involves not only a large state space, but also dynamic actions that agents execute concurrently. This problem cannot be optimally solved through general Q-learning techniques, so we propose a solution using a Semi Markov Decision Process (SMDP). We present a time-based reward shaping technique, TRS, to speed up the learning process in reinforcement learning. Especially, we show that our technique preserves the solution optimality for some SMDP problems. We evaluate the performance of our method in the Spring game Balanced Annihilation, and provide some benchmarks showing the performance of our approach.

1 Introduction

Reinforcement learning (RL) is an interesting concept in game AI development since it allows an agent to learn by trial-and-error by receiving feedback from the environment [2]. Learning can be done without a complete model of the environment which means that RL can be easily applied to even complex domains. RL has been applied to many types of problems, including many different board games [3], a simple soccer game [4] and robot navigation [5]. In the context of RTS games, RL has been applied to a small resource gathering task [7], and a commercial real-time strategy game (RTS) called MadRTS [6].

The benefit of applying RL is to create adaptive agents (Non-player characters in computer games) that may change their behaviour according to the way opponents play throughout games. Generally, RL requires that the problem shall be formulated as a Markov decision process. This demands that environmental states, as well as actions, must be well defined in the studied domain. However, a complex and dynamic RTS game always involves a very large state-action space. Moreover, agents' actions may last several time steps and exhibit a dynamic influence on the states. Both of these problems prevent a fast convergence to an optimal policy in RTS games. Recently, Kresten *et al.* [10] showed that the hierarchical decomposition of game states may mitigate the dimensional problem. This paper will investigate solutions to the second problem and speed up the convergence using relevant techniques.

We resort to Semi Markov Decision Processes (SMDPs) that extend MDPs for a more general problem formulation. SMDPs allow one single action to span multiple time steps and change environmental states during the action period. This property

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 115–125, 2010.

matches well with solutions to the dynamic actions within RTS games of our interest. However, the problem of slow convergence occurs when an optimal policy needs to be compiled in a short period of gameplay.

To speed up the learning process, we present a time-based reward shaping technique, TRS, which makes it possible to apply the standard Q-learning to some SMDPs and to solve SMDPs in a fast way. We let the agent get a reward only after it completes an action, and the reward depends on how much time it takes to complete the action. This means that after many trials, the agent will converge towards the fastest way of completing the scenario, since all other solutions than the fastest possible one will result in a larger negative reward. Our proposed approach puts some constraints on the properties of the SMDP problems on which time-based reward shaping can be applied without compromising the solution optimality.

We compare TRS to the Q-learning algorithm SMDPQ presented by Sutton *et al* [1]. SMDPQ extends Q-learning to support SMDPs by changing the update rule. We theoretically analyse the equivalence of optimal policies computed by both SMDPQ and our proposed method, TRS, given the constraints. We evaluate performance of TRS on a scenario in the RTS game of Balanced Annihilation[1] and show a significant improvement on the convergence of SMDPQ solutions. We also apply TRS to other RL methods such as Q-learning with eligibility traces, Q(λ), and show that this also can be successfully extended to support some SMDP problems.

The rest of this paper is organized as follows. Section 2 describes related work on using RL in computer games. Section 3 discusses some background knowledge. Subsequently, Section 4 presents our proposed method, TRS, and analyse the policy equivalence. Section 5 provides experimental results to evaluate the method. Finally, Section 6 concludes our discussion and provides interesting remarks for future work.

2 Related Work

RL methods have been well studied in the machine learning area where much of the work focuses on the theoretical aspect of either the performance improvement or the extension from a single-agent case to a multi-agent case. Good survey papers can be found in [11]. Although RL techniques have been demonstrated successfully in a classical board game [12], computer games are just recently starting to follow this path [13].

RTS games are very complex games, often with very large state- and action-spaces and thus when applying RL to these problems we need ways to speed up learning in order to make the agent converge to an optimal solution in reasonable time. For example, some methods like backtracking, eligibility traces [2], or reward shaping [9] have been proposed for this purpose. Laud demonstrates[9] that reward shaping allows much faster convergence in RL because the reward horizon is greatly decreased when using reward shaping, i.e. the time that passes before the agent gets some useful feedback from the environment is decreased. Meanwhile, Ng *et al.* [8] prove if the reward shaping function takes the form of the difference of potentials between two states the policy optimality is preserved.

[1] http://springrts.com/wiki/Balanced_Annihilation

Recently, Marthi *et al.* made a hierarchical concurrent approach to a small gathering task in an RTS game [7]. This is a very basic scenario only constituting a small part of an entire RTS game, but it is a clear proof that a hierarchical division of an RTS game may very well be possible. This means that in some cases we are able to identify isolated subtasks which we want to solve as fast as possible, so we will be able to apply time-based reward shaping to achieve this goal.

The Q-learning variant SMDPQ[1] is used for proving policy equivalence with TRS on the supported SMDP problems.

3 Background

In this section we will describe some of the prerequisites needed for discussing reward shaping in problems with non-discrete time. We will cover Semi Markov Decision Processes (SMDPs) and Q-learning (for MDPs). This forms the basis for our proposal on time-based reward shaping.

3.1 SMDP

An SMDP is a simple generalisation over an MDP where an action does not necessarily take a single time step, but can span several time steps. The state of the environment can change multiple times from when an action has been initiated until the next decision is initiated. Rewards are also slightly different from those of standard MDP, as the decision maker will both receive a lump sum reward for taking an action and also continuous rewards from the different states entered during the decision epoch. This all means that SMDPs can describe more general scenarios and environments than MDPs [1].

Formally an SMDP can be described as the 5-tuple (S, A, T, F, R), where:

- S : the finite state set
- A : the finite set of actions
- T : the transition function defined by the probability distribution $T(s'|s, a)$
- F : the probability of transition time for each state-action pair defined by $F(s', \tau|s, a)$
- R : the reward function defined by $R(s'|s, a)$

The F function specifies the probability that action a will terminate in s' after τ timesteps when starting from s.

3.2 Q-Learning

Q-learning is an algorithm for finding an optimal policy for a MDP (not SMDP). It does so by learning an action-value function (or Q-function) that returns the expected future reward of taking an action in a specific state. Q-learning does not need a model of its environment to be able to learn this function. The Q-function is a mapping from a state and an action to a real number, which is the expected future reward: $Q : S \times A \mapsto \mathbb{R}$.

After the Q-function has been learned, it is trivial to find the optimal policy, as one simply has to choose the action with the highest return from the Q-function in a given state.

Learning the Q-function is done using an action-value, and with an update rule shown in Eq. 1. In addition, the algorithm chooses its actions in an ϵ-greedy manner. It has been shown that by using this update rule the algorithm converges to Q^*, the optimal value function, and using this to choose actions will result in the optimal policy, π^* [2].

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{1}$$

Q-learning is called an off-policy algorithm as it does not use its policy to update the Q-function, but instead uses the best action given its current Q-function. This however is not necessarily its policy, as the algorithm will have to perform exploration steps from time to time. These choices are however needed for the algorithm to converge to an optimal solution as it will have to take all actions in all states to be sure that it converges to the optimal policy.

Sutton *et al* [1] proposed an algorithm, SMDPQ, which extends Q-learning to support SMDPs by changing the update rule. Normally the discount factor, γ, is multiplied with the expected future reward, as all actions in MDP problems are assumed to take constant time to complete. In the modified version, the discount factor is raised to the power of the number of time steps that the action took to complete. This results in an update in Eq. 2.

$$[Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(R_t + \gamma^k \max_a Q(s_{t+1}, a) - Q(s_t, a_t))] \tag{2}$$

3.3 Reward Shaping

Ng *et al.* [8] proposed reward shaping to speed up Q-learning while preserving the policy optimality. Every optimal policy in an MDP, which does not use reward shaping will also be an optimal policy in an MDP which uses reward shaping, and vice versa. To achieve this, we need a function $\Phi(s)$ that can calculate a value based on state s, which should represent the potential of the state, making it comparable to another state. The potential-based shaping function is formally defined as:

$$F(s, a, s') = \gamma\Phi(s') - \Phi(s) \tag{3}$$

An example of such a function could be the euclidean distance to a goal from a current position in a game world, as negative. As a rule of thumb, $\Phi(s) = V_M^*(s)$, where $V_M^*(s)$ means the value of the optimal policy for MDP M starting in state s, might be a good shaping potential.

4 Time-Based Reward Shaping

We perceive that reward shaping can be used to extend the standard Q-learning algorithm to support SMDPs. In this section, we firstly propose a time-based reward shaping method, *TRS*, and discuss the solution optimality in connection with SMDP. Then, we elaborate the validity of our proposed method through two counter examples.

4.1 Our Solution

We propose the simple solution of using the time spent for an action as an additional negative reward given to the agent after it completes that action. Formally, we define the time-based shaping function in Eq. 4.

$$F(s, a, s') = -\tau(a) \tag{4}$$

where $\tau(a)$ is the number of time steps it takes the agent to complete action a

As the time spent for an action is independent of transition states, the reward shaping is not potential based. Consequently, we can not guarantee the solution optimality for any SMDP problem. However, we observe that solution optimality may be preserved for some SMDP problems if the problem satisfies some properties.

We begin by showing that using reward shaping (in Eq. 4), the agent will never learn a solution consisting of a cyclic sequence of states for which the individual rewards total to a positive reward, as this allows for an infinite loop. Then, we proceed to prove a guaranteed consistency between the optimal policy when learning by our approach, time-based reward shaping, and when using an approach like SMDPQ, i.e. that our approach converges to the optimal policy as SMDPQ.

No Cyclic Policy. As shown in Eq. 4, only negative reward is given in the learning process so that it is clear that no cyclic sequence of states can result in a positive reward. This ensures that the possible cyclic issue of a poorly chosen reward function can not occur in our solution.

Policy Equivalence. We find that the termination reward is actually insignificant using our TRS method, since it will converge to the fastest way to termination even when reward is e.g. zero for all states in the SMDP problem. However, the termination reward is required in order to compare it to SMDPQ. In SMDPQ the algorithm selects the fastest path to the goal, by discounting the reward over time, while our approach gives a negative reward for each time step used until termination. Both of these approaches make sure that a faster path has a higher reward than all of other paths. This however is only true if the reward received at the terminal states is the same for all terminal states. If they were to differ, the two approaches are not guaranteed to find the same policy, as the negative reward earned by the time spent, together with the discounting, may have conflicting "priorities".

The Q functions for *TRS* and SMDPQ , denoted Q_{TRS} and Q_{SMDPQ}, can be seen respectively in Eqs. 5 and 6. The shown values are only for special cases where α is 1 for both approaches and γ is 1 for Q_{TRS} and between 0 and 1 for Q_{SMDPQ}. τ is here the time to termination by taking action a in state s and following the policy after this. The equation shows that any action that leads to a faster termination, will have a higher Q-value, and as this is the case for both algorithms they end up with the same policy.

$$Q_{TRS}(s, a) = r - \tau \tag{5}$$

$$Q_{SMDPQ}(s, a) = \gamma^{\tau} * r \tag{6}$$

Formally, we assume that a specific group of SMDPs shall have the following properties:

1. Reward is only given at the end, by termination,
2. The goal must be to terminate with the lowest time consumption,
3. The reward must be the same for all terminal states.

Then, time-based reward shaping preserves the solution optimality of SMDPQ as indicated in proposition 1. However, the policies can be equal even though the restrictions do not hold, but this is not guaranteed.

Proposition 1 (Policy Equivalence)

$$\forall s : \arg\max_a Q^*_{TRS}(s,a) = \arg\max_a Q^*_{SMDPQ}(s,a)$$

We note that our approach is not simply a matter of guiding the learning, but actually giving it essential information to ever finding the optimal policy. This however is only the case when using a MDP algorithm to solve the problem. When the algorithm converges the agent will choose a policy allowing for the fastest solution of the given problem—measured in time and not number of actions. The application of the reward penalty encourages the agent to achieve a goal as fast as possible, thus making our approach independent of game-specific properties such as specific units etc.

4.2 Counter Examples

Here we present two examples of why the previousy mentioned properties on applicable SMDPs have to be obeyed in order for TRS to converge to the optimal policy, i.e. the same as the SMDPQ algorithm. We create examples for properties 1 and 3, and show that if the propeties are not obeyed, the two algorithms are not guaranteed to converge to the same optimal policy.

Rewards in Non-Terminal States. Figure 1 on page 121 shows the state-space of an example and illustrates why it is important that any reward should only be given in terminal states in order for Q_{TRS} to converge to the same optimal policy as Q_{SMDPQ}. For this case we use $\gamma = 0.99$, define the actions of the environment as A_1, A_2 and A_3, and the time steps required to take transitions as $\tau(A_1) = 1$, $\tau(A_2) = 2$, $\tau(A_3) = 50$. This results in the optimal policy using Q_{TRS} would go directly from S to T, but the optimal policy for Q_{SMDPQ} would be from S to T through S'.

The following shows the calculations resulting in the optimal policy using Q_{TRS}:

$$Q_{TRS}(S, A_1) = (100 - \tau(A_1))\gamma = \mathbf{98.0}$$

$$Q_{TRS}(S, A_2) = (50 - \tau(A_2))\gamma + (100 - \tau(A_3))\gamma^2 = 96.5$$

While the optimal policy using Q_{SMDPQ} is caculated as follows:

$$Q_{SMDPQ}(S, A_1) = \gamma^{\tau(A_1)}100 = 99.0$$

$$Q_{SMDPQ}(S, A_2) = \gamma^{\tau(A_2)}50 + \gamma^{\tau(A_2)+\tau(A_3)}100 = \mathbf{108.3}$$

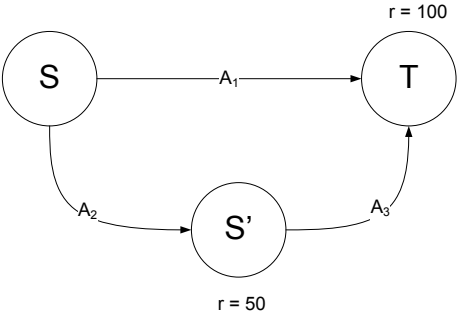


Fig. 1. A case illustrating a counterexample of why it is important that rewards are only given in terminal states

The goal of a Q_{TRS} problem should be to reach termination in the fewest time steps, which means that giving rewards in non-terminal states, and thereby not obeying the restrictions, will result in a possible suboptimal solution.

Different Rewards in Terminal States. Figure 2 on page 121 shows its importance that all terminal states must yield the same reward in order to assure that Q_{TRS} converges to the same optimal policy as Q_{SMDPQ}. In this case the following parameter values are used: $\gamma = 0.99, \tau(A_1) = 60, \tau(A_2) = 1$. This results in the optimal policy using Q_{TRS} would go from S to T_2, but the optimal policy for Q_{SMDPQ} would be T_1 instead.

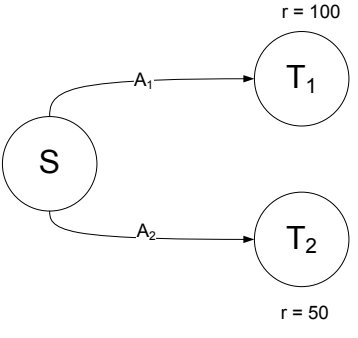


Fig. 2. A case illustrating a counterexample of why it is important that all the rewards given in terminal states need to be the same

The following shows the reward calculations for the optimal policy using Q_{TRS}:

$$Q_{TRS}(S, A_1) = (100 - \tau(A_1))\gamma = 39.6$$
$$Q_{TRS}(S, A_2) = (50 - \tau(A_2))\gamma = \mathbf{48.5}$$

While the reward calculations using Q_{SMDPQ} are as follows:

$$Q_{SMDPQ}(S, A_1) = \gamma^{\tau(A_1)}100 = \mathbf{54.7}$$

$$Q_{SMDPQ}(S, A_2) = \gamma^{\tau(A_2)}50 = 49.5$$

All terminal states in a Q_{TRS} problem must yield the same reward, and if this is not obeyed the optimal policy will, as exemplified above, not be guaranteed to be equivalent to the optimal policy of Q_{SMDPQ}.

5 Experiments

To test the proposed time-based reward shaping, we set a simple scenario in the RTS game Balanced Annihilation which can be described as an SMDP problem. The optimal solution for this problem is learned using Q-learning and Q(λ)[2] with time-based reward shaping, and the proven SMDP approach SMDPQ[1].

5.1 Game Scenario

The scenario is a very simple base-building scenario. The agent starts with a single construction unit, a finite amount of resources and a low resource income. The agent controls the actions of the construction unit, which is limited to the following three actions:

- Build a *k-bot lab*, for producing attack-units (production building)
- Build a *metal extractor*, for retrieving metal resources (resource building)
- Build a *solar collector*, for retrieving solar energy (resource building)

All actions in the scenario are sequential, as the construction unit can only build one building at a time. The goal of the scenario is to build four of the production buildings as quickly as possible (in terms of game time). The build time depends on whether we have enough available resources for constructing the building; e.g. if we have low resource income and our resource storage is empty, it takes much more time to complete a new building than if we have high resource income. Therefore the optimal solution is *not* to construct the four production buildings at once, without constructing any resource building, as this would be very slow. Figure 3 on page 123 shows a screenshot of a possible state in this scenario in Balanced Annihilation.

As state variables, the number of each type of building is used; the number of production building has the range $[0; 4]$, and the numer of each of the two types of resource buildings has the range $[0; 19]$. This results in a state space of $5 \times 20 \times 20 = 2000$ and an state-action space of $2000 \times 3 = 6000$. This means that it is not possible for the agent to take the current amount of resources into account, as this may vary depending on the order in which the buildings have been constructed. We do not think that this will have a great impact on the final policy though.

Fig. 3. A screenshot of the scenario in Balanced Annihilation. This shows the state of the game after five *metal extractors*, one *solar collector*, and two *k-bot labs* have been build. The construction unit is currently in the process of building a third *k-bot lab*. At the top of the screen a white and a yellow indicator can be seen, representing the current level of metal end solar energy, respectively.

5.2 Results

Figure 4 on page 123 shows the results of four different settings of reinforcement learning in the scenario: Standard Q-learning and Q(λ) with time-based reward shaping, and SMDPQ both with and without time-based reward shaping.

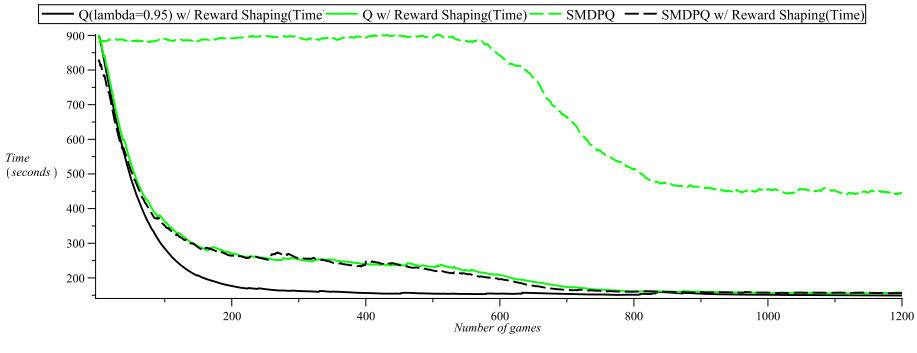


Fig. 4. A graphical representation of the convergence of the different approaches for SMDP support in Q-learning. Q-learning values are: $\alpha = 0.1$, $\epsilon = 0.1$, $\gamma = 0.9$. Exponential smoothing factor = 0.02. This experiment was done on the base-building scenario.

The standard MDP Q-learning algorithm extended with time-based reward shaping shows a significant improvement over standard SMDPQ. SMDPQ with reward shaping also provides much faster convergence than standard SMDPQ. However, there is no difference on applying reward shaping to standard MDP Q-learning and SMDPQ. This can be explained by the fact that SMDPQ and our reward shaping both help solve the problem as fast as possible, since the reward decreases as time increases. The algorithms are not additive, so applying one time penalty-algorithm to another does not increase the convergence rate. In addition, both algorithms only pass reward back one step, and thus the two approaches converge at the same rate.

When using the MDP algorithm on the scenario, which is in fact an SMDP problem, this kind of reward shaping is actually necessary in order to make the algorithm converge to a correct solution. Without reward shaping, the agent would know nothing about the time it took to complete an action, and so it would converge to the solution of building four production buildings in as few steps as possible, namely four. This solution is not the optimal one in terms of game time, since the production of resources would be very low given the fact that the agent does not build any resource buildings.

From Figure 4 on page 123 it is not clear whether or not SMDPQ without reward shaping actually ever converges. In this experiment SMDPQ converged to the optimal policy after approximately 22000 runs, but this result is not a part of the figure, since it would obfuscate the other information contained within the graph.

Adding eligibility traces, in this case Q(λ), to the Q-learning with TRS furthermore significantly improves the convergence rate, as it can be seen in Figure 4 on page 123.

5.3 Discussion

As we have shown in our case, it is important to use reward shaping to reduce the time spent on the learning. It is especially important when each learning session takes more than a few seconds. In this case the change from 22000 sessions to 700, can result in a problem being feasible to learn and not take several hours or even days to learn.

TRS can be a good choice when using RL for a sub-task in a game, where the agent must solve a problem as fast as possible and can not fail to achieve the goal. This however does not include an RTS game as a whole, as it is possible to fail to achieve the goal, by loosing the game. General reward shaping can be applied to all MDP problems, but it is especially important in games, where the state space can be very large. Another task which could be solved using TRS is pathfinding (in general).

6 Conclusions and Future Work

Scenarios in computer games are often time dependent and agents' actions may span multiple time steps. We propose an SMDP solution to have agents learn adaptive behaviours. We make a further step to speed up the learning process through reward shaping techniques. We propose a time-based reward shaping method, *TRS*, that punishes the delayed actions using times. In particular, we show that our method may result in the same policy as SMDP solutions for some specific problems. Our experiments on the Balanced Annihilation game show that applying time-based reward shaping is a

significant improvement for both general Q-learning, SMDPQ and Q(λ), allowing fast convergence when solving some SMDPs. In addition, the technique allows us to solve SMDP problems with the standard Q-learning algorithm.

We found a way to use reward shaping in Q-learning that reduced the number of runs needed to converge for a restricted group of SMDP problems. It would be interesting to see if this, or some similar approach, could be applied in a more general case. There is already a general concept of reward shaping, which has been proved to work. But we feel that some more specific concepts about including time usage in reward shaping could really be beneficial. Improvements of this approach could result in loosing the restrictions for the SMDP problems, optimally allowing support for any SMDP problem.

Applying TRS to more advanced problems could also be done, to show whether this approach is too limited. An example of such an advanced scenario could be one with cooperative agents. This is especially relevant in RTS games, where units need to cooperate in order to achieve their goals faster.

References

1. Sutton, R., Precup, D., Singh, S.: Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. Artificial Intelligence 112, 181–211 (1999)
2. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
3. Ghory, I.: Reinforcement learning in board games. Technical Report, Department of Computer Science, University of Bristol (2004)
4. Littman, M.L.: Markov games as a framework for multi-agent reinforcement learning. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 157–163 (1994)
5. Mataric, M.J.: Reward Functions for Accelerated Learning. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 181–189 (1994)
6. Sharma, M., Holmes, M., Santamaria, J., Irani, A., Isbell, C., Ram, A.: Transfer Learning in Real-Time Strategy Games Using Hybrid CBR/RL. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence, pp. 1041–1046 (2007)
7. Marthi, B., Russell, S., Latham, D., Guestrin, C.: Concurrent Hierarchical Reinforcement Learning. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, pp. 779–785 (2005)
8. Ng, A.Y., Harada, D., Russell, S.: Policy invariance under reward transformations: Theory and application to reward shaping. In: Proceedings of the Sixteenth International Conference on Machine Learning, pp. 278–287 (1999)
9. Laud, A.D.: Theory and application of reward shaping in reinforcement learning. University of Illinois at Urbana-Champaign (2004)
10. Andersen, K.T., Zeng, Y.F., Tran, D., Christensen, D.D.: Experiments with Online Reinforcement Learning in Real-Time Strategy Games. Applied Artificial Intelligence: An International Journal 23(9), 855–871 (2009)
11. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: a survey. Journal of Artificial Intelligence Research 4, 237–285 (1996)
12. Tesauro, G.: Td-gammon, a self-teaching backgammon program, achieves master-level play. Neural Comput. 6(2), 215–219 (1994)
13. Manslow: Using reinforcement learning to solve ai control problems. AI Game Programming Wisdom 2, 591–601 (2004)

Wise Search Engine Based on LSI

Yang Jianxiong and Junzo Watada

Graduate School of Information, Production and Systems, 2-7 Hibikino, Wakamatsu, Kitakyushu 808-0135 Japan
leoworldplus@yahoo.co.jp, junzow@osb.att.ne.jp

Abstract. The objective of this work is to provide, as a search engine, latent semantic indexing (LSI), which is a classical method to produce optimal approximations of a term-document matrix and has been used for textual information mining. The use of this technique is examining mine content which based web document, using keyword features of documents. Experimental results show that together with both textual and latent features LSI can extract the underlying semantic structure of web documents, thus improve the search engine performance significantly.

Keywords: Latent semantic indexing, search engine, data mining, semantic web.

1 Introduction

Search engines are still in their infancy. Although the extracting function of many algorithms is characteristic for finding appropriate pages [1], Hawking suggests that their algorithms were not very accurate [2]. General search engine always give user a lot of results without users commitment. So existing search engines and their users are typically at cross purposes. While these systems normally retrieve documents based on low-level capability, users usually have more abstract notion of what will satisfy them when conducting a query for certain information. For instance, most search engines use low-level syntactic properties to characterize the semantics of web documents. These properties usually adapt various sophisticated variations of simple keyword matching counts. There are many research prototypes that take link information into account [3] in order to judge users purposes using logs, but they still do not overcome the so-called semantic gap[4] which characterizes the difference between two descriptions of an object by different linguistic representations, for instance languages or symbols. In computer science, the concept is relevant whenever ordinary human activities, observations, and tasks are translated into a computational representation. Sometimes, the users have in mind so abstract a concept that they themselves do not know what they want exactly until they see it. At that point, they may want documents similar to what they have just seen or can envision. Again, however, the notion of similarity is typically based on high-level abstractions, such as activities or events described in the document, or some evoked emotions, among others. The standard definitions of similarity using low-level features generally will not produce results with high quality.

However, while search engines definitely work well for certain search tasks such as finding the home page of an organization, they may be less effective for satisfying broad

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 126–136, 2010.

or ambiguous queries. The results of queries about different subtopics or meanings will be mixed together, thus implying that the user may have to sift through a large number of irrelevant items to find our interesting point. On the other hand, there is no way to exactly figure out what is relevant to the user because the queries are usually very short and their interpretation is inherently ambiguous in the absence of a context.

In contrast, if we submit the query "web of science" to a general search engine, we see that each element is scattered in the ranked list of search results, often through a large number of result pages. Major search engines have this problem and interweave related documents to different topics (a technique known as implicit clustering) but given the limited capacity of a single result page, certain topics will be inevitably hidden from the user. On the same query, LSI can do a good job in grouping the items.

2 Related Works in Semantic Web Search

One of the first attempts to enhance Semantic Web search engines with ranking capabilities is reported in [21]. The authors define a similarity score measuring the distance between the systematic descriptions of both query and retrieved resources. They first explode an initial set of relations (properties) by adding hidden relations, which can be inferred from the query. Similarity is then computed as the ratio between relation instances linking concepts specified in the user query and actual multiplicities of relation instances in the semantic knowledge base. This method is applied on each property individually and requires exploring all the Semantic Web instances. Moreover, the user is requested to specify all the relations of interest. Thus, since it is predictable that the number of relations will largely exceed the number of concepts [15], its applicability in real contexts is severely compromised. A similar approach, aimed at measuring the relevance of a semantic association (that is, a path traversing several concepts linked by semantic relations) is illustrated in [23]. The authors provide an interesting definition of relevance as the reciprocal of the ambiguity of the association itself. However, this approach suffers from the same limitations of [21], since queries have to be specified by entering both concepts and relations, and ambiguity is measured over each relation instance.

Nevertheless, the idea of exploring the set of relations that are implicit in the user's mind (but which are not made explicit in defining the query) has been pursued in many works. In [20], ontology-based lexical relations like synonyms, antonyms, and homonyms between keywords (but not concepts) have been used to "expand" query results. In this case, search is targeted to the Web, rather than to the Semantic Web. In [24], a similar approach has been integrated into artificial intelligence methodologies to address the problem of query answering. In [27], query logs are used to construct a user profile to be later used to improve the accuracy of Web search. Semantic Web search from the point of view of the user's intent has been addressed also in [23] and [28], where the authors present two methodologies for capturing the user's information need by trying to formalize its mental model. They analyze keywords provided during query definition, automatically associate related concepts, and exploit the semantic knowledge base to automatically formulate formal queries.

A totally different solution is represented by OntoLook [27]. The basic idea is that if a graph-based representation of a Web page annotation can be provided, where concepts and relations (together with their multiplicities) are modeled as vertices and weighted edges, respectively, it becomes possible to define a series of cuts removing less relevant concepts from the graph. This allows for the generation of a so-called candidate relation-keyword set (CRKS) to be submitted to the annotated database, which can significantly reduce the presence of uninteresting pages in the result set. It is worth observing that the strategy behind OntoLook only allows us to empirically identify relations among concepts that should be less relevant with respect to the user query. This information is used to reformulate the user query by including only a subset of all the possible relations among concepts, which is later used to retrieve web pages from the annotated database. The user is not requested to specify relations of interest during query definition. However, the effectiveness of the approach is strongly limited by the fact that there does not exist any ranking strategy. Even if the authors claim that any of the existing page ranking algorithms can be used to order the obtained result set, it is worth remarking that this is not completely true. In fact, a ranking strategy like the PageRank [18], [19] used by Google [17] is only one of the ranking algorithms used to organize results to be displayed to the user. Many other statistical and text-matching techniques are used together with PageRank. Of course, PageRank can be used in conjunction with [27] to exploit relevance feedback and post process the result set. But the use of the remaining techniques is not feasible since they cannot be reasonably applied into a concept-relation-based framework where ontology is predominant on pure text. The authors themselves state that what is really needed is a relation-based page rank algorithm.

3 LSI

Latent Semantic Indexing (LSI) is an indexing and retrieval method that uses a mathematical technique called Singular Value Decomposition (SVD) to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of texts. LSI is based on the principle that words tend to have similar meanings in the same context. A key feature of LSI is its ability to extract the conceptual content of a body of texts by establishing associations among those terms that occur in similar contexts.

Latent Semantic Indexing has its ability to correlate semantically terms in a collection of texts. It was first applied to texts at Bell Laboratories in the late 1980s. The method, also called Latent Semantic Analysis (LSA), uncovers the underlying latent semantic structure in the usage of words in a body of texts and it can be used to extract the meaning of the text in response to user queries, commonly referred to as concept searches. Queries, or concept searches, against a set of documents that have undergone LSI will return results that are conceptually similar in meaning to the search criteria even if the results do not share a specific word or words with the search criteria.

3.1 Challenges to LSI

Early researches on LSI focused on whether it can be usable or not. It was in test. So early challenge to the LSI stayed at scalability and performance. LSI requires relatively high computational performance and memory in comparison to other information retrieval techniques.[7] However, with the implementation of modern high-speed processors and the availability of inexpensive memory enables us to interpret these issues in LSI. Real-world applications are common to fully process more than 30 million documents through the matrix and SVD computations in some LSI applications.

Another challenge to LSI has been the alleged difficulty in determining the optimal number of dimensions when performing the SVD. As a general rule, fewer dimensions allow for broader comparisons of the concepts contained in a collection of text, while a higher number of dimensions enable more specific (or more relevant) comparisons of concepts. The number of documents in the collection constrains the actual number of dimensions that can be used. Researches have demonstrated that around 300 dimensions can usually provide the best results with moderate-sized document collections (hundreds of thousands of documents) and perhaps 400 dimensions for larger document collections (millions of documents).[8] However, recent studies indicate that 50-1000 dimensions are suitable depending on the size and nature of the document collection.[9]

3.2 Mathematics of LSI

Here we give a calculation example which can build a 5×5 matrix for expressing this method in Subsection 2.3. LSI uses common linear algebra techniques to find the conceptual correlations in a collection of text. In general, the process involves constructing a weighted term-document matrix, performing a Singular Value Decomposition on the matrix, and using the matrix to identify the concepts contained in the text.

LSI begins by constructing a term-document matrix, **A**, to identify the occurrences of unique terms ***j*** within a collection of documents ***i***. In a term-document matrix, each term is represented by a row, and each document is represented by a column, with each matrix cell $\boldsymbol{l}_{ij}$ initially representing the number of times the associated term appears in the indicated document $\boldsymbol{tf}_{ij}$. This matrix is usually very large and very sparse.

Once a term-document matrix is constructed, local and global additional functions can be applied to it to condition the data. The additional functions transform each cell l_{ij} which describes the relative frequency of a term in a document. In these cells column t_j describes the relative frequency of **term *j*** in every document and row d_i describes the relative frequency of every term in ***document i*** (matrix (1)).

Some common functions are defined in the following table 1.

Table 1. The common local functions

Binary	l_{ij} = **1** if the term exists in the document, or else **0**
Term Frequency	l_{ij} = tf_{ij}, the number of occurrences of term(t) ***j*** in document(d) ***i***

$$\mathbf{A}=\begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & \cdots\cdots & t_j \end{matrix} \\ \left(\begin{matrix} l_{11} & l_{12} & l_{13} & \cdots & l_{1j} \\ l_{21} & l_{22} & l_{23} & & \vdots \\ l_{31} & l_{32} & l_{33} & & \vdots \\ \vdots & & & \ddots & \vdots \\ l_{i1} & \cdots & \cdots & \cdots & l_{ij} \end{matrix}\right) \end{array} \left|\begin{matrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_i \end{matrix}\right. \quad \cdots\cdots (1)$$

where $L_j = (l_{1j}, l_{2j}, l_{3j}, \ldots, l_{ij})$.

But we need another transformed matrix $\mathbf{T}_r$ (matrix (2)) and parameter to format it to each cell between 0 and 1 for judging value is big or small easily. So the term-

$$\mathbf{T_r}=\begin{pmatrix} \frac{1}{Max(L_1)} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{Max(L_2)} & 0 & \cdots & \vdots \\ 0 & 0 & \frac{1}{Max(L_3)} & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \frac{1}{Max(L_j)} \end{pmatrix} \quad \cdots\cdots (2)$$

where $L_j = (l_{1j}, l_{2j}, l_{3j}, \ldots, l_{ij})$.

But every term does not necessarily have the same relation rank and each term has different possibility for our purpose, so we need give them influence factor. The factor matrix is elicited as matrix (5). The last result computes the term and document document matrix $\mathbf{A'}= \frac{1}{j}\mathbf{A}\times\mathbf{T_r}$ is written in following:

$$\mathbf{A'}=\frac{1}{j}\begin{array}{c} \begin{matrix} t_1 & t_2 & t_3 & & t_j \end{matrix} \\ \left(\begin{matrix} \frac{1}{Max(L_1)}l_{11} & \frac{1}{Max(L_2)}l_{12} & \frac{1}{Max(L_3)}l_{13} & \cdots & \frac{1}{Max(L_j)}l_{1j} \\ \frac{1}{Max(L_1)}l_{21} & \frac{1}{Max(L_2)}l_{22} & \frac{1}{Max(L_3)}l_{23} & \cdots & \vdots \\ \frac{1}{Max(L_1)}l_{31} & \frac{1}{Max(L_2)}l_{32} & \frac{1}{Max(L_3)}l_{33} & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ \frac{1}{Max(L_1)}l_{i1} & \cdots & \cdots & \cdots & \frac{1}{Max(L_j)}l_{ij} \end{matrix}\right) \end{array} \left|\begin{matrix} d_1 \\ d_2 \\ d_3 \\ \\ d_i \end{matrix}\right. \quad \cdots\cdots (3)$$

vector spaces by transforming the single term-frequency matrix, R, into two other matrices — a term-concept vector matrix: A'(matrix(3)) and a singular factor values matrix, F(matrix (5)), which satisfy the following relations:

Term-frequency matrix R=A'×F · · · · · · (4).

$$F=\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_i \end{pmatrix} \begin{matrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_i \end{matrix} \quad \cdots\cdots (5)$$

The best search result has the biggest influence value—Max$\{r_i \mid r_i \in R^T, i=1, 2, 3, \ldots n\}$. So this document($d_i$) is the best one for us.

3.3 Example of Term-Frequency Matrix

In our example, terms 1, 2, 3, 4 and 5 are the our purpose:

Term 1 appears 3, 1, 0, 5, 0 times in document 1, 2, 3, 4, 5;
Term 2 appears 6, 3, 9, 1, 0 times in document 1, 2, 3, 4, 5;
Term 3 appears 2, 0, 2, 7, 0 times in document 1, 2, 3, 4, 5;
Term 4 appears 0, 0, 0, 0, 0 times in document 1, 2, 3, 4, 5;
Term 5 appears 6, 3, 5, 6, 0 times in document 1, 2, 3, 4, 5;
Factor matrix F is denoted as matrix(6):

$$F=\begin{matrix} 0.100 \\ 0.900 \\ 0.800 \\ 0.200 \\ 0.900 \end{matrix} \begin{matrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{matrix} \quad \cdots\cdots (6)$$

Term-document matrix **A** is shown in matrix 7:

And the transforming matrix $\mathbf{T}_r$ is shown in matrix 8:

Term-frequency matrix R is obtained as shown in matrix 10:

From the term-frequency matrix R^T=(0.358, 0.154, 0.376, 0.380, 0.000), we know documents 1, 3 and 4 are the fittest ones to our purpose; document 2 has some possibility; the term-frequency value of document 5 is 0, so it has nothing to do with our purpose.

$$\begin{array}{c} t_1\ t_2\ \ t_3\ \ t_4\ \ t_5 \\ \left(\begin{array}{ccccc} 3 & 6 & 2 & 0 & 6 \\ 1 & 3 & 0 & 0 & 3 \\ 0 & 9 & 2 & 0 & 5 \\ 5 & 1 & 7 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \end{array} \left|\begin{array}{c} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{array}\right. \quad \cdots\cdots (7)$$

$$\begin{pmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 6 \end{pmatrix} \quad \cdots\cdots (8)$$

Term-document matrix $\mathbf{A'} = \frac{1}{5}\mathbf{A} \times \mathbf{T_r}$ is shown in matrix 9:

$$\frac{1}{5} \times \begin{pmatrix} 3 & 6 & 2 & 0 & 6 \\ 1 & 3 & 0 & 0 & 3 \\ 0 & 9 & 2 & 0 & 5 \\ 5 & 1 & 7 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 6 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} 0.120 & 0.133 & 0.057 & 0.000 & 0.200 \\ 0.040 & 0.067 & 0.000 & 0.000 & 0.100 \\ 0.000 & 0.200 & 0.057 & 0.000 & 0.167 \\ 0.200 & 0.022 & 0.200 & 0.000 & 0.200 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \end{pmatrix} \quad \cdots\cdots (9)$$

$$\begin{pmatrix} 0.120 & 0.133 & 0.057 & 0.000 & 0.200 \\ 0.040 & 0.067 & 0.000 & 0.000 & 0.100 \\ 0.000 & 0.200 & 0.057 & 0.000 & 0.167 \\ 0.200 & 0.022 & 0.200 & 0.000 & 0.200 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \end{pmatrix} \times \begin{pmatrix} 0.100 \\ 0.900 \\ 0.800 \\ 0.200 \\ 0.900 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.358 \\ 0.154 \\ 0.376 \\ 0.380 \\ 0.000 \end{pmatrix} \left|\begin{array}{c} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{array}\right. \quad \cdots\cdots (10)$$

4 Efficiency and Implementation Issues

The performance of this method was evaluated using a database of accidents from the NITE (National Institute of Technology Evaluation). Our purpose is to find the reasons of seek similar accidents occurrence.

The main characteristics of the database used in our experiment are summarized in Table 2. The first column of this table gives the data set name, while the other columns indicate the number of cases.

Table 2. Database used in the experiment

Data Set	Cases
Electric appliance	200
Oil appliance	200
Gas appliance	200

4.1 Accident Causes Extracted by General Search Method

At first, we found key words relating to accident reasons such as "*fire started from around an old television*" in an accident explanation. Of course, by this character string, we can extract any accident reason which includes linear sequence of words. Table 3 lists the results. It has a very poor result and in fact we miss a lot of similar cases. So it can not be used in the practice.

Table 3. The results of general search method

Data Set	Cases	Matching cases
Electric appliance	200	2
Oil appliance	200	1
Gas appliance	200	1

4.2 Using LSI Method in Accident Search System

In fact, the accident description did not always use the same linear sequence of words. So we can not extract similar cases if they do not include linear sequence of words such as "*fire started from around an old television*". So we have to use LSI method to analyze them. First we split up linear sequence of words into separate elements and change them to standard form by parsing.[6] So it is parsed to be split into seven elements as [*fire*], [*start*], [*from*], [*around*], [*the*], [*old*] and [*television*]. In these seven elements, [from], [*around*] and [the] are preposition and article which have low relation rank. We gave them relation value 0.001, and then analyzed the remaining elements. The adjective and adverb have superior relation rank, so we give them relation values 0.500. But the noun and verb are the main ingredient of subject and predicate. So we gave them higher relation values 0.900 and the factor matrix F is obtained in following (matrix(11)):

$$F = \begin{pmatrix} 0.900 \\ 0.900 \\ 0.001 \\ 0.001 \\ 0.001 \\ 0.500 \\ 0.900 \end{pmatrix} \begin{vmatrix} fire \\ start \\ from \\ around \\ the \\ old \\ television \end{vmatrix} \cdots\cdots (11)$$

In the experiment database, we got the frequency in which seven elements appear in each case. So we calculated matching value by term-frequency matrix R (matrix(4)) and counted the cases included by each matching value interval as shown in Table 4. (V_M is matching value).

Table 4. The results of LSI method

Data Set	Matching case ($V_M \in [1,0.8)$)	Matching case ($V_M \in [0.8,0.6)$	Matching case ($V_M \in [0.6,0.3)$	Other ($V_M \in [0.3,0]$
Electric appliance	8	15	33	144
Oil appliance	3	5	9	183
Gas appliance	1	7	8	184

4.3 Searching Effect of LSI

Table 4 indicated the number of all correlative cases with each matching value. It is not surprising that rule pruning extract the closest cases besides probable cases. The matching value can make the information sort to gain expected information easily.

5 Application of LSI

LSI is used to perform automated document categorization. Several experiments have demonstrated t a number of successful correlations between the way LSI and humans' process and effectively categorized text [10]. Document categorization is the assignment of documents to one or more predefined categories based on their similarity to the conceptual content of the categories.[8] LSI uses template documents to establish the conceptual basis for each category. During categorization processing, the concepts contained in the documents categorized are compared to the concepts contained in the template items, and a category (or categories) is assigned to the documents based on the similarities between the concepts they contain and the concepts that are contained in the template documents.

LSI can realize effectively dynamic clustering based on the conceptual content of documents can also be accomplished using LSI. Clustering is a way to group documents based on their conceptual similarity to each other without using template documents to establish the conceptual basis for each cluster. This is very useful when dealing with an unknown collection of unstructured text.

LSI is not restricted only to working with words. It can also process arbitrary character strings. Any object if it can be expressed in text can be represented in an LSI vector space. For example, texts with MEDLINE abstracts have shown that LSI is able to effectively classify genes based on conceptual modeling of the biological information contained in the titles and abstracts of the MEDLINE citations.[11]

LSI has proven to be a useful solution to a number of conceptual matching problems.[12][13] The technique has been shown to capture key relationship information, including causal, goal-oriented, and taxonomic information.[14]

6 Conclusions

In this paper, we have presented an important scientific and technical aspect of Web search result clustering. The search results cluster entirely to fulfill the promise of being the PageRank of the future. First, more work needs to be done to improve the quality of the cluster labels and the coherence of the cluster structure. Second, more studies on user queries must be made to understand when clustering of search results is most useful. Third, there is a need for carefully engineered evaluation benchmarks to allow cross-system comparison, and to measure progress. Fourth, advanced visualization techniques might be used to provide better overviews and guide the interaction with clustered results.

Currently we are experimenting with LSI techniques for web documents mining. We are also planning to apply the LSI technique to web database. We firmly believe that semantic-based mining is a promising approach to significantly unleashing the power of both the World Wide Web and multimedia technology.

References

1. Pringle, G., Allison, L., Dowe, D.L.: What is a tall poppy among web pages? In: Proc. 7th IWWWC, Brisbane, Australia, pp. 369–377 (1998)
2. Hawking, D.: Results and challenges in web search evaluation. In: Proc. 8th IWWWC, Toronto, ON, Canada, pp. 1321–1330 (1999)
3. Schechter, S., Krishnam, M., Smith, M.D.: Using path profiles to predict HTTP requests. In: Proc. 7th IWWWC, Brisbane, Australia, pp. 457–467 (1998)
4. Gudivada, V.N., Raghavan, V.V.: Content-based image retrieval systems. IEEE Comput. 28, 18–22 (1995)
5. Dumais, S., Nielsen, J.: Automating the Assignment of Submitted Manuscripts to Reviewers. In: Proceedings of the Fifteenth Annual International Conference on Research and Development in Information Retrieval, pp. 233–244 (1992)
6. Yang, J., Watada, J.: Wise Mining Method with Ant Colony Optimization. In: IEEE International Conference on Systems, Man, and Cybernetics, USA, San Antonio, October 2009, pp. 1902–1908 (2009)
7. Karypis, G., Han, E.: Fast Supervised Dimensionality Reduction Algorithm with Applications to Document Categorization and Retrieval. In: Proceedings of CIKM-00, 9th ACM Conference on Information and Knowledge Management, pp. 12–19 (2000)
8. Bradford, R.: An Empirical Study of Required Dimensionality for Large-scale Latent Semantic Indexing Applications. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, California, USA, pp. 153–162 (2008)

9. Landauer, T.K., Dumais, S.T.: Latent Semantic Analysis. Scholarpedia 3(11), 43–56 (2008)
10. Landauer, T., et al.: Learning Human-like Knowledge by Singular Value Decomposition: A Progress Report. In: Jordan, M.I., Kearns, M.J., Solla, S.A. (eds.) Advances in Neural Information Processing Systems, vol. 10, pp. 45–51. MIT Press, Cambridge (1998)
11. Homayouni, R., Heinrich, K., Wei, L., Berry Michael, W.: Gene Clustering by Latent Semantic Indexing of MEDLINE Abstracts, pp. 104–115 (August 2004)
12. Ding, C.: A Similarity-based Probability Model for Latent Semantic Indexing. In: Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 59–65 (1999)
13. Bartell, B., Cottrell, G., Belew, R.: Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling. In: Proceedings, ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 161–167 (1992)
14. Graesser, A., Karnavat, A.: Latent Semantic Analysis Captures Causal, Goal-oriented, and Taxonomic Structures. In: Proceedings of CogSci 2000, pp. 184–189 (2000)
15. Aleman-Meza, B., Halaschek, C., Arpinar, I., Sheth, A.: A Context-Aware Semantic Association Ranking. In: Proc. First Int'l Workshop Semantic Web and Databases (SWDB '03), pp. 33–50 (2003)
16. Baeza-Yates, R., Caldero'n-Benavides, L., Gonza'lez-Caro, C.: The Intention behind Web Queries. In: Proc. 13th Int'l Conf. String Processing and Information Retrieval (SPIRE '06), pp. 98–109 (2006)
17. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: Proc. Seventh Int'l Conf. World Wide Web (WWW '98), pp. 107–117 (1998)
18. Junghoo, C., Garcia-Molina, H., Page, L.: Efficient Crawling through URL Ordering. Computer Networks and ISDN Systems 30(1), 161–172 (1998)
19. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Technologies Project (1998)
20. Pisharody, A., Michel, H.E.: Search Engine Technique Using Keyword Relations. In: Proc. Int'l Conf. Artificial Intelligence (ICAI '05), pp. 300–306 (2005)
21. Priebe, T., Schlager, C., Pernul, G.: A Search Engine for RDF Metadata. In: Galindo, F., Takizawa, M., Traunmüller, R. (eds.) DEXA 2004. LNCS, vol. 3180, pp. 168–172. Springer, Heidelberg (2004)
22. Rocha, C., Schwabe, D., Aragao, M.P.: A Hybrid Approach for Searching in the Semantic Web. In: Proc. 13th Int'l Conf. World Wide Web (WWW '04), pp. 374–383 (2004)
23. Stojanovic, N., Studer, R., Stojanovic, L.: An Approach for the Ranking of Query Results in the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 500–516. Springer, Heidelberg (2003)
24. Sun, R., Cui, H., Li, K., Kan, M.Y., Chua, T.S.: Dependency Relation Matching for Answer Selection. In: Proc. ACM SIGIR '05, pp. 651–652 (2005)
25. Tran, T., Cimiano, P., Rudolph, S., Studer, R.: Ontology-Based Interpretation of Keywords for Semantic Search. In: Proc. Sixth Int'l Semantic Web Conf., pp. 523–536 (2007)
26. Lei, Y., Uren, V., Motta, E.: SemSearch: A Search Engine for the Semantic Web. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 238–245. Springer, Heidelberg (2006)
27. Li, Y., Wang, Y., Huang, X.: A Relation-Based Search Engine in Semantic Web. IEEE Trans. Knowledge and Data Eng. 19(2), 273–282 (2007)
28. Stojanovic, N.: An Explanation-Based Ranking Approach for Ontology-Based Querying. In: Proc. 14th Int'l Workshop Database and Expert Systems Applications, pp. 167–175 (2003)

Pattern Recognition in Online Environment by Data Mining Approach

MohammadReza EffatParvar[1], Mehdi EffatParvar[2], and Maseud Rahgozar[1]

[1] Computer & Electrical Eng Department, University of Tehran, Tehran, Iran
mr.effatparvar@ece.ut.ac.ir,
rahgozar@ut.ac.ir
[2] Computer & Electrical Eng Department, Islamic Azad University
Ardabil Branch, Ardabil, Iran
effatparvar@iauardabil.ac.ir

Abstract. Nowadays, applying artificial intelligence to all aspects of human life is one of the most challenging efforts, including Robocup competitions. In this paper we will describe our research in case of using Expert System as a decision-making system. We made our attempt to expose a base strategy from past log files and implement an online learning system which receives information from the environment. Regarding to its world reputation, this article tries to investigate some parts of soccer coach rules and use up-to-date methods according to data mining and neural networks concepts. We study the type and time of occurring patterns in coach with our experience and ideas clearly. Also using suitable algorithms both in online and offline parts, we can improve the team precision and lead our team by analyzing opponent team.

1 Introduction

Soccer coach simulation was one of the leagues of Robocup competitions in which a simulated coach analyzes the match to detect occurrence of special patterns in opponent team to lead its own team [6]. This coach is an Agent that sends proper messages to players which are known as Multi Agents [3]. Coach simulation tries to imitate a real soccer coach by analyzing opponent movements and watching how they play a game. After that, it'll configure positions for all the robots and send proper messages to each of them. The coach can resend messages in specific time intervals regarding the quality of the game whether the robots play good or not. After sending messages to the players, robots can decide if they need to accept and execute the received command message or ignore it. We use a special programming language called Clang to communicate with players. In this article we introduce the precious algorithms which let the coach to detect the patterns in the game. By pattern we mean special skills from opponent team robots which coach should detect them and send the result to the server. Server is an online machine which acts as supervisor to the game rules and gathers complete information about the play ground and players. Then it will give this information to coach. It can also analyze the pattern recognition results just received from coach and check if they are correct or not. Currently in Robocup rules, the main goal of the coach is correct and on-time pattern recognition. The online part includes all the messages

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 137–145, 2010.

that coach transmits to robots during the game. The offline part is a Log file which is saved during the match for later analysis by the coach. According to the rule of soccer coach and because of importance of the time that a pattern occurs and is recognized by the coach, it is tried to improve coach responsiveness dramatically. This goal has been achieved by changing the threshold of pattern occurrence, using data mining methods to remove redundant messages and storing the relevant data in the data warehouse, and also applying neural networks to adjust the threshold and foreseeing the occurrence of a special pattern. An expert system is a program which attempts to mimic human expertise by applying inference methods to a specific body of knowledge. In this system the information is consistently sensed from the environment, and using a forward chaining method, system that compares data in the working memory against the conditions of the rules and determines which rules to fire, then the suitable advice is generated and sent to players. In this paper it has been tried to simulate some patterns. Later we'll describe the simulation method and its functions. In next section we describe the offline and online parts, section 3 shows the system architecture, section 4 describes the time prediction and its accuracy, in section 5 we show the determining threshold, section 6 presents our simulation result and finally we have conclusion and future work.

2 Offline and Online Parts

In each match, two teams play with each other. The one which is controlled by coach is called coach-able and the other one is called opponent. Each game takes 6000 time cycles and coach can send its messages to the robots in predetermined times. During the game, all the events, commands and messages transmitted between robots and coach are saved in a log file. In offline part, the coach can analyze the log file and improve its proficiency we call it offline learning. Therefore the main goal of offline learning is to read, parse and analyze the log file with RCG file extension. The result will be delivered to the coach to be used in future matches. The offline part is simpler compared to the online part but we should keep an eye on its results to avoid possible errors [2]. Online coach is divided to sense phase, expert system phase, message generation phase and act phase that first and second phase is more important.

Sense Phase. By a set of features in which consist: our score, opponent score, goal difference, cycle number, distance from each player to the ball, coordinate of the ball and coordinate of each player calculates some skills such as detecting formation, pass graph, each player's activity area, ball position, sequence possession and number of shoots to target then the results of these skills are used to predict opponent ability and generating appropriate advice.

Expert System. In this phase coach makes its effort to analyze and investigate abilities of the opponent team and finally models the state of match in order to provide a strategy against the opponent team. Unlike the offline part, the online part doesn't need to parse files. Using the real game and the art of the programmer, it can earn its required data and send proper commands to the robots. Note that functions and procedures should have the least possible overlap and complexity and all of them should regard the final target of the programmer to analyze the game and deliver the information to the coach

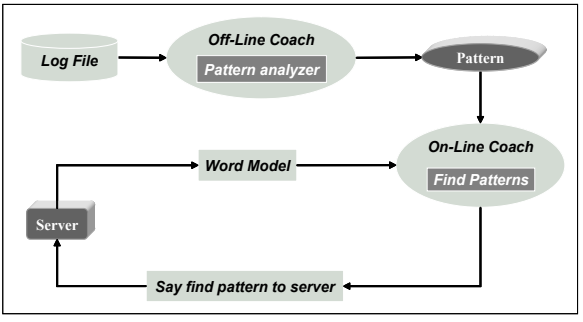


Fig. 1. The connection between online and offline parts

[10]. The other parts which are used by coach in online part is to appoint formation for his team by help of Clang language, appointment of each player duty according to their abilities, analyzing the opponent team during the game, updating all data's according to analyzed information and etc. There is a nearly connection between online and offline parts, so it is important that we have exact connection between them, you can see it in figure 1.

3 Overview on System Architecture and Choosing Useful Data

If we want to have a general view from special manner of connection between parts of data mining in online part at coach simulation, we should explain by this way that in online data mining there was not any data from past and all data are happened random in online environment. We can use these data for training the coach. For training the internal data mining in online environment we should use these data, so in this paper we used four internal layers for training, prediction and data mining.

The first layer that we call it pattern analyzer, is connected to the server directly and it main duty is gathering information from environment and analyzing the part of gotten information and transferring this information to other parts. The second layer is data warehouse, and its main duty is omitting the extra data and changing them to the suitable format for data mining. The third layer is called threshold unit; this unit after connecting to the server finds the online requirement and uses neural network for this matter, and then give these information to next unit. The fourth layer is called prediction layer that it can predict the time which pattern happened. It takes information from last units and by using decision tree predicts the time of pattern then it gives this time to other internal unit. If predicted time was true the kind of pattern reported to the server and if it was not corrected so the time had to correct and other time will be produce. Figure 2 show the general view of system architecture and its event.

As we know in the first stage data are not ready for data mining, so we should do some action in order to use them for data mining. About sixty percent of data mining time is belong to this part. In online environment data are produced periodically, so we should receive the useful data from the source and save them in the data store. It must be notice that in this project the server was used both as a database and producer the online

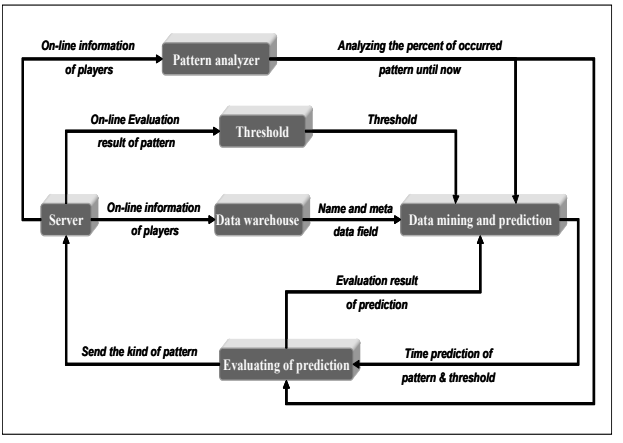


Fig. 2. System architecture and its events

environment. Although server gives much data to coach but all these data are not used in mining and predicting. So to decrease processing on plenary data and avoiding from wrong result, we should distinguish between useful and plenary data and save them in the data warehouse. Data of data store are not connected to the other systems directly so when an operational system wants to use data warehouse it should see data mart. To estimate the abilities of the opponent team, information such as ratio of shoots to goal (which amounts to number of attacks of either team), successful passes, ball possession and compression ratio in triple areas of field is investigated. We define some weights for the parameters mentioned above to calculate opponent ability using the following formula:

$$\sum_{i=1}^{max}\left(\left(\kappa_i / |\kappa_i|\right)\left[\beta_i + (|\kappa_i|)\times\alpha_i\right]\right)$$

max: number of parameters
β_i: base point for each parameter
k_i: difference between the parameters of our team in comparison with the opponent
α_i: point scored for each k_i

For next step, using a decision-making tree designed on the basis of rule-based expert system architecture, the opponent team is modeled.

4 Time Prediction and Evaluate the Accuracy

After finding the kind of pattern by pattern analyzer unit, now time prediction has high importance for us. For doing this we use prediction for finding time of happening pattern. We can use many manners like decision tree, neural network, clustering and etc for data mining [9]. Here we used decision tree for prediction because in this project this manner has higher speed than two other way and also it can train itself by less data

Table 1. Decision tree learnt and its IF-THEN rules for giving strategy

Assertions:
IF (GoalDifference $>$ 0) assert (Score Win)
IF (GoalDifference $<$ 0) assert (Score Lose)
IF (GoalDifference = 0) assert (Score Equal)
IF (OpponentAbility $>$ 0) assert (OpponentAbility Strong)
IF (OpponentAbility $<$ 0) assert (OpponentAbility Weak)
Rules:
IF (Score = Equal **AND** OpponentAbility = Strong)
OR
(Score = Win **AND** RiskTime = No **AND** OpponentAbility = Strong)
THEN switch to defensive mode
IF (Score = Win **AND** RiskTime = Yes)
OR
(Score = Lose **AND** RiskTime = No **AND** OpponentAbility = Strong)
THEN switch to absolute defensive mode
IF (Score = Equal **AND** OpponentAbility = Weak)
OR
(Score = Lose **AND** RiskTime = No **AND** OpponentAbility = Weak)
THEN switch to offensive mode
IF (Score = Lose **AND** RiskTime = Yes)
THEN switch to absolute offensive mode
IF (Score = Win **AND** RiskTime = No **AND** OpponentAbility = Weak)
THEN don't change strategy

and less time. For implementation of this method we use some agent. At first we should import data with less noise then we should classify the data [1]. In order to have data with less noise it is necessary to recall information from data warehouse such as data mart and evaluate them. The second part which has high importance in classifying is make suitable question. For each pattern we have separate question. In order to have these questions we should save some fields therefore we will have more suitable classify. We pay attention on data gathering or percent of data gathering in leaves to find important questions and if percent of data in selection time become less, we can get suitable percent by changing question. Last layer of decision tree has leaves that define some data. According to mass of data in each leaf, maximum percent of existing data is the reason to do that pattern.

Since all predicts are based on same reason, when percents of data that are used in leaf come in to threshold, we accept present time in to happening time of pattern. We can predict the time that pattern arrive in to threshold in future by having threshold and give it to predict unit in order to send it to server if it is true. Figure 3 shows the condition of decision tree and its connection with other unit.

Since each pattern starts from a time and increasing arrive to its maximum limit then after while it will be finish, we can understand this point that percent of improvement in each pattern is different in period of time so we cannot choose any especial percent for all time and games which are done by server. For improving predict situation, the decision tree always should learn from past and study results which are got from predict

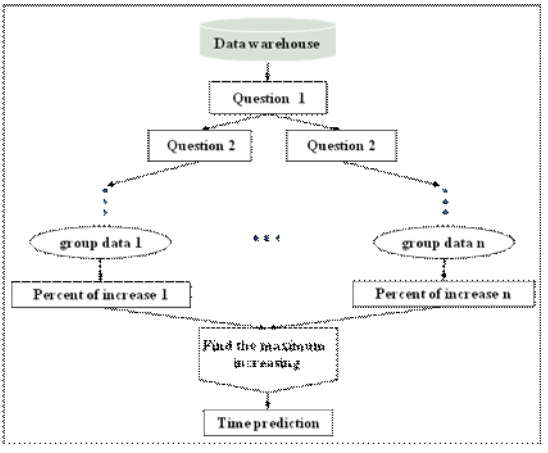


Fig. 3. Prediction method in decision tree

unit in order to find nearly time that pattern is happen [7]. Figure 4 shows a pattern that was started in tense of time and then finished. Choose predict unit for reporting kinds of pattern to server in present time.

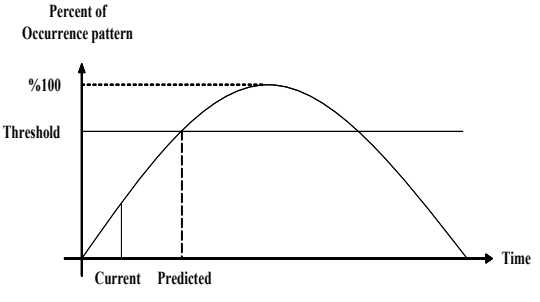


Fig. 4. Pattern increasing stages

After sending predict time from predict unit into predict evaluate unit, if the predication was true kinds of that pattern which is reported by predict unit, will send to server. For studying about predicted time by predict unit we need to threshold and percent of pattern that it send from predict unit into evaluating unit shows that "the predict time in pattern analyzer unit is as the same as threshold". If the time of pattern analyzer unit be equal or bigger than the threshold time this pattern send to server, otherwise a message will send to predict unit and this message reports mass of faults. Predict unit sends again new time by examination to evaluating unit. evaluating unit continue these manners until decreasing mass of wrong in to zero; when predict unit find enough accuracy we can stop checking predict time. In case percent of improvement faced to decrease, it means that the pattern is going to finish.

5 Determining the Threshold

Threshold is one of the important parts in this project and it is regular with time and also it is one of parameters effective in neural networks and data mining. Threshold is different in various environments and it will change according to the type and application of neural networks [11]. Threshold is minimum sensitive of server as compared with message which is sent from coach to find pattern, if type of pattern and its time are true, coach will get positive mark. We need one prime value for threshold at the beginning that it could be chose by person or program chooses accidentally percent for threshold. After choosing this value it will be sent to predict unit on order to specifying minimum time of happing pattern for threshold. Then threshold unit waits for receiving this time in order to gets true or wrong results of specify pattern from server. If server sent message based on wrong pattern it means that threshold considered less and we should increasing that and in opposite side it means that threshold is accepted. Otherwise we should decrease some limitation of threshold and study results again. We continue these manners until we will sure about the threshold. Because of being in online environment and in each time there are several pattern that are going to start or finish, we should continue this process until end of game.

For implementation of this part by neural networks we have several layers [4]. First layer is used for inputting the data and consist of 2 parts. One of them is results which got from the server and other is used for threshold in last stage. We charge these inputs in to a number between 0-1 in order to have high accuracy. Other layer is hidden layer, which is used for updating inputs weight. Third layer is output layer which is return threshold in to output.

There is a different between these manners with other manners in neural network in way of training neural network. For training in this network we can use results and information which is gotten from server such as online and also there are not any data that we are sure about them for experiment neural network. Because a neural network that used in online environment we should consider some extra parameter, so we can say that we have another type of neural network. Figure 5 shows the connection between system components to determining the threshold.

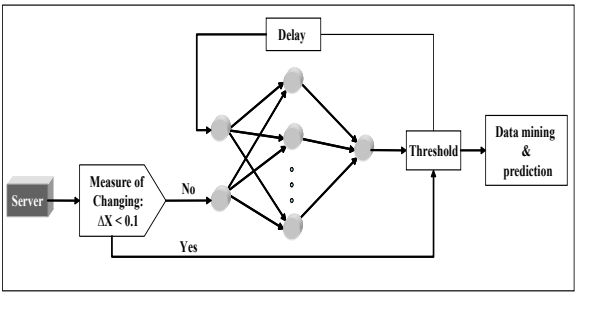


Fig. 5. Threshold determining with recurrent neural network

6 Simulation and Experimental Result

According to what explained above our goal is present useful and effective algorithm in order to specifying true existing patterns. In our simulation we used C++ language and Linux operation system. Mentioned algorithms are executed base on Aria team, which, it could get second place in international Japan competition in 2005. We reconstructed this team by our new methods. As you see, Table 2 shows performance of these algorithms in compare with original team. This tables show the improvement of coach in finding patterns.

Table 2. Comparing the pattern finding in two stages before and after training

intercept		hold		shoot		pass		Dribble		Kind of pattern
After	Before	After	Before	After	Before	After	Before	After	Before	Stages / Run
22 %	18 %	42 %	64 %	54 %	54 %	85 %	42 %	51 %	37 %	1
51 %	38 %	59 %	58 %	82 %	68 %	57 %	68 %	62 %	23 %	2
23 %	24 %	88 %	86 %	58 %	51 %	78 %	48 %	58 %	42 %	3
58 %	54 %	92 %	86 %	98 %	75 %	96 %	73 %	76 %	49 %	4

After using necessary methods and executing program on coach simulation for several times; results show that the coach can find the patterns in minimum time. Also table 3 shows the patterns finding time in use of neural network in compare with ordinary situation. Each execute consist of 15 pattern and we run it 4 times, in 92% this method can find pattern truly. After training the network these statistic was improved. The maximum accuracy that we got is 95% in 17 executions.

Table 3. Time of pattern finding in two stages, before and after training; simulation results showed by second

shoot				pass				Dribble				Kind of pattern
Result	After (s)	Result	Before (s)	Result	After (s)	Result	Before (s)	Result	After (s)	Result	Before (s)	Stages / Run
True	289	False	273.4	True	28.4	True	35.6	True	418.1	True	422.7	1
True	376.1	True	385.2	False	112	True	132.1	True	18.4	False	15.6	2
True	62.6	True	76.5	True	561.8	True	562.2	True	157.5	False	123.3	3
True	402.8	False	428	True	382.4	False	345.7	True	205	True	211.2	4

7 Conclusion and Future Work

Coach simulation was one of the Robocop competitions that it simulates the football coach duties. It tries to analyze the opponent team and its own team. After analyzing it chooses the best formation according player abilities, so the coach should be can

analyze the players too. In this paper we explain the coach simulation structure and we described the important parts of it, also we showed the connection of its parts. Our prospective effort is to develop a learning system which is able to generate advice in online mode with usage of algorithm for modeling the opponent team. We also plan and try to improve and develop learning algorithms and find out better solutions for generating advice.

Some of algorithm which is used for finding pattern is also explained completely and showed the improvement of coach simulation. This paper is out research result which could get second place in Japan international competition in 2005, our aim is finding more patterns and decreasing necessary time for finding pattern in future works. According to the passage we can say using data mining is one of the important and useful ways for deciding in noisy environment, so using of data mining and intelligent methods such as neural network and decision tree can improve our work. By the help of decision tree with doing true analysis and explaining suitable question can save important part of information and omit extra parts, then it will be easy to divide and classify data by suitable speed. In other words, we can use this way in order to an original manner in all online environments. By merging the online and offline part we can design a new architecture and we can improve the coach simulation to find pattern in best time.

References

1. Berry, M., Linoff, G.: Data Mining Techniques: For Marketing, Sales, and Customer Support. John Wiley and Sons, New York (1997)
2. Boer, R., Kok, J., Groen, F.: The UvA Trilearn 2001 Robotic Soccer Simulation Team. BN-VKI Newsletter 18(4) (August 2001)
3. Wooldridge, M.: An Introduction to Multiagent Systems. John Wiley and Sons, Chichester (February 2002)
4. Guestrin, C., Koller, D., Parr, R.: Multiagent planning with factored MDPs. In: Advances in Neural Information Processing Systems (NIPS-14), Canada (December 2001)
5. Cutting, D.R., Karger, D.R., Pedersen, J.O., Tukey, J.W.: Scatter/Gather, a cluster-based approach to browsing large document collections. In: Proceedings of the 15th International ACM, SIGIR Conference on Research and Development in Information Retrieval, pp. 318–329 (1992)
6. Catlin, M.G.: The Art of Soccer. Soccer Books (1990)
7. Kaufman, L., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, Chichester (1990)
8. Buckley, C., Salton, G., Allen, J., Singhal, A.: Automatic query expansion using SMART. In: The Third Text Retrieval Conference (TREC-3). U.S. Department of Commerce (1995)
9. An Introduction to Data Mining, http://www.thearling.com/text/dmwhite/dmwhite.htm
10. Fathzadeh, R., Mokhtari, V., Shahri, A.: Coaching with Expert System implemented in Robocup Soccer Coach Simulation. In: Bredenfeld, A., et al. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, pp. 488–495. Springer, Heidelberg (2006)
11. Fathzadeh, R., Mokhtari, V., Kangavari, M.R.: Opponent Provocation and Behavior Classification. In: Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.) RoboCup 2007. LNCS (LNAI), vol. 5001, pp. 540–547. Springer, Heidelberg (2008)

Part IV

Agents Mining Applications

A Multiple System Performance Monitoring Model for Web Services

Yong Yang, Dan Luo, and Chengqi Zhang

Data Sciences & Knowledge Discovery Lab,
Center for Quantum Computation and Intelligent Systems.
Faculty of Engineering & Information Technology,
University of Technology, Sydney
{yongyang,dluo,chengqi}@it.uts.edu.au

Abstract. With the exponential growth of the world wide web, web services have becoming more and more popular. However, performance monitoring is a key issue in the booming service-orient architecture regime. Under such loosely coupled and open distributed computing environments, it is necessary to provide a performance monitoring model to estimate the likely performance of a service provider. Although much has been done to develop models and techniques to assess performance of services (i.e. QOSs), most of solutions are based on deterministic performance monitoring value or boolean logic. Intuitively, probabilistic representation could be a more precise and nature way for performance monitoring. In this paper, we propose a Bayesian approach to analyze service provider's behavior to infer the rationale for performance monitoring in the web service environment. This inference facilitates the user to predict service provider's performance, based on historical temporal records in a sliding window. Distinctively, it combines evidences from another system (For example, recommendation opinions of third parties) to provide complementary support for decision making. To our best of knowledge, this is the first approach to squeeze a final integrated performance prediction with multiple systems in Web services.

1 Introduction

Web services have become a promising technology for e-business and e-commerce. In such an open and volatile environment, possibly a service provider may not deliver the quality of service (QoS) it declared [1]. In the online business world, it is necessary for participants to estimate each other's performance and predict behavior of web services before initiating any commercial transactions [2,3]. Therefore it is important to monitor the service provider's performance before and during enjoying its so-claimed services.

Normally a service's performance is temporary and variable [4]. It can hardly be accurately indicated from information available because of not only incompleteness of data but also instinct uncertainty involved. For instance, the service may be delayed by incoming real-time tasks. In other words, giving a probabilistic description is more practical than a deterministic answer when estimating the performance [5,6]. The performance monitoring model proposed in this paper allows users to predict the future behavior of a service in such a way:

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 149–161, 2010.

The probability that downloading a DVD movie of *Avatar* for *bronze* membership service within 15 minutes is at least 90%.

Our model is based on Bayesian network [7,8,9,10]. A "Bayesian network" (BN) or "belief network" is a probabilistic graphical model whose nodes represent events and arcs depict probabilistic dependency relations among the events via a directed acyclic graph. Take diseases diagnosing in a medical domain as an example. A Bayesian network could represent the probabilistic relationships between diseases and symptoms, and compute the probabilities of various diseases from shown symptoms. Similarly by employing a Bayesian network, we can predict service's future behavior based on the observed historical evidences.

Indeed, most of Qos assessment algorithms tend to ignore user-specific preferences (different subitem service levels) [11,12,13]. For example, a download website service may provide quicker speed for paid members and much limited bandwidth for anonymous users. For example, at *Easy-share* [14] and *MEGAUPLOAD* [15] a premium user is given much higher priority. With regard to premium users, there are 1TB online storage with *filemanager*, whereas for non-members with none. Therefore, simply taking an average of them would neglect the relevance of user's individual feature/situation, resulting in an incorrect assessment of Qos.

Another issue worth considering is service' performance monitoring can be complemented by opinions from third parties [16,17,18]. For example, if we ignored the fact that another user left a message saying the movie is indistinct and favorless, we probably be on the road to "ruin". In other words, the performance monitoring computing can be more comprehensive when taking into account evidences from another system. In a real world situation under various restrictions, only part of evidences might be observed by the user himself or herself whereas others are done by another system such as recommendation, which surely provides useful supplementary information for decision-making. As far as we know, opinions on performance from another system could be an important complement and should be fused to get a combination value for performance monitoring computing.

This paper exploits the problem of performance monitoring prediction by user's previous behavior evidences. The theory of Bayesian network gives us robust prediction theoretical tool, which combines graph theory and probability theory. We can use it to flexibly predict service's future performance on different dimensions, which provides more accurate predictions given the circumstances. The use of Bayesian networks is similar to that of expert system technologies. A Bayesian network represents beliefs and knowledge about a particular class of situations. Given a Bayesian network (i.e. a knowledge base) for a class of situations and evidence (i.e. observations or facts) about a particular situation in that class, conclusions can be drawn in such a situation. Besides, a service may provide some functional and value-add features. These different level of services can be modeled as different status of corresponding variables in the network. And the conditional probability tables of nodes express the predicted performance in the form of probability, which provide sound basis to facilitate the user's decisions. In the meantime, when selectively integrating the cases from external systems into our Bayesian network, opinions from another system are fused to compute the complementary performance monitoring.

To sum up, this paper makes the following main contributions:

- A Bayesian network is introduced and the performance monitoring is represented by a probability value. Furthermore, performance monitoring under different dimensions is discussed and a Bayesian network performance monitoring model is proposed.
- We design a multiple-system combination performance monitoring algorithm to fuse evidences from both internal and external systems. It is capable of providing more complementary and performance trustworthy predictions.

2 Probabilistic Performance Monitoring Estimation in Bayesian Networks

In this section, we will first give a brief definition to our Bayesian network. Then a typical application scenario (a file download web service) will be described as an illustration. After formalizing Bayesian networks on two different levels, we will discuss performance monitoring in a multiple-system environment.

2.1 Overview of the Performance Monitoring Model

First of all, we define a Bayesian network generally. A Bayesian network consists of:

- A set of variables and a set directed edges between variables
- Each variable has a finite set of states
- The variables together with the directed edges from a directed acyclical graph (DAG) [19].
- For each variable A with parents $B_1, .., B_n$ there is an attached conditional probability table Pr(A|$B_1, ..., B_n$)

The fundamental rule for probability calculus is $P(a \mid b)P(b) = P(a, b)$, where $P(a, b)$ is the probability of the joint event $a \bigwedge b$, $P(a \mid b)$ is the probability of a given b.

Remembering that probabilities should always be conditioned by a context c, the formula should yield the well known Bayes's rule

$$P(b \mid a) = \frac{P(a \mid b)P(b)}{P(a)} \tag{1}$$

. Bayes's rule conditioned on c reads

$$P(b \mid a, c) = \frac{P(a \mid b, c)P(b \mid c)}{P(a \mid c)} \tag{2}$$

$P(b)$ is the prior probability of hypothesis b; $P(a)$ is the prior probability of evidence a; $P(b \mid a)$ is the probability of b given a. $P(a \mid b)$ is the probability of a given b.

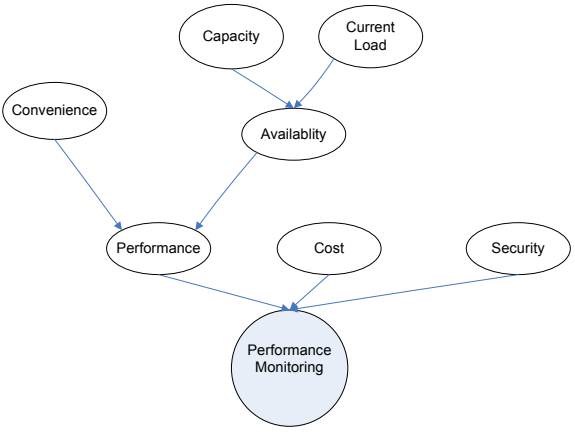


Fig. 1. Bayesian network performance monitoring model

In causal networks, the basic concept in the Bayesian treatment of certainties is *conditional probability* [20]. Whenever a statement of the probability $P(a)$ of an event a is given, then it is given conditioned by other known factors.

Let's take file provider as an example. In the online world, file providers' competence are not uniform. For example, some file providers may link to a high-speed fabric network, while others connect through a slow modem. Some file providers might like music, so they share a lot of music files. Some may be interested in movies and share more movies. Some may be very picky about file quality, so they only keep and share files with high quality. Therefore, the file provider's capability can be presented in various aspects, such as performance, cost and security.

The Bayesian network performance monitoring model is depicted in Figure 1, which looks like a upside down tree. The root node denotes the service capability or performance monitoring value for this service, and element nodes denote users' various requirements to each Qos property. For simplicity, we assume that these Qos properties do not influence each other (That is, all the metrics are independent when estimating service's performance). In order to use BN to predict the service capability, some continuous features, such as availability and current load, should be discretized. Two approaches are presented to discretize continuous features in this article. The first approach simply specifies different levels for each Qos property in advance. For example, three levels, $0 \sim 0.5$ as low, $0.5 \sim 0.8$ as medium and $0.8 \sim 1$ as high, are specified to availability property. The second approach adopts statistical threshold.

As for the example, we simply divided the performance monitoring after each invocation into two categories: *Positive,Negative*. Likewise, *Cost* has three values: *premium membership, member, non-member*. They represent user's grades (different levels of membership) regarding to the server.

The nodes under the root node represent the file provider's capability in different aspects. Each of them is associated with a **conditional probability table** (CPT). Its CPT is showed in Figure 2. Each column follows one constraint, which corresponds to one value of the root node. The sum of values of each column is equal to 1.

	$PerformanceMonitoring = Positive$	$PerformanceMonitoring = Negative$
$Premium$	*p(C="Pr"|T="Po")*	*p(C="Pr"|T="Ne")*
$Member$	*p(C="Me"|T="Po")*	*p(C="Me"|T="Ne")*
$Nonmember$	*p(C="No"|T="Po")*	*p(C="No"|T="Ne")*

C=cost,T=Performance Monitoring.

Fig. 2. CPT table of performance monitoring node

$p(C = \text{“}Pr'' \mid T = \text{“}Po'')$ is the conditional probability with condition that an invocation quality is judged to be positive. It measures the probability that user's levels involved in a download, given the downloading performance is satisfying. For example, the bronze user's given performance is excellent is 0.8. After each download, the network is updated. It can be calculated according to the following formula:

$$P(C = \text{“}Pr'' \mid T = \text{“}Po'') = \frac{P(C = \text{“}Pr'', T = \text{“}Po'')}{P(T = \text{“}Po'')} \tag{3}$$

$P(C = \text{“}Pr'', T = \text{“}Po'')$ is the probability that the service capability is excellent and the availability requirement is high in the past invocations.

$$P(C = \text{“}Pr'', T = \text{“}Po'') = \frac{m}{n} \tag{4}$$

m is the number of invocations that the service capability is excellent and the user's availability requirement is high. n is total invocation numbers of the service.

The root element P(T="Po") represents the probability that the service performance is regarded to be positive.

$$P(T = \text{“}Po'') = \frac{p}{n} \tag{5}$$

p is the number of invocations that invocation is regarded to be good.

P(C="Pr") is the probability that service'availability is Premium level, which is calculated according to the following formula:

$$P(C = \text{“}Pr'') = \frac{r}{n} \tag{6}$$

r is the number of invocations that the service's availability is premium level.

The conditional probability functions can be computed before next invocation as follows:

$$P(T = \text{“}Po'' \mid C = \text{“}Pr'') = \frac{P(C = \text{“}Pr'' \mid T = \text{“}Po'')P(T = \text{“}Po'')}{P(C = \text{“}Pr'')} \tag{7}$$

These predictable information is extremely important for a user's decision making. For example, we can clearly tell in what probabilities the bronze or premium level of service can have some quality of service, which is very helpful for the user to decide whether it is worth to choose them.

The variables can have internal relationship between them. For example, some of the web servers will lose its availability during the peak even for its remarkable capacities. That is, *availability* is jointly determined by *capacity* and *current-load*. We use *hierarchical* structure to reflect and represent these features.

2.2 General Theory

Many existing probabilistic models use naive Bayesian networks. We extend it to hierarchical structure, which is believed to be more precise to the real world. Based on the analysis of factors contributing to web service's performance, the formalized structure for our performance monitoring model is shown in Figure 3. $M_{11}, ..., M_{1a}$ represent all the aspects for evaluation of performance monitoring. Any leaf nodes in the tree may be defined recursively. However we limit our discussion to two level Bayesian networks, as any complex BN graphs can be equivalently splitted into a combination of two level models.

The all aspects regarding to performance monitoring is denoted as set of aspects $\Psi = \pi_1, \pi_2, ..., \pi_p$. The set of agents: $A = a_1, a_2, ...$ and $B = b_1, b_2, ...$ denotes the agents who have direct relationship with the target service. $C = c_1, c_2, ...$ denotes the agents who have indirect relationship with the target service. D denotes the rest of them.

The set of ratings p can be viewed as a mapping from $A \times \Psi$ to a real number between 0 and 1:

$$p = A \times \Psi \rightarrow [0, 1] \tag{8}$$

The chain rule for Bayesian networks. Let $U = \{A_1, ..., A_n\}$ be a universe of variables. Then the joint probability distribution $P(U)$ is the product of all potentials specified in BN

$$P(u) = \prod_i P(A_i \mid pa(A_i)) \tag{9}$$

where $pa(A_i)$ is the parent set of A_i.

We can use Eq. 9 to estimate the uncertainty of variables we are interested in. Tractability is not a yes or no issue. Intuitively, these equation still hold even for hierarchical network. After each interactive with this web service, the performance monitoring network is updated. By cumulative effects of the evidences, we can predict service's future behavior. This procedure is named *training*. Furthermore, with this prediction, we can build user-based performance monitoring model for this services. Based on user's history, service can offer user-specified intelligent service which is greatly convenient for the user and assist them with the proper decision.

There are algorithms for probability updating in bayesian networks, however, basically probability updating in NP-hard. This means that some models have an updating time exponential in the number of nodes.

2.3 Representing Multiple-System Combination Performance Monitoring in Bayesian Networks

When computing the performance, it is important to combine the evidences from another system, such as recommendations. In other words, the predicted performance can be refined by considering recommendations.

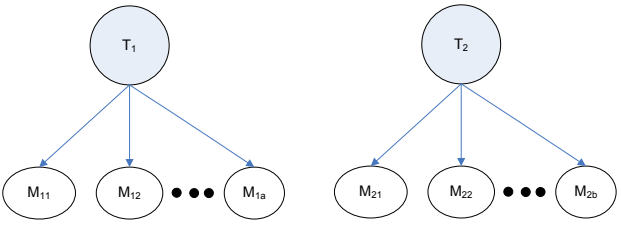


Fig. 3. Two level bayesian networks

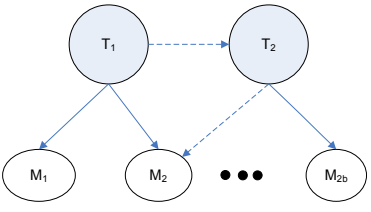


Fig. 4. Bayesian network with combination performance monitoring

We use our bayesian network to alert the user's requirements whether there is any change about certainty. This network can be used to follow how a change of certainty in one variable may change the certainty for other variables. We propose a set of rules for that way of reasoning. The rules are independent of the particular calculus for uncertainty.

For example, service 1 is a movie download web service. The opinions from users may leave comments for it (e.g., the downloading speed is rather slow or this movie is poor-quality). Afterwards that information clearly is helpful for users to estimate the performance monitoring. Internal and external evidences are merged and thereby the performance monitoring model can be refined to Figure 4. The solid lines represent all the direct invocations with the target service. In contrast, the solid lines represent indirect experiences with target services.

$$T = (1 - \theta)T_B + \theta T_C \tag{10}$$

where $\theta \in [0, 1]$. The weight θ represents the importance of these two probabilities respectively and are decided by the personal characteristics of the entity.

With respect to modeling the recommendation, we use the following formula to its estimator:

$$p(\hat{\theta}) = Beta(c_1, c_2) \tag{11}$$

where c_1 and c_2 are parameters determined by prior assumptions and $\hat{\theta}$ is the estimator for the true proportion of a's approval of b based on invocation between them. Assuming that each invocation's approval probability is independent of other invocations between a and b, the likelihood of having p approvals and (n-p) disapprovals, which can be modeled as:

$$L(D|\theta) = \theta^p(1-\theta)^{n-p} \tag{12}$$

$$p(\theta|D) = Beta(c_1 + p, c_2 + n - p) \tag{13}$$

$$E[\theta|D] = \frac{c_1 + p}{c_1 + c_2 + n} \tag{14}$$

$$\sigma^2_{\theta|D} = \frac{(c_1 + p)(c_2 + n - p)}{(c_1 + c_2 + n - 1)(c_1 + c_2 + n)^2} \tag{15}$$

How is the prior belief about θ created? There are two ways:

$$p(\theta) = \begin{cases} 1 & 0 < \theta < 1 \\ 0 & otherwise \end{cases} \tag{16}$$

For the Beta prior, values of $c_1 = 1$ and $c_2 = 1$ yield a uniform distribution.

After each invocation, the client updates θ and corresponding knowledge in the Bayesian network. It is a new case for this learning. But it is often a problem for updating the CPTs in the Bayesian network because the initial counts are kept when the network try to accommodate new cases. Particularly when the conditional probabilities in the environment change over time, the accumulated counts will prevent the network from new changes. Therefore, the vacuous counts will build up to make parameters too resistant to change. It is known that the performance monitoring of service provider is on a temporary basis as the behavior of service provider is constantly changing with time. The value of past performance monitoring does not represent the current performance monitoring. To this end, we set up a parameters k as the **window size** of the learning procedure. All the case are updated in the fixed size window. When the cases in the window is greater than k, the newest case is put into the window and the oldest one are swapped out. The performance monitoring in Eq. 10 is described below. In order to avoid the duplicate cases observed by indirect and direct users, a hash table is used to record what has been observed by B and avoid any duplicates.

Algorithm 1. Bayesian network performance monitoring

Input: Initial setting, structure of Bayesian network,invocation records.
Output: $PerformanceT$

(a). setup and system initiation. Set the window size. Create a table HashTl for the store the cases objected.
(b). compute T_B using bayesian network.
(c). compute T_C for recommendation from others.
(d). compute T for this invocation.
(e). search Hash(case) in HashTl, if there is a hit, ignore it and return step 2. Otherwise add a new record to HashTl and do the invocation.
(f). After each invocation, the learning algorithms is used to update the Bayesian network and compute T.

Probability learning algorithm of bayesian network is described below (We assume that the network itself is constructed using a priori knowledge):

Algorithm 2. Bayesian network learning

Input: T_n
Output: $T_{(n+1)}$

(a). compute the probability of $T_{B(n+1)}$
(b). if $|T_{Bn} - T_{B(n+1)}| < |T - T_{B(n+1)}|$ then
    n++;
    compute Eq. 13
  else
    n++;
    p++;
    compute Eq. 13.
  end if
(c). compute the new θ and $T_{(n+1)}$.

Then we take priority of evidence from set B because people trust themselves intuitionally. But on the other hand, they probably can not observe all the cases from the target service. Under that circumstantiates, the recommendation plays an important role in the performance monitoring estimation. In order to speed up the classification of cases, a hash table data structure is designed in table 1.

Table 1. Hash table H for case history

id	$casesummary$	$Time$	$HashValue$
1	...	*081022:2456*	*FFEFs3va*
2	...	*081022:2456*	*df12456*
3	...	...	...

3 Analysis

A Bayesian network serves as a model for networks occurring in trust estimation, and the relations in the model reflect causal impact between events. The reason for building these computer networks is to use them when taking decisions. The probabilities provided by the network are used to support decision making. To that end, we can distinguish forward-looking and backward-looking approaches. Although the examples presented in this paper are very elementary only, they serve as building blocks for these types of decisions.

When making decision in a complex situation, a proper loss function as well as a set of possible actions have to be taken into consideration. We make a distinction

here between intervening actions, which force a change of state for some variables in the model, and non-intervening actions, whose impacts are not a part of the model. Intervening action may also force the direction of causality to be inverted. It is beyond the scope of this paper to extend much beyond this notion. The interested reader is refereed to [8] for further study.

A Bayesian network is also useful for illustrating the effects of different scenarios. For that purpose, stochastic simulations are useful. It can then be analysed what the effects are if some actions are taken. For example, the effects of lowering a threshold on the *availability* feature of a service may be illustrated.

Finally, Bayesian networks can fit well into a built-in hierarchical model. Hierarchical structure is very useful considering the complex relationships among real world factors.

4 Case Study

In this section we present the work done as a case study on real-life data. To evaluate the effectiveness of our approach in estimating trust probability in different web service scenarios, we implement a prototype in Java. The system architecture is depicted in Figure 5.

We have carried out several experiments to study the effectiveness of our model. The environment is file downloading web services running on a Pentium computer with 3.40 GHz dual CPUs. The RAM is 1 G. and LAN connection is 100 M bps Ethernet card. Operation system is windows XP with SP3 pack. The service's performance we predicted is described as a trust value (boolean variable, positive or negative).

The first study is to check whether our model can identify user-specific preference. We invoked the service for 100 times with difference system trust. Table 2 shows the

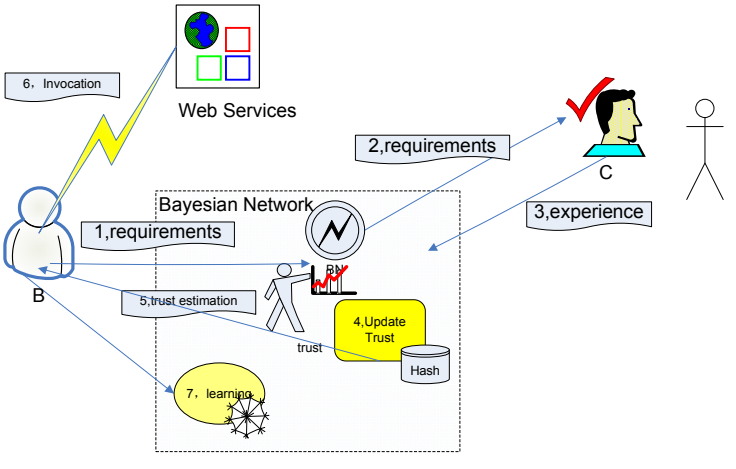


Fig. 5. System architecture

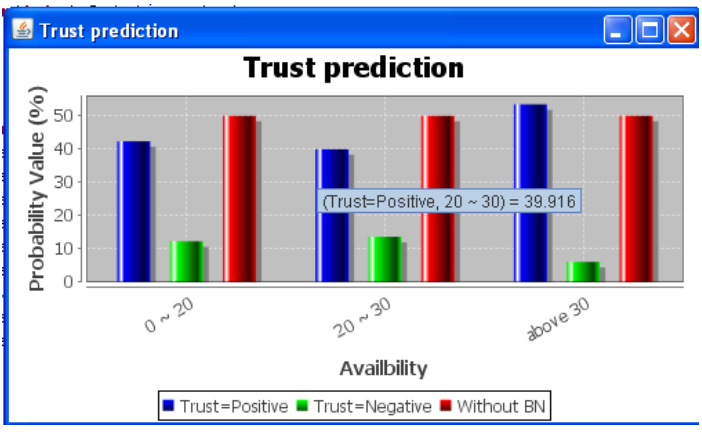


Fig. 6. Prediction result of bayesian network

Table 2. Service with two different levels

	Availability		
	Min	*Max*	*Average*
Level1	*0*	*20*	*10*
Level2	*20*	*30*	*25*

two different service levels with availability feature. The user-specific duration was modeled into 2 levels: $0 \sim 20$ (level 1) and $20 \sim 30$ (level 2). When we did the 1000 invocations, we calculated the posterior probability distribution with 3 intervals: (0,20),[20,30),[30,∞).

Fig. 7. Trust estimation with/without BN

The prediction result by Bayesian network is calculated in Figure 6. It can provide more informative prediction by considering user's preferences. For instance, there is 39.916 percent of chance that overall judgement of this service's performace is positive given that availability is in the interval $20 \sim 30$.

The next experiment is to see if our BN model has improve the performance estimation of the services. The BN model is compared to without BN selection, in Figure 7. Our service provider provides level 1 availability services (within $0 \sim 20$) for 800+ invocations. (The small red squares depict introduced evidences assumed from third parties.) The result shows that BN model presents a more stable and reliable (closer to true value, fitting in with availability *Level 1* range) description about service's performance.

5 Conclusion

Bayesian network is a probabilistic model to represent relations between attributes and classes. We have described an application of Bayesian networks, a relatively new technology for probabilistic representation and inference, to web service performance monitoring. The advantages that Bayesian networks bring to the performance prediction task include an intuitive representation of uncertain relationships and a set of efficient inference algorithms. Probabilistic models view performance prediction as a problem of estimating the probability that the service provider matches or satisfies user's requirements. The salient feature of this model is that it can correctly predict the service performance on the consideration of user's specific requirements. Furthermore, this model combines evidences from multiple sources, which places higher level of trust intuitively.

Our network structure is a complete graph. A disadvantage of this model is that the representation of this model is exponential with respect to the number of attributes. Consequently, it is difficult to estimate exponential number of parameters from limited data and perform computations with this model.

As future work, we plan to explore our model into composition of web services.

References

1. Menascé, D.: QoS Issues in Web Services. IEEE Internet Computing, 72–75 (2002)
2. Dasgupta, P.: Trust as a commodity. Trust: Making and Breaking Cooperative Relations, 49–72 (1988)
3. Baier, A.: Trust and Antitrust. Ethics 96(2), 231 (1986)
4. Zak, P.: Trust: A temporary human attachment facilitated by oxytocin. Behavioral and Brain Sciences 28(03), 368–369 (2005)
5. Hilden, J., Habbema, J., Bjerregaard, B.: The measurement of performance in probabilistic diagnosis. II. Trustworthiness of the exact values of the diagnostic probabilities. Methods Inf. Med. 17(4), 227–237 (1978)
6. Despotovic, Z., Aberer, K.: A Probabilistic Approach to Predict Peers' Performance in P2P Networks. LNCS, pp. 62–76. Springer, Heidelberg (2004)
7. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, San Francisco (1988)
8. Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer, Heidelberg (2001)

9. Pearl, J.: Causality: Models, Reasoning, and Inference. Cambridge University Press, Cambridge (2000)
10. Heckerman, D., Geiger, D., Chickering, D.: Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20(3), 197–243 (1995)
11. Welch, L., Shirazi, B.: A Dynamic Real-Time Benchmark for Assessment of QoS and Resource Management Technology. In: The IEEE Real-Time Technology and Applications Symposium, pp. 36–45 (1999)
12. Ito, Y., Tasaka, S.: Quantitative assessment of user-level QoS and its mapping. IEEE Transactions on Multimedia 7(3), 572–584 (2005)
13. Bernardi, S., Merseguer, J.: QoS Assessment via Stochastic Analysis. IEEE Internet Computing, 32–42 (2006)
14. Easy Share, http://www.easy-share.com/
15. Megaupload, http://www.megaupload.com/
16. Doney, P., Cannon, J.: An Examination of the Nature of Trust in Buyer-Seller Relationships. Journal of Marketing 61, 35–51 (1997)
17. La Porta, R., De Silanes, F., Shleifer, A., Vishny, R.: Trust in Large Organizations. In: Nber Working Paper (1996)
18. Massaro, D., Friedman, D.: Models of integration given multiple sources of information. Psychological Review 97(2), 225–252 (1990)
19. Diestel, R., Diestel, R.: Graph theory. Springer, New York (2000)
20. Jensen, F.: An introduction to Bayesian networks. UCL Press, London (1996)

Implementing an Open Reference Architecture Based on Web Service Mining for the Integration of Distributed Applications and Multi-Agent Systems

Dionisis D. Kehagias[1], Dimitrios Tzovaras[1], Efthimia Mavridou[1,2], Kostantinos Kalogirou[3], and Martin Becker[4]

[1] Centre for Research and Technology Hellas,
Informatics and Telematics Institute,
57 001, Thermi, Greece
diok@iti.gr, dimitrios.tzovaras@iti.gr
[2] Aristotle University of Thessaloniki,
Department of Electrical and Computer Engineering,
54 124, Thessaloniki, Greece
efi@iti.gr
[3] Centre for Research and Technology Hellas,
Hellenic Institute of Transport,
57 001, Thermi, Greece
kalogir@certh.gr
[4] Fraunhofer Institute for Experimental Software Engineering, Fraunhofer-Platz 1,
67663, Kaiserslautern, Germany
martin.becker@iese.fraunhofer.de

Abstract. This paper introduces an open reference architecture that enables seamless integration of distributed applications and agent-based systems. The reference architecture has been designed in order to be holistic, open and standards-abiding. It is based on an ontological framework that operates as a middleware between application developers and service providers on the one side and multi-agent systems on the other one. The proposed architecture enables seamless integration of closed (standalone) applications, provided that these applications export their functionality in the form of public web services. Moreover, we propose a data mining framework that operates on web service data and performs classification of web services and their operations into their semantically described counterparts. This enables seamless integration of applications through the corresponding web service interfaces, based on the ontological framework. On the other hand we show that knowledge derived from web service mining can be used by multi-agent systems in order to provide composite functionalities derived from the distributed web service operations. Moreover, agents are capable to provide personalized services that fulfill user requirements, by taking into account the personal profile, as well as the history of the end user. The paper presents the prototype tools that have been implemented for the realization of the proposed reference architecture.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 162–177, 2010.

1 Introduction

Currently a plethora of heterogeneous, standalone or web-enabled applications exist providing various functionalities that could be exploited in innumerable contexts for the development of personalized agent-based solutions for the end user. The integration of these applications in a standard and seamless way to enable content-rich services for the end-user is not generally feasible. The main reason for this is that these applications are by nature heterogeneous, developed for different development plat-forms, using different software development technologies. In this paper we present a reference architecture and support tools designed to address the problem of seamless integration of heterogeneous software applications through data mining (DM) on web service (WS) data ("web service mining") in order to enhance personalization, pervasiveness and efficiency on behalf of agent-based end-user applications.

Work presented in this paper is part of a European funded project called OASIS [1], whose main objective is the implementation of an ontology-driven, open reference architecture, which will enable and facilitate interoperability, seamless connectivity and sharing of content between different services and ontologies in application domains for the elderly and beyond. OASIS promotes new ways to integrate all supported applications into a common environment that enables access to information and content from the existing applications through WS-based software interfaces and content delivery to end-user in a pervasive manner through multi-agent applications. The achievement of this objective forms the main motivation of our work that results in a WS mining framework for the delivery of personalized services to the elderly users through a multi-agent system (MAS).

WS mining concerns the use of DM techniques on WS data for extracting useful knowledge. In [1] WS mining is used to increase the possibilities to compose useful services from existing services and particularly it is applied on biological process for the discovery of useful pathways. Another application of WS mining concerns the discovery of useful execution patterns among WS usage and interaction [2]. None of these approaches involves interaction of WS mining with agents or MAS. From an application point of view they are considered more as DM-oriented applications rather than agent-oriented ones. Our approach executes DM on WS in order to semantically categorize WS in terms of an ontology shared by agents. In this way, agents are facilitated on the process of WS discovery by executing appropriate semantic queries to the common ontology, that return the real services that fulfill user requirements.

One similar approach that combines WS mining and agents has been noted in [3] where agents are used in order to perform WS mining processes for the process of service discovery according to a set of user preferences. The main difference of this approach compared to ours is that it uses agents to perFform WS mining, while in our approach WS mining is not performed by agents, but in order to be used by agents.

Focusing on the details of the deployed DM mechanism, an approach for SOAP[2] WS classification, which is similar to the one presented in this paper is the one adopted by the MWSAF tool [4], based on schema matching. Specifically, MWSAF matches

[1] OASIS project official website: `http://www.oasis-project.eu`

[2] `http://www.w3.org/TR/soap12-part1/`

documents written in the Web Service Description Language (WSDL) to concepts from an appropriate ontology by converting both WS and ontologies to corresponding common graph-like structures. Similarities between graphs are then extracted by running matching algorithms on the graphs and the matching with the highest similarity is finally selected. An improved version of this approach was proposed [5] that uses a machine learning technique. Another tool called ASSAM [6] uses an ensemble machine learning approach for WS classification. Apart from the aforementioned WSDL-based categorization techniques, there are also additional approaches that use external data from non-WSDL sources [7,8].

Based on the OASIS architecture that we present in this paper, our contribution in the field of DM and agents interaction is that we provide a reference architecture and describe one of its possible implementations, through which WS mining can be applied for the seamless integration of applications to be consumed by agents and MAS that represent the end-user personalized services. Our goal in this context is to present a live prototype that realizes DM on WS data for the creation of a knowledge base in the form of a set of inter-connected ontologies that are shared between agents with the purpose to exploit the new knowledge for enabling new applications and services for the elderly.

In what follows we present the OASIS architecture and describe its major components and functionality. Later on, we show how all these components are orchestrated within the common reference architecture and to this end we provide an illustrative example. Moreover we evaluate our WS mining approach and compare it with a well known related effort. Finally we discuss the contributions and benefits introduced by the presented architecture and WS minin approach.

2 Open Reference Architecture

In this section we provide an overview of the OASIS reference architecture and we place it with respect to the end user applications and service providers. Fig. 1 illustrates the main components of the OASIS platform and shows how these interact with each other in order to achieve delivery of integrated services to the end-user through a variety of applications. The OASIS platform defines the logical platform, on which resources from different providers can be shared in an integrated way. This platform encompasses all those components that are required for the seamless integration of distributed applications in the form of WS and their provision to the end-user through an ambient intelligence (AmI) framework of agents.

As Fig. 1 shows, the different applications that belong to the user space represent the end-user applications enabled on a wide range of devices, mainly mobile ones. These applications meet the user requirements and OASIS defined use cases, covering the needs of the elderly and their caregivers in terms of Independent Living, Socialization, Autonomous Mobility and Smart Workplaces. These applications are provided to the end-users through the AmI framework of agents and it would have been impossible for them to exist unless a wide range of corresponding standalone applications were integrated into the platform through WS interfaces. These applications belong to the provider-space and their integration into the platform is enabled by an appropriate tool namely the Content Connector Module (CCM) through a mechanism that applies DM

techniques on WS data. The result of this process is the semantic alignment of services according to a set of interconnected ontologies that are known in the context of OASIS as the ”hyper-ontology”. The basic requirement for any provider-space application to be integrated, is that is should export its functionality in the form of WS. Thus, the idea behind the integration scenario is that standalone application providers can integrate (part of) their application’s functionality by constructing one or more appropriate WS. Then it is up to the CCM to integrate the application through the WS interface in a seamless way.

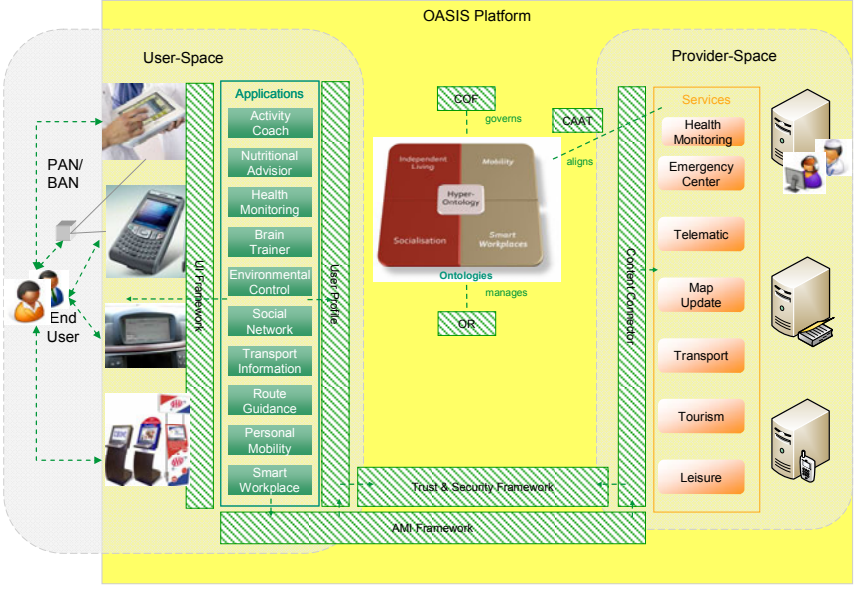


Fig. 1. The OASIS framework reference architecture encompasses all those tools and modules that are necessary for the seamless integration of distributed applications in the form of web services and their provision to the end user through an AmI framework of agents

The specific concepts involved in the OASIS architecture are explained in what follows.

- *AMI Framework*. The Ambient intelligence framework that provides seamless interactivity between OASIS services, applications and the hyper-ontology. It is comprised of the multi-agent platform that is describe in detail in Section 4.
- *Common Ontological Framework (COF)*. The COF defines a formal specification of ontology modules, and how they relate. The COF defines a methodology and best practice for ontology construction. It makes possible to define a hyper-ontology and will also facilitate and optimize the integration of new emerging ontologies. This hyper-ontology will reside in the OR (Ontology Repository) also provided by the COF.

- *Content Anchoring and Alignment Tool (CAAT)*. This tool aligns the functionality of the provided WS through its WSDL file with the ontologies stored in the OR. The concepts of the same or different application areas, after being aligned with other ontological concepts, will be able to anchor in the hyper ontology framework, thus being ready to be used seamlessly through the CCM.
- *Content Connector Module (CCM)*. The role of the CCM is twofold: it supports automatic integration of WS, which takes place when new service providers are willing to register their WS in OASIS, and it receives a request for service by the end-user (client) application via the AmI and invokes the appropriate service that returns the required content to the client. CCM applies DM techniques in order to classify the WS with respect to their ontological descriptions.
- *Ontology Repository (OR)*. It is the technological layer that supports the Ontologies storage and management. The COF (Common Ontological Framework) provides one specific repository for OASIS, the OOR.
- *Trust and Security Framework (TSF)*. The TSF is a module responsible for identification, authentication, authorization, including delegation, federa-tion between domains and the integration of the identity services.
- *UI Framework*. Allows automatic user interface self-creation for new connected services and self adaptation to the device used, the context of use and the user needs and preferences.
- *User Profile*. It contains all the context information related to a specific user. If one OASIS components needs to retrieve some information related to the user context but out of its own scope, it should make a query to this user profile.

3 The Web Service Mining Mechanism

As opposed to the most common available service classification techniques, which deal with the classification of WSDL documents into different application domains, the WS mining framework described in this section focuses on the classification of all structural elements of a WS, i.e. operations and their input and output (i/o) parameters into different classes that describe operations of the same application domain, and their parameters respectively.

A typical view of a WSDL file is illustrated in Fig. 2. Appropriate XML structures are used to annotate the various WSDL structural elements. Amongst these, a WSDL file uses the *operation* XML element in order to annotate WS operations, while the *input, output* elements are used for the definition of the corresponding i/o operations. A WSDL file also includes other elements in order to specify the invocation details of its operations.

Thus, any WSDL description represents a WS with the following hierarchy:

- The top level of the hierarchy is represented by the WS that is structured as a collection of operations and potential data type definitions.
- The WS operations that represent methods or functions that are invoked by the WS clients belong to the next hierarchical level.
- The bottom-most level includes the input and output parameters of the WS operations.

With respect to the aforementioned analysis of the basic hierarchical levels that represent the internal WSDL structure, we introduce a semantic categorization schema composed of the following three layers.

1. A *first layer* whose goal is to semantically categorize the overall WS into one application domain.
2. A *second layer* to categorize the WS operations into a set of "ideal" operations defined in the ontology.
3. A *third layer* that concerns categorization of input and output operation parameters into ontological concepts.

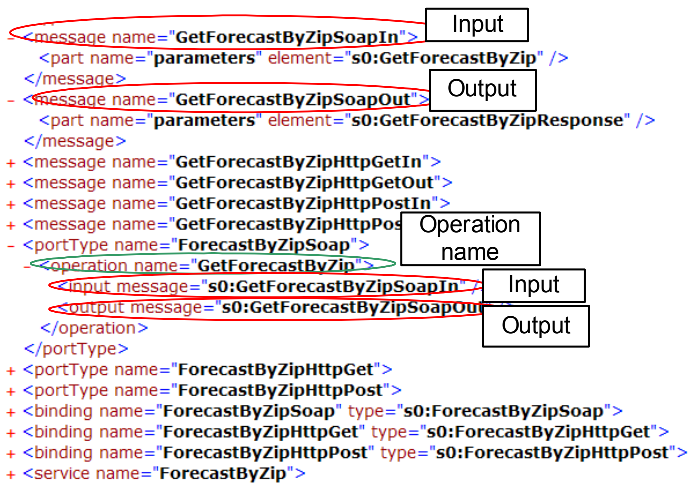


Fig. 2. A typical WSDL file. In this example we show the *GetForecastByZip* operation that includes the *GetForecastByZipSoapIn* input and *GetForecastByZipSoapOut* output parameter. The details of these parameters (e.g. their data types) are defined under the message element.

In the hyper-ontology appropriate associations between ontological concepts are defined in the form of object properties to sufficiently annotate real WS and their structural elements. For example, the *hasInput*, object property is defined in order to associate a WS operation with its input parameters.

The categorization of a WS in a given domain, which is undertaken on the first layer, implies that all operations that are included in the WSDL document belong to the same domain as well. In the second categorization layer, any WSDL document that represents a WS is analyzed into its operations. The purpose of automatic classification of operations is to allow automated annotation of operations apart from WS classification and thus facilitate more automated ways of service invocation. After describing the details of the categorization mechanism, we present the results derived by its application to a set of real WS data.

In the first layer, the semantic categorization mechanism aims to accurately determine the application domain to which an arbitrary WS belongs. The problem of WS

categorization is formally described as follows. Let $A = a_1, \ldots, a_m$ be a vector of m WS and $D = d_1, \ldots, d_n$ a vector of n application domains. The categorization mechanism aims to develop an accurate classification model based on the collection A, capable of predicting the application domain d_i to which any arbitrary WS a_j belongs. The result of the categorization K can be expressed in algebraic form:

$$K = A \cdot D \tag{1}$$

where

$$k_{ij} = \begin{cases} 1 & \text{if } a_i \text{ belongs to } d_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

We impose the following restriction to equations (1) and (2). For each $i \leq m$ there is exactly one $j \leq n$, such that: $k_{ij} \neq 0$.

Matrix K is the result of a composite process, consisting of five stages, represented as boxes in the diagram illustrated in Fig. 3. Each stage corresponds to the required actions for the deployment of an accurate WS classification model.

In the first stage, a collection of WS already categorized to a set of application domains are used as input to the process. In this stage each WSDL document is parsed and WS-specific data are extracted.

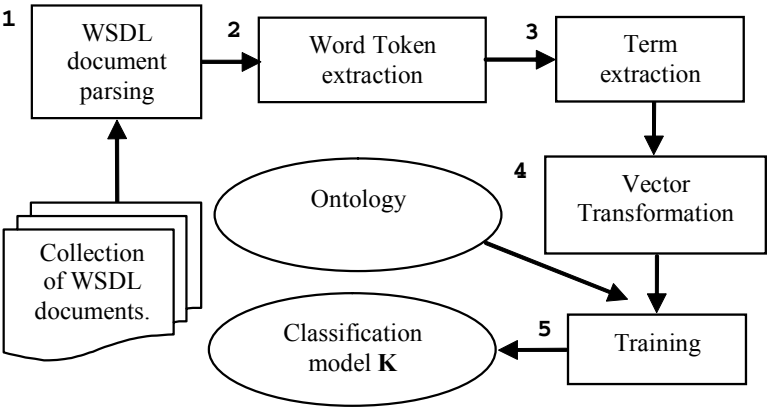


Fig. 3. The creation of the classification model is a five stage process: The WSDL documents are parsed (1) and word tokens (2) and terms (3) are extracted from them. These are transformed into numerical vectors (4) which are used for training (5) in combination with semantic descriptions of terms in the ontology.

The extracted data pass to the next stage where the word tokens extraction is performed. Word tokens extraction involves splitting data into distinct words. Data splitting is performed in different ways according to the specific part of the WSDL that provides the data. If, for example data to be split correspond to the operations names and their i/o parameters, the commonest naming conventions adopted by the developers are taken into account. Generally, a valid operation name includes strings written in

camel case or strings using the underscore character '_' to concatenate separate words. Considering this, we process the operation name and i/o parameter names and extract distinct words using string manipulation functions. For example the following operation names *getCountryCodes*, *get_company_profile* and *get_Weather_byZIP* result in the following word token lists [*get, Country, Codes*], [*get, company, profile*] and [*get, Weather, by, ZIP*], respectively. Moreover, we process documentation tags including comments written by developers, where potential grammatical errors are taken into account.

The third stage performs filtering of all extracted word tokens in order to determine the "uniqueness" of each token and its importance regarding the information it encapsulates. For this purpose a stop-list of words is used. This list contains articles, prepositions, and generally words that appear frequently in each operation name (e.g. the words *get, set, by*) or HTML tags, i.e., words that do not provide rich information. Generally speaking, removing stop words is considered to be a useful technique for filtering non-relevant terms [9]. In addition, our categorisation technique discovers all the synonym words by the use of the WordNet [3] dictionary, and performs word stemming, by adopting the Porter stemming algorithm [10].

The actions that take place in stage 3 result in a set of terms that are provided as in-put to stage 4, where each WS is transformed into a term vector of the form:

$$v = [w_0, w_1, \ldots, w_n, c] \tag{3}$$

Each vector element w_i represents a weight associated with the importance of term t_i, while the last element c refers to the category, to which the WS belongs (the application domain in the first categorization layer).

Term-weighting schemes have been widely used in information retrieval and text categorization [11]. Among the most important and widely used ones, we recognize the term frequency (*tf*) and term frequency-invert document frequency (*tf-idf*). The *tf* simply corresponds to the frequency of occurrence of a term within a document. The *tf-idf* is defined as $tf \cdot log(N/n_t)$, where N corresponds to the total number of documents in the collection and n_t is the number of documents containing the term t.

In stage 5, all produced vectors are provided as training data to a learning classification algorithm. Learning algorithms learn patterns from data and generate a classification model for predicting the class attribute for new unknown instances. Several algorithms such as the Naïve Bayes can be used for the generation of an accurate classification model. In this stage the classes that represent the application domain are derived from an ontology formatted in OWL (Web Ontology Language) [4].

For the implementation of the second categorization layer, whose purpose is the categorization of WS operations into their semantically described counterparts in the ontology, we re-use the same process adopted in the first layer. In order for the first layer mechanism to be applicable in the second layer as well, the semantics of equations (1) and (2) had to be re-defined as follows.

Vector A in equation (1) represents the vector of operations included in all WS that belong to the same application domain (specified in the previous categorization layer). Furthermore, vector D includes all "ideal" operations, i.e., WS operation descriptions

[3] `http://wordnet.princeton.edu`
[4] `http://www.w3.org/TR/owl2-overview/`

defined in the ontology. Thus in equation (2) it holds that $k_{ij} = 1$ if WS operation a_i belongs to "ideal" operation d_j. Finally, parameter c in equation (3) represents the "ideal" operation to which a real WS operation is categorized. We also assume that in the first three stages of the categorization mechanism depicted in 3, for each operation only tokens belonging to the operation name, as well as its i/o parameters, are used.

In the third categorization layer i/o parameters are classified into their ontology counterparts (i.e. concepts). Since the third layer does not contain adequate information, as opposed to the previous two layers, a different classification method was required. Categorization in this layer is dealt with a composite scheme for matching i/o of a WS operations with operation i/o defined in the ontology. The algorithm used consists of three levels of matching, lexicographic, structure and data type matching.

At the first level of this comparison each tuple of i/o names is tokenized, and tokens are placed in a bipartite graph. For every tuple a (normalized to 1) score is assigned, using *WordNet:Similarity*, cf. [12], and *n-grams* model, according to their semantic and lexicographic similarity. The resulting score matrix S consists of all score combinations between tokens. The assignment problem of finding the best matches of tokens (represented with matrix S) is solved by the Kuhn-Munkres algorithm (also known as the Hungarian algorithm) in polynomial time, cf. [13]. Then, the score of the comparison between two inputs or two outputs is estimated by the average number of scores of best assignments.

4 The Multi-Agent System

In the context of the OASIS platform, the AmI framework is implemented as a MAS that provides a pervasive software infrastructure to support the interaction between the user and the OASIS reference architecture, through self-adaptation of the interface layer to the type of device used (PC, tablet PC, PDA, smart phones), the specific service layout, the context of use and the elderly user personal needs and preferences.

OASIS adopts a rather light multi-agent architecture of flexible and efficient agents, to ensure a high performance AmI framework. OASIS agents undertake the process of low-level information filtering using real-time data. Agents of the OASIS framework act in a deliberative function. From a technical perspective this is realized by the development of specific soft computing techniques for learning and decision-making, such as Bayesian belief networks, and the adoption of rational decision making abstract models, such as LORA [14]. These approaches empower agents with operations which are fundamental for agents to be considered intelligent from an Artificial Intelligence point of view.

The multi-agent community that implements the AmI framework is illustrated in Fig. 4. The interaction between AmI agents and other OASIS modules is shown at the same figure.

The main types of agents that are identified as critical in order for the AmI to properly operate are the following:

- *Content Connector Agents*. These agents are responsible for interacting with the Hyper-Ontology through the Content Connector Module and obtain the information required to satisfy the user requests.

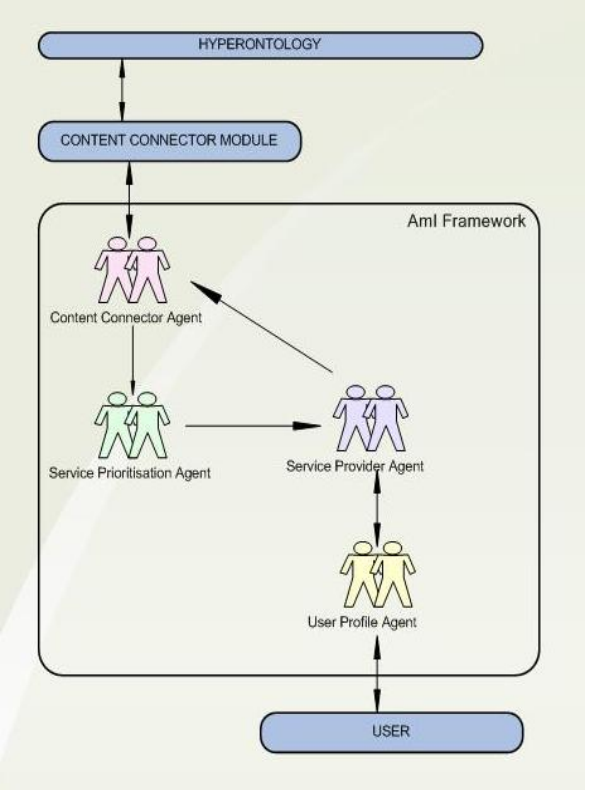


Fig. 4. The OASIS ambient intelligence framework is implemented a multi-agent system. The diagram shows the different agent types and the ways the interoperate with each other.

- *Service Prioritization Agents*. They interact with the User Profile Agents in order to assist them e.g. in filtering the returned results to the user. They have to do with contextual data such as user profile, preferences, history, physical user environment, etc., and their functionality depends on the user profile agents' responsibilities.
- *Service Provider Agents*. The task of the Service Provider Agents is to receive information from the Service Prioritization Agents and orchestrate the invocation of the services that best suit user needs, based on the information stored in the User Profile Repository and on the output of the Service Prioritization Agents. During this procedure, the Service Provider agents return the required information back to the user.
- *User Profile Agents*. All interactions between the AMI framework and the users are implemented through a set of personal assistant agents, known in the OASIS framework as User Profile agents.

The way in which the MAS presented in the previous subsection interacts with the WS mining mechanism presented in Section 3 is described as follows. WS mining results in a knowledge base of mappings between real WS and the corresponding ontological

concepts that describe all existing WS. According to the three- layer classification schema, each one of the WS construct element is mapped to the corresponding ontological counterpart i.e., operation and i/o. This knowledge base is part of the hyper-ontology and it is accessed by the Content Connector agent that responds to incoming requests for available services. As soon as the Service Provider agent requests appropriate WS to fulfill the needs of a particular use case executed by the end-user, it interacts with the knowledge base (through the Content Connector agent) and receives knowledge from the shared hyper-ontology. It then uses this knowledge in order to retrieve the appropriate WS operations and finally posts a re-quest for the invocation of the operation that best suits user preferences according to a set of personalization criteria.

The actual interaction between the MAS and the WS mining module is undertaken by the Content Connector Agents. Their role is to interact with the CCM and initiate the semantic search procedure based on the hyper-ontology through an interface. They exploit user-related information and execute rational logic in order to discover the services that best match the user needs. They do this by receiving combined information from User agents and Service prioritization agents. Finally, they perform all required actions in order to forward the user request to the CCM and the hyper-ontology and receive the appropriate results in an agent understandable format.

4.1 An Example of Use

The following example shown in Fig. 5 illustrates the workflow that takes place from the MAS point of view starting from the service registration and ending up to the service invocation process. Let us assume that the end user is equipped with a mobile device on which the end user application runs. Through this, the user selects the route guidance option that requires the invocation of routing web service and more precisely the invocation of a web service operation that receives as input the coordinates of two points A and B and returns as an output a route from point A to B as a list of connected route segments commonly displayed on a map. From the moment at which the user selects the routing operation, the following sequence of actions take place (numbered as on Fig. 5) before the required information is displayed on the user's device.

1. The device sends to the CCM a request for the invocation of an appropriate operation that returns routes on maps.
2. CCM performs a semantic search in the hyperontology in order to find all aligned operations that correspond to route guidance. This category of operations are described in the hyperontology as an ideal operarion called *getRoute*. CCM returns a ranked list of two operations that semantically belong to the *getRoute* ideal operation and send this list to the MAS. The top-ranked operation is considered to be the most similar to the one requested by the application.
3. The MAS performs a selection of the operation that best matches personal user preferences. This operation is transparent to the user. Alternatively, the list of operations that is being ranked by the agents could be visible to the users in order to allow them select the one they believe best suits their preferences. The application displays a form that prompts users enter information required for web service invocation.

4. CCM receives user input from the GUI, as well as the operation that the MAS agent-user has selected. It then dynamically creates a wrapper for the invocation of the selected operations and calls the corresponding web service that in turn returns its output to the CCM. After that, CCM maps the returned output to a commonly understood concept defined in the hyper-ontology and returns the appropriate content to the MAS agents.
5. The MAS receives the requested content from CCM and displays it on the user device.

5 Evaluation of the Web Service Classification Accuracy

For the evaluation of the WS classification mechanism that is introduced in this paper we setup a set of experiments whose goal is to assess the impact of various parameters involved in the different stages of the overall process that is illustrated in Fig. 3.

For our experiments a set of WS was collected from publicly available sources, such as the public WS repositories: WebServiceX.NET[5], XMethods[6], the seekda! web service search engine[7] and random publicly available web services. The data set contained 297 WSDL documents which where manually categorized to five categories. These are: (1) Business and Money, (2) Geographic services, (3) Communication, (4) Converters and (5) Tourism and Leisure. The distribution of the web services collection in each one of the five categories is shown in Table 1.

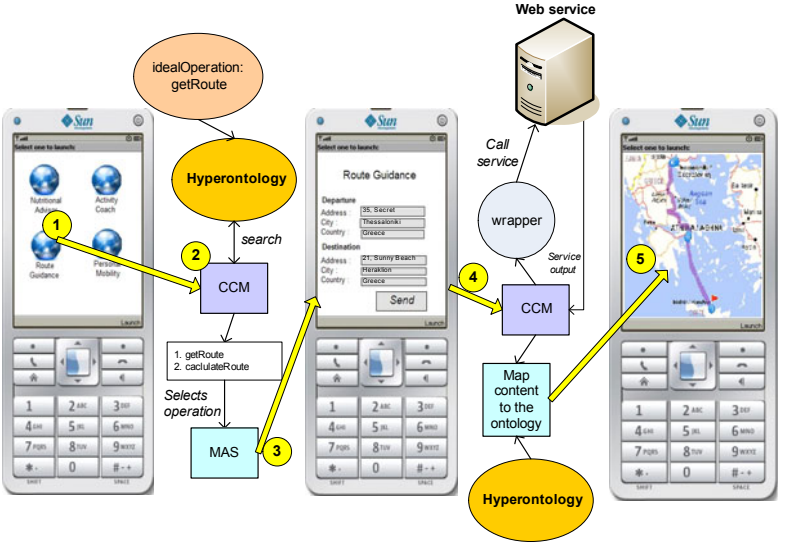


Fig. 5. An example that illustrates the sequence of actions required by the service invocation process

[5] http://www.webservicex.net
[6] http://www.xmethods.com
[7] http://webservices.seekda.com

Table 1. Input data used for our evaluation process

Domain	Number of web services
Business and Money	99
Geographic	79
Communication	69
Converters	29
Tourism and Leisure	21
Total	297

The experiments were conducted subsequently, starting from a subset of the training set and gradually increasing until the 100% of the training set becomes available. More specifically we started the experimentation process by selecting only the two domains with the most WS available, i.e., *Business and Money* and *Geografic*. We proceeded by selecting the rest of the classes according to the number of WS instances they contained. This gradual selection enabled us to determine how each parameter influences accuracy as the number of classes and consequently the number of WS increases.

A k-fold validation process [15] was adopted in all of our experiments. According to this method, the dataset is divided into k subsets. For each iteration, one of the k subsets is used as the test set, while the remaining $k-1$ subsets form the training set. In our experiments we have used 10 as a value for k.

In the first set of experiments we varied the value of the parsing level parameter while keeping the rest of the parameters constant. Fig. 6 presents the classification accuracy of our WS mining mechanism for various values of the parsing level parameter. This parameter expresses the level of information that is taken into account in WSDL document parsing and term extraction (stages 1-3 of our classification model depicted in Fig. 3) when we used web services from 2, 3, 4, and 5 domains, respectively. It takes three values: operations (only the WSDL operation names are taken into account), input-output (only the WSDL input-output parameter names are taken into account) and operations/input-output (all of the above).

In our next experiment our goal is to assess the performance of the classification mechanism when different classification algorithms are applied. We consequently applied the following algorithms: Naïve Bayes [16], support vector machines (SVM) [17], Naïve Bayes Multinomial [18], Discriminative Multinomial Naïve Bayes classifier (BDMNText) [19] and the PART algorithm, which is a rule-based classifier [20]. The results are shown in Figure 7, where the horizontal axis represents the classification accuracy for each one of the classification algorithms shown on the vertical axis. The number of domains used for classification of WS was set to 5, i.e. all available WS were involved in this experiment.

Concluding the evaluation process, as it is seen on Fig. 6 and 7 the accuracy of the classification mechanism reaches up to the level of 76% when the Naïve Bayes Multinomial or the Discriminative Multinomial Naïve Bayes classifiers are applied. The overall performance is also boosted when the operation names, as well as the input and output parameters are taken into account at the WSDL parsing stage.

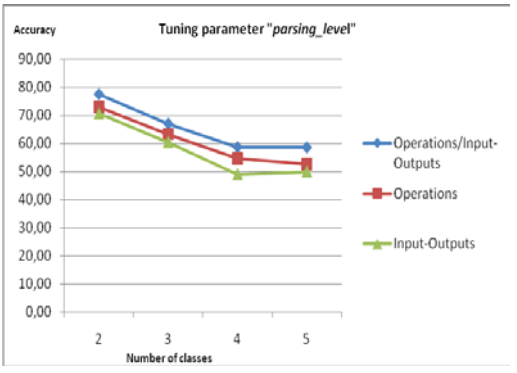


Fig. 6. Classification accuracy of web services into various numbers of different domains (2-5) for different levels of information that is taken into account at the WSDL document parsing and word token extraction stages

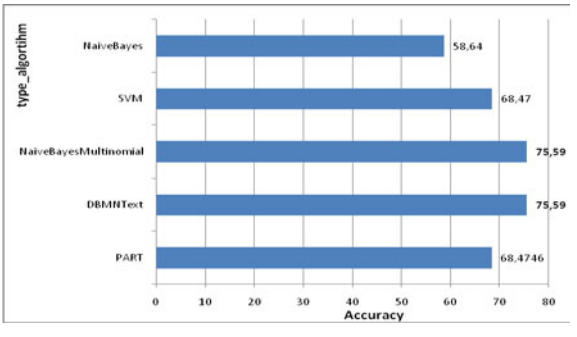


Fig. 7. Classification accuracy of web services into five different domains when various classification algorithms are applied

Last but not least we conducted comparison of our classification model to ASSAM [6] that is the most representative and accurate web service classification tool we identified so far. ASSAM has been evaluated using a repository of 391 WS divided into 11 categories. A leave-one-out cross validation methodology was adopted.

Fig. 8 presents the results of the experimental comparison. A tolerance value t was used for determining the classification accuracy while allowing near misses. Thus, the correct category is included in a sequence of $t + 1$ suggestions. For example, the tolerance value $t = 1$ implies that we allowed the classifier to fail only once. The horizontal axis in Fig. 8 corresponds to the tolerance threshold t, while the vertical axis renders classification accuracy. Our mechanism outperforms ASSAM for $t = 0$, $t = 1$. This implies that our mechanism predicts more accurately the right category of a WS when this appears in the first two suggestions. When $t = 2$, i.e., the right category is the third suggestion, both approaches achieve almost the same accuracy.

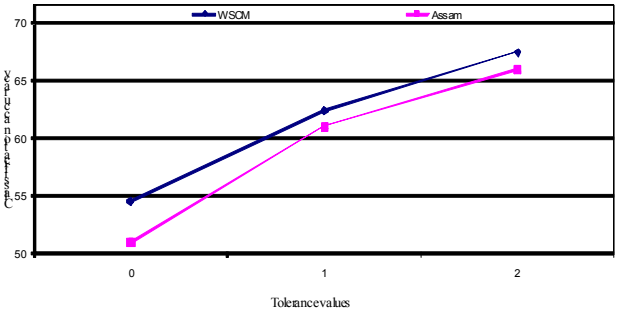


Fig. 8. Comparison of our web service classification mechanism to ASSAM. Our approach results in better classification performance.

6 Summary and Conclusions

In this paper we have presented the implementation of an open reference architecture that allows existing applications to be integrated in a common platform that provides end-user applications through a multi-agent framework. The knowledge about WS that the main ontology encompasses is generated by the application of DM techniques on WS data. We argued that the existing WS mining applications are mainly used to extract useful knowledge from services in order to optimize service-oriented operations. None of the existing approaches apply WS mining to be used by agents for enabling the provision of advanced functionalities and services to the users they represent. In contrast, our approach applies WS mining for the semantic categorization of services to be used by service discovery and prioritization agents in order to provide more personalized end-user facilities.

Regarding the classification accuracy of our proposed WS mining model we conducted a set of experiments whose purpose was to evaluate our approach in different configurations of the involved parameters. Finally a comparison of our approach to ASSAM, one of the most accurate WS classification tools, shows that our approach performs marginally better after fine tuning some of the involved parameters. These results seem to be quite encouraging but more effort is needed for the improvement of the overall classification accuracy. Our future work involves a large scale implementation of the reference architecture and the extension of the proposed mechanism in order to support other types of WS (e.g. Restful [21] web services).

References

1. Zheng, G., Bouguettaya, A.: Service Mining on the Web. IEEE Transactions on Services Computing 2(1), 65–78 (2009)
2. Asbagh, M.J., Abolhassani, H.: Web service usage mining: mining for executable sequences. In: Revetria, R., Cecchi, A., Schenone, M., Mladenov, V.M., Zemliak, A. (eds.) Proceedings of the 7th Conference on 7th WSEAS International Conference on Applied Computer Science, vol. 7, pp. 266–271. World Scientific and Engineering Academy and Society (WSEAS), Stevens Point (2007)

3. MuthuMeena, M., Jayakumar, S.K.V., Dhavachelvan, P.: Service Mining Architecture for Web-based Services using Agents. International Journal of Computer Science And Applications 1(1), 56–59 (2008)
4. Patil, A.A., Oundhakar, A., Sheth, A.P., Verma, K.: METEOR-S Web service annotation framework. In: Proc. of the 13th International Conference on WWW. ACM Press, New York (2004)
5. Oldham, N., Thomas, C., Sheth, A., Verma, K.: METEOR-S Web Service Annotation Framework with Machine Learning Classification. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 137–146. Springer, Heidelberg (2005)
6. Heß, A., Kushmerick, N.: Learning to attach semantic metadata to Web services. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 258–273. Springer, Heidelberg (2003)
7. Corella, M.A., Castells, P.: Semi-automatic semantic-based Web service classification. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 459–470. Springer, Heidelberg (2006)
8. Crasso, M., Zunino, A., Campo, M.: Awsc: An approach to Web service classification based on machine learning techniques. Inteligencia Artificial, Revista Iberoamericana de IA 12(37), 25–36 (2008)
9. Feldman, R., Sanger, J.: The Text Mining Handbook: Advanced Approaches in Analyzing unstructured Data. Cambridge University Press, Cambridge (2006)
10. Porter, M.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
11. Lan, M., Tan, C.-L., Low, H.-W., Sung, S.-Y.: A comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In: Special Interest Tracks and Posters of 14th International Conference on WWW (2005)
12. Pedersen, T., Patwardhan, S., Michelizzi, J.: Word Net: Similarity - Measuring the Relatedness of Concepts. In: Proceedings of Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-04), Boston, MA, May 3-5, pp. 38–41 (2004)
13. Burkard, R.E., Dell'Amico, M., Martello, S.: Assignment Problems. SIAM, Philadelphia (2009)
14. Wooldridge, M.: Reasoning about rational agents. MIT Press, Cambridge (2009)
15. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, vol. 2(12), pp. 1137–1143 (1995)
16. John, G.H., Langley, P.: Estimating Continuous Distributions in Bayesian Classifiers. In: Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345. Morgan Kaufmann Publishers, San Mateo (1995)
17. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation 13(3), 637–649 (2001)
18. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classification. In: AAAI-98 Workshop on Learning for Text Categorization (1998)
19. Su, J., Zhang, H., Ling, C.X., Matwin, S.: Discriminative Parameter Learning for Bayesian Networks. In: International Conference on Machine Learning, ICML (2008)
20. Frank, E., Witten, I.H.: Generating Accurate Rule Sets Without Global Optimization. In: Fifteenth International Conference on Machine Learning, pp. 144–151 (1998)
21. Pautasso, C., Zimmermann, O., Leymann, F.: Restful web services vs. "big"' web services: making the right architectural decision. In: Proceeding of the 17th International Conference on World Wide Web, WWW '08, Beijing, China, April 21-25, pp. 805–814. ACM, New York (2008)

Minority Game Data Mining for Stock Market Predictions

Ying Ma, Guanyi Li, Yingsai Dong, and Zengchang Qin*

Intelligent Computing and Machine Learning Lab
School of Automation Science and Electrical Engineering
Beihang University, Beijing, 100191, China
zcqin@buaa.edu.cn

Abstract. The Minority Game (MG) is a simple model for understanding collective behavior of agents in an idealized situation for a finite resource. It has been regarded as an interesting complex dynamical disordered system from a statistical mechanics point of view. In previous work, we have investigated the problem of learning the agent behaviors in the minority game by assuming the existence of one "intelligent agent" who can learn from other agent behaviors. In this paper, we propose a framework called Minority Game Data Mining (MGDM), that assumes the collective data are generated from combining the behaviors of variant groups of agents following the minority games. We then apply this framework to time-series data analysis in the real-world. We test on a few stocks from the Chinese market and the US Dollar-RMB exchange rate. The experimental results suggest that the winning rate of the new model is statistically better than a random walk.

1 Introduction

An economic market is regarded as a complex adaptive system (CAS) by the physicists and computer scientists from the econophysics [10] community, which is an interdisciplinary research field that applies theories and methods originally developed by physicists in order to solve the problems in economics, Agent-based models of complex adaptive systems provide invaluable insight into the highly non-trivial collective behavior of a population of competing agents, this has been used in experimental economics [15,5] financial market modeling [9,8] and market mechanism designs [11].

Agent-based experimental games have attracted much attention among scientists in different research areas [10], e.g., in financial modeling people are trying to solve the problem of analyzing the real market system where involving agents with similar capability are competing for a limited resource. One of the most important issues is: in such a complex system, every agent knows the history data in the market and must decide how to trade based on this global information. Among these agent-based models, minority game (MG) [4] is an important model in which an odd number N of agents successively compete to be in the minority side. This model can be regarded as a simplified version of EI Farol bar problem [1], in which a number of people decide weekly

* Corresponding author.

L. Cao et al. (Eds.): ADMI 2010, LNCS 5980, pp. 178–189, 2010.

whether go to the EI Farol bar to enjoy live music in the risk of staying in a crowd place or stay at home. As a new tool for learning complex adaptive systems, the minority game has been applied to variety areas [8,9] especially in financial market modeling [2,3]. But this model has limitations that agents are always assumed to be identical and employing similar strategies, which is an unrealistic assumption.

In real-life scenarios, some agents make random decisions and some groups employ similar strategies. The complexity of marketing world is embodied in existence of varieties types of agents using strategies based on their own rules. Also it is unrealistic to have the whole population following the same rules of a game. There could be some more intelligent ones who are standing out by learning from the others.

In this paper, we propose a model that efficiently figures out limitations of previous models when apply to the real markets. Through observing the dynamics, we increase complexity of this model by divided the real market into several diverse types of agents. Using a simple machine learning algorithm, an intelligent agents can analyze and estimate the real dynamic markets to maximize own profits in terms of winning probability. It is a new way to understand the relationship of micro-behaviors and macro-behaviors by utilizing minority game model and machine learning. This paper is organized as the following: section 2 introduces the minority game and we propose learning methods of using decision tree and genetic algorithm for discovering the composition of agents in order to predict the macro-behavior of the MG. In section 3, a series of experimental results are presented using real-world data from stock price and currency exchange rate. We verify the effectiveness of this learning mechanism and conclusions are given in the end.

2 Learning in the Minority Games

EI Farol bar problem proposed by Arthur [1] is one of the best-known experimental game models in economics, in which a number of people decide weekly whether go to the El Farol bar to enjoy live music in the risk of staying in a crowd place or stay at home. Formally: N people decide independently to go or stay each week, they have two actions: go if they expect the attendance to be less than an integer $[\alpha N]$ (where $0 < \alpha < 1$), otherwise, stay at home if they expect it will be overcrowded. There is no collusion or prior communication among the agents; the only information available is the numbers who came in past weeks. Note that there is no deductively rational solutions to this problem, since only the numbers attending in the recent past are given.

One simplified version of the El Farol Bar problem is the Minority Game proposed by Zhang and Challet [4]. In the minority game, there is an odd number of players and each must choose one of two choices independently at each round. The players who end up on the minority side win. The minority game examines the characteristic of the game that no single deterministic strategy may be adopted by all participants in equilibrium [16]. In this section, we will set an environment that is populated by a few diverse groups of agents. In the next section, the mathematical treatments of the MG will be introduced in details.

2.1 Strategies of Agents

A population of N (an odd number) agents decide between two possible options, say to attend room A or B at each round of the game. Formally, at the round t ($t = 1, \ldots, T$): each agent need to take an action $a_i(t)$ for $i = 1, \cdots, N$, to choose room A or B.

$$a_i(t) = \begin{cases} A & \text{Agent } i \text{ choose room } A \\ B & \text{Agent } i \text{ choose room } B \end{cases} \tag{1}$$

In each round of the game, the agents belonging to the minority group win. The winning outcomes can be considered as a binary function $w(t)$. Without loss of generality (w.l.o.g), if A is the minority side, we define that the winning outcome is 0, otherwise, it is 1. In this paper, the winning outcomes are known to public.

$$w(t) = \begin{cases} 0 & \#(a_i(t) = A)|_{i=1,\ldots,N} \leq (N-1)/2 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

where $\#()$ is the counting function: for all the agents (i runs from 1 to N), if the number of agents that satisfy the condition $a_i(t) = A$ is less than or equal to $(N-1)/2$, then $w(t) = 0$; otherwise, it is $w(t) = 1$. We assume that agents make choices based on the most recent m winning outcomes $h(t)$, which is called the *memory* and m is the *length of memory*.

$$h(t) = [w(t-m), \ldots, w(t-2), w(t-1)] \tag{3}$$

Given the outcome $w(t)$ at the moment t, agent i keeps a record $r_i(t)$ that tells whether it has won the game or not.

$$r_i(t) = \begin{cases} T & \text{Agent } i \text{ wins at time } t \\ F & \text{Agent } i \text{ loses at time } t \end{cases} \tag{4}$$

In minority game, we usually assume that each agent's reaction based on the previous data is governed by a "strategy" [2,3,4]. Each strategy is based on the past m-bit memory which are described as a binary sequence. Every possible m-bit memory are mapped corresponding to a prediction of choosing room A or B in the next round. Therefore, there are 2^{2^m} possible strategies in the strategy space. Agents in the same fixed strategy group share one strategy randomly selected from the strategy space. Given the memory $h(t)$, the choice for the agent i guided by the strategy S is denoted by $S(h(t))$. Table 1 shows one possible strategy with $m = 3$. For example, $h(t) = [000]$ represents that if the agent who choose A in the latest three steps win, the next round (at round t) choice for this agent will be $S([000]) = A$. A strategy can be regarded as a particular set of decisions on the permutations of previous winning outcomes.

Figure 1 shows that giving the history data of all winning outcomes, training data for an agent can be sampled by a sliding window with size m on each time step. At t round, the target value (either A or B) is the agent's actual choice at the current round. Therefore, training set for agent i can be formally defined as:

$$\mathcal{D}_i = \{(h(t), a_i(t))\} \qquad for \quad t = m+1, \ldots, T \tag{5}$$

Based on this training set, we hope to find an effective algorithm to learn other agents' strategies.

Table 1. One possible strategy with $m = 3$: the current choice of agent is decided by its previous 3 step memory

$h(t)$	000	001	010	011	100	101	110	111
$S_1(h(t))$	A	A	B	B	B	A	B	A

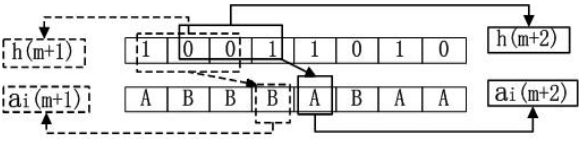


Fig. 1. Training data $\mathcal{D}$ is obtained from a sliding window of size m

2.2 Decision Tree Learning

Decision tree learning is one of the simplest and most effective learning algorithms. It has been widely used in numerous machine applications for its simplicity and interpretability [13] (A decision tree can be decomposed into a set of rules). In this paper, the tree model we use is actually probability estimation tree [12] because we are considering the probability distribution over two possible choices of A and B.

In decision tree learning, for each branch $W = [w(t-m), \ldots, w(t-2), w(t-1)]$, there is an associated probability distribution on possible agent's choices (i.e., A or B) that is calculated based on the proportion of these two kinds of data falling through the branch. Or formally:

$$P(A|W) = \left. \frac{\#(h(t) = W \wedge a_i(t) = A)}{\#(h(t) = W)} \right|_{(h(t),a_i(t)) \in \mathcal{D}_i} \tag{6}$$

where $\#()$ is the same counting function appeared in eq.2, therefore, eq.6 means that: given a branch, it looks for all matching strings from the training data and checks how many of them chose A. If agent i follows a particular fixed strategy and game is repeated for n ($n \gg 0$) rounds. We then can get a fair good estimation of the behaviors of agent i given the 3-step memory training data.

For each agent i, its current choice $a_i(t)$ can be predicted by the decision tree learning based on the training data $\mathcal{D}_i$(see eq. 5). At t round of the game, the probability that the intelligent agent choose A, $P_I(A)$, is calculated based on its estimation of other agents' choices $a_i(t)$ where $i = 1, \ldots, N$.

$$P_I(A) = 1 - \left. \frac{\#(a_i(t) = A)}{\#(a_i(t) = A) + \#(a_i(t) = B)} \right|_{i=1,\ldots,N} \tag{7}$$

and $P_I(B) = 1 - P_I(A)$. The above equation can be interpreted that the intelligent agent will go to the room that most of agents won't go based on its predictions of other agents' behaviors. Simply, the intelligent agent makes choice based on its estimation $P_I(A)$ and $P_I(B)$.

$$a_I(t) = \begin{cases} A & P_I(A) > P_I(B) \\ B & \text{otherwise} \end{cases} \tag{8}$$

The accuracy of wining for the intelligent agent I can be obtained by:

$$AC_I(t) = \frac{\#(r_I(t) = T)}{\#(r_I(t) = T) + \#(r_I(t) = F)}\Bigg|_{t=1,\ldots,T} \tag{9}$$

This estimation is based on *complete information*, i.e., all records of agents' choices and history outcomes are known to public, Our previous work [7] has shown the experimental results in diverse environments. These results indicate intelligent agent can outperform other agents using this simple learning algorithm.

2.3 Genetic Algorithm for Minority Game Data Mining

In the previous section, we designed an intelligent agent that uses machine learning method to learn the patterns of other agents in the MG. However, in most cases, we can only obtain the collective data $w(t)$. In this section, we propose a framework that assumes the macro-behavior can be decomposed into several groups of agents, within each group, agents employ similar strategies. A *Genetic Algorithm* [6,11] is used to estimate the parameters of this decomposition.

Genetic Algorithms (GA), developed by Holland [6], is a fast and intelligent way to search one suitable parameter in large space of parameters. Possible solutions are coded as *chromosomes* that evolves through generations based on a *fitness function* which evaluates the quality of the solutions. Chromosomes with higher fitness are more likely to be selected to be used to reproduce off-springs. New generations of chromosomes are reproduced by two genetic operators: *crossover* and *mutation* [11]. Successive generations are created in the same way until ultimately the final population satisfies certain criterion or reach a specified number of runs.

We use a vector of parameters to represent the number of agents in each group and the strategy they use, a GA is used to optimize these parameters in order to find the most likely combinations of single behaviors that could generate a certain macro-level sequence. After getting the most likely compositions, intelligent agent can make decision by using decision trees.

In our model, intelligent agent only use the information of winning outcomes $w(t)$ and a guessed maximum number of groups using fixed strategies K, such that the agents can be divided into $K+1$ groups: $\{G_R, G_1, \ldots, G_K\}$, where group G_R is the group of random agents and G_k for $k = 1, \ldots, K$ employs the strategy S_k. A chromosome $\mathbf{x}$ is consisted by the the following parameters: percentage of random agents P_R (among all agents), corresponding percentage of agents with one certain fixed strategy P_{S_k} and the strategy S_k.

$$\mathbf{x} = \{P_R, P_{S_1}, S_1, \ldots, P_{S_K}, S_K\}$$

Figure 2 illustrates that, the collective data is a combination of choices from a diverse group of agents. Based on the give history sequence $w(t)$, intelligent agent use GA to explore all possible combinations of subgroups in order to find original compositions of markets. Intelligent agent then uses the information to make predictions and be in the minority side.

The fitness function is calculated as the following: at t round of the game, in order to evaluate one chromosome $\mathbf{x}_j$ ($j = 1, \ldots, J$ where J is the population size in the GA),

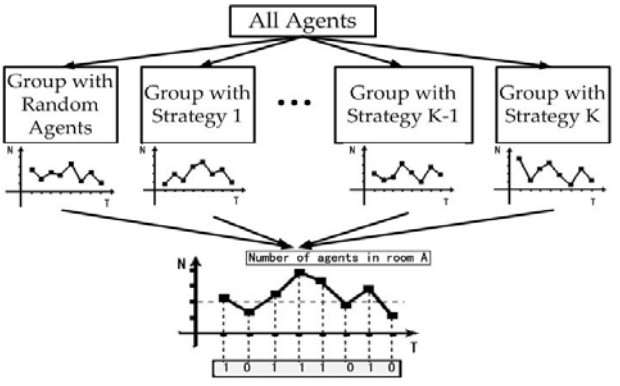


Fig. 2. The process of generating collective data. The whole system can be divided into $K+1$ groups. At each round, the collective data is considered as an aggregation of actions of every group of agents. When collective data indicates A is the minority side, the sequence of winning outcomes will be 0, otherwise, it will be 1.

we run the MG with the parameter setting given by $\mathbf{x}_j$ to obtain the history of winning outcomes $y_j(t)$. The fitness function is defined based on the comparisons between $y_j(t)$ and the actual sequence of winning outcomes $w(t)$: for t runs from 1 to a specified T, once $y_j(t) = w(t)$, we add one point to $f(\mathbf{x}_j)$, formally:

$$f(\mathbf{x}_j(t)) \leftarrow \begin{cases} f(\mathbf{x}_j(t)) + 1 & \text{if: } y_j(t) = w(t) \\ f(\mathbf{x}_j(t)) & \text{otherwise} \end{cases} \tag{10}$$

At each round t, the best chromosome $\mathbf{x}_j^*$ is selected from the pool:

$$\mathbf{x}^*(t) = \arg\max_j f(\mathbf{x}_j(t)) \quad for \quad j = 1, \ldots, J$$

The best chromosome $\mathbf{x}^*(t)$ gives the parameters can be interpreted into a combination of the subgroups. The parameters can give the best possible complete information of a MG so that the intelligent agent can learn with decision trees discussed in section 2.

3 Experiments on Real Markets

This learning process points us a new way of using the minority game model and evolutionary optimization in understanding the relationship between micro-data and macro-data. The collective behaviors of MGs can be decomposed into several micro-behaviors from different group of agents. Combining the behaviors of these groups may generate complex and seemingly unpredictable macro-behaviors. Given a sequence of history wining outcomes, by using genetic algorithm, we can find the most likely combinations of single behaviors that could generate this sequence. Then, the intelligent agent use the decision tree learning algorithm to learn the behaviors of other agents and make prediction based on it.

Many real-world complex phenomena are related to the minority game [8,9,4]. We consider the stock market and the currency exchange rate in this research. Although the macro-level data are seemingly random and unpredictable, we could build models with the MGDM framework to reconstruct the mechanism for generating these data and predict possible future results.

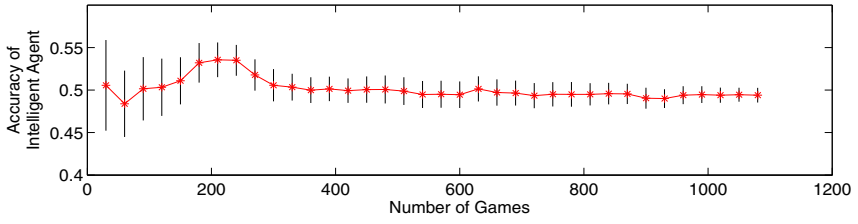


Fig. 3. Accuracy of intelligent agent with randomly generated sequence

3.1 Experimental Results

In the following experiments, we randomly selected 12 stocks from the Chinese market and US Dollar-RMB (Chinese Renminbi) exchange rate to test the MGDM model. The stock prices are from a downloadable software [17] and the exchange rate from the

Table 2. Descriptions on stock data and the Dollar-RMB exchange rate

Information / Name	Stock index	Beginning of test date	Mean accuracy after 1100 round
1. The Exchange Rate of RMB against U.S.Dollar	-	Dec 04 2000	63.92%
2. Shanghai Pudong Development Bank Co.	600000	Nov 10 1997	55.90%
3. CITIC Securities Co.	600030	Jan 06 2003	54.40%
4. China Yangtze Power Co.	600900	Nov 18 2003	55.22%
5. Shandong Bohui Paper Industrial Co.	600966	Jun 08 2004	55.81%
6. Kweichou Moutai Co.	600519	Aug 27 2001	53.66%
7. China Minsheng Banking Co.	600016	Dec 19 2000	55.41%
8. China Vanke Co.	000002	Dec 23 1991	53.36%
9. Baoshan Iron & Steel Co.	600019	Dec 12 2000	53.44%
10. Shenzhen Development Bank Co.	000001	Dec 23 1991	53.85%
11. Shenergy Co.	600642	Apr 16 1993	53.61%
12. Shenzhen Chine Bicycle Co.	000017	Apr 01 1992	52.71%
13. China Petroleum and Chemical Co.	600028	Aug 08 2001	52.06%

website [18]. For each stocks (with stock index) or exchange rate, we test successive 1100 trading days from the beginning date listed in table 2.

Each round of the game represents one trading day. Only given macro-level data $w(t)$, intelligent agent use MGDM to predict the most likely winning choice next round. Suppose the opening price is V_b and the closing price is V_f. Every trading day t, the fluctuation of the stock or exchange rate can be transferred to $w(t)$ by:

$$w(t) = \begin{cases} 1 & V_f \geq V_b \\ 0 & otherwise \end{cases}$$

At the end of every trading day t, intelligent agent select the best chromosome $\mathbf{x}^*(t)$ to make prediction for the next day based on equation 10.

The parameters for MGDM are as the following: the guessed maximum number of groups using fixed strategies $K = 20$, strategies S_k generated with memory length $m = 3$ and the population size $J = 50$. In order to avoid the influence of randomness from GA, we run the whole experiments for 30 times and the mean (marked with stars) and the standard deviations are recorded and plotted in figure 4. The final results after 1100 trading days are also shown in table 2.

In order to test the statistical significance, we first tested on a random sequence and the mean accuracy associated with standard deviations of the intelligent agent is shown in figure 3. It is gradually approaching 50% - which is true, because no learning method could take any advantage from a totally random market. Agents must have learnt something useful as long as the accuracy is over 50%. In figure 4, all prediction results are compared with such random sequence (or random walk) to test effectiveness of the MGDM framework.

3.2 Data Analysis

Figure 4 shows all the test results using data from real markets where each experiments contains 1100 steps (trading days). In each graph of figure 4, the upper curve indicates the mean accuracy (associated with the range of plus or minus standard deviations) of the intelligent agents. The lower curve shows the accuracy when intelligent agent make random choice without using any learning methods. As we can see that, in all the graphs showed in figure 4, intelligent agent using MGDM models performs considerably better than a random walk (by a margin from 2% to 14%), detailed results are listed in table 2, demonstrating the superiority the model in highly unpredictable stock markets.

In the graph of the exchange rate of RMB against U.S. Dollar (subfigure 1), the mean accuracy is up to 63.93% along with the ascendant trend. This kind of markets may has a strong pattern inside. The mean accuracy of most of the graphs is around range from 53% to 55%, e.g., the mean accuracy of the upper curve in subfigures 2, 3, 4 (Shanghai Pudong Development Bank Co. and CITIC Securities Co. and China Yangtze Power Co.) are almost stable at after 1100 rounds. However, in the last picture, after 1100 rounds, the deviations of the two curves are overlapped, which means that the MGDM is not statistically better than a random walk in this situation. The computation time is about 5 hours by running the Matlab code of the experiment for 30 times with an Intel Pentium dual-core PC.

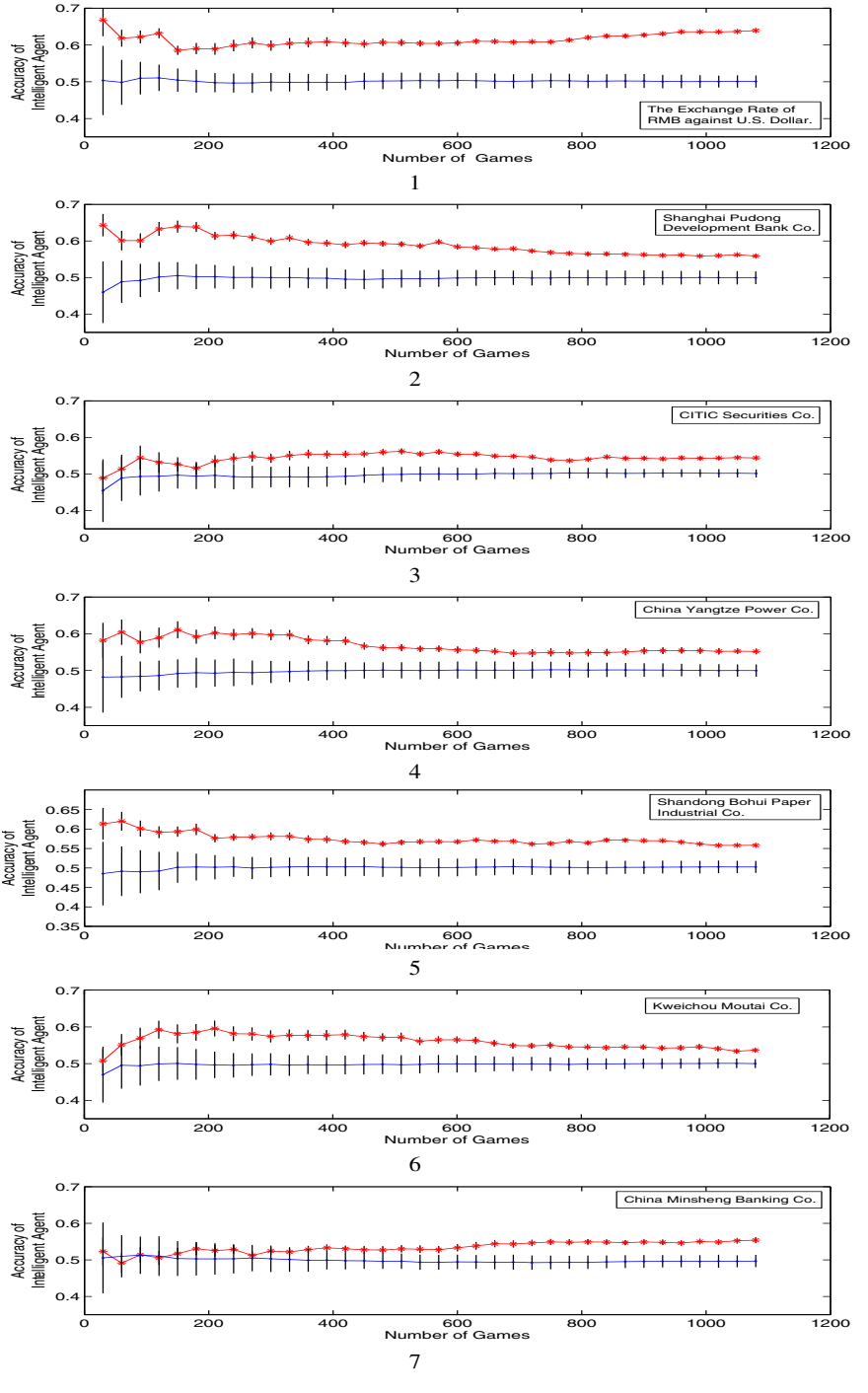
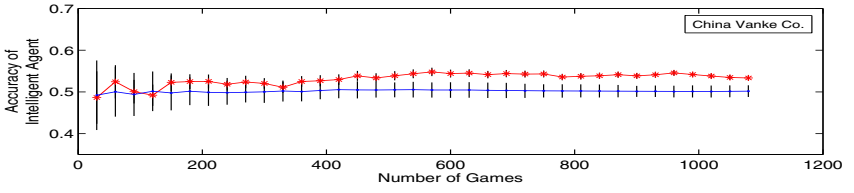
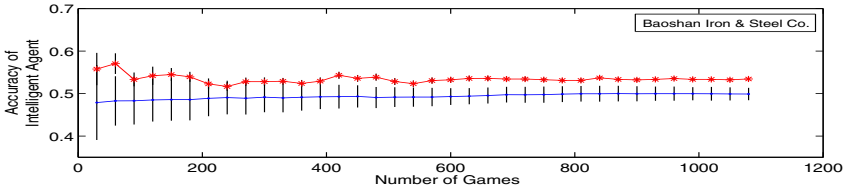


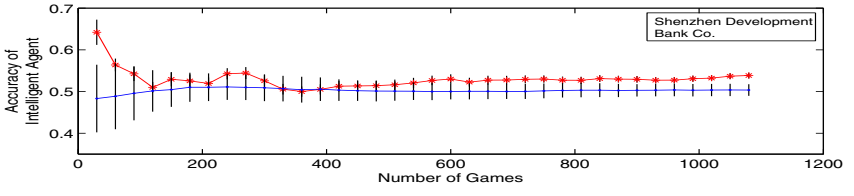
Fig. 4. Performance of MGDM in real stock and foreign exchange markets



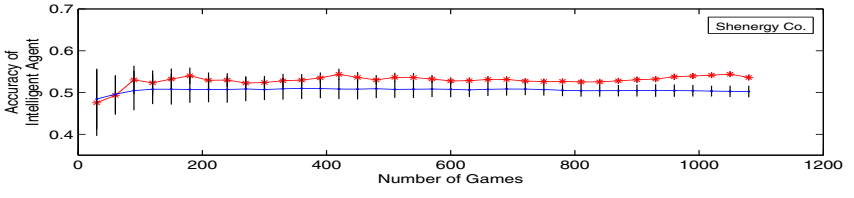
8



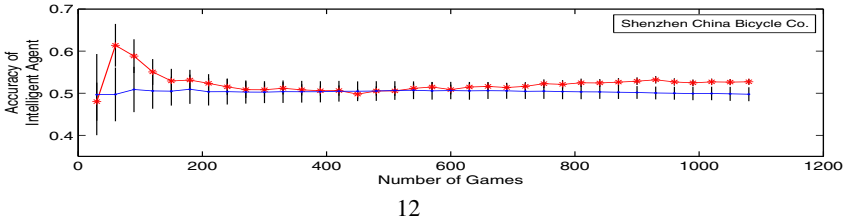
9



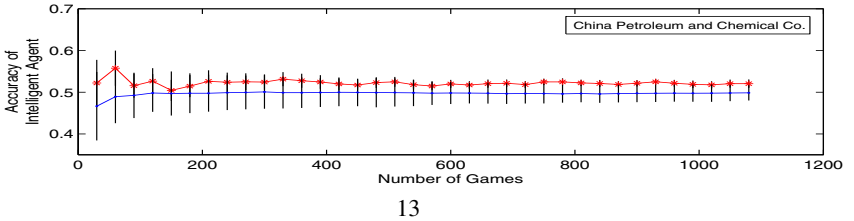
10



11



12



13

Fig. 4. *(continued)*

4 Conclusions

In this paper, we proposed a framework of learning time-series data by considering the collective data is an aggregation of several subgroup behaviors where each group of agents are following the minority game. By using a GA to explore the combination of decomposition structure of the system, an intelligent agent can beat the mark with a small winning rate.

We tested the framework on a few real-world stock prices and Dollar-RMB exchange rate. For most of the cases, the new framework of MGDM performs statistically better than a random walk - that proves the inefficiency of the market. The future work will focus on obtaining the real returns on stock markets.

Acknowledgment

This work is partially funded by the NCET Program of the Chinese Ministry of Education.

References

1. Arthur, W.B.: Bounded rationality and inductive behavior (the El Farol problem). American Economic Review 84, 406 (1994)
2. Challet, D., Marsili, M., Zhang, Y.C.: Stylized facts of financial markets and market crashes in minority games. Physica A 294, 514 (2001)
3. Challet, D., Marsili, M., Zecchina, R.: Statistical mechanics of systems with heterogeneous agents: Minority Games. Phys. Rev. Lett. 84, 1824 (2000)
4. Challet, D., Zhang, Y.C.: Emergence of cooperation in an evolutionary game. Physica A 246, 407 (1997)
5. Gode, D.K., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. Journal of Political Economy 101(1), 119–137 (1993)
6. Holland, J.H.: Emergence: From Chaos to Order (1998)
7. Li, G., Ma, Y., Dong, Y., Qin, Z.: Behavior learning in minority games. To appear in Collaborative Agent-Research and Development International Workshop (CARE) (2009)
8. Johnson, N.F., Jefferies, P., Hui, P.M.: Financial Market Complexity. Oxford University Press, Oxford (2003)
9. Lo, T.S., Hui, P.M., Johnson, N.F.: Theory of the evolutionary minority game. Phys. Rev. E 62, 4393 (2000)
10. Mantegna, R.N., Stanley, H.E.: An Introduction to Econophysics: Correlations and Complexity in Finance. Cambridge University Press, Cambridge (1999)
11. Qin, Z.: Market mechanism designs with heterogeneous trading agents. In: Proceedings of Fifth International Conference on Machine Learning and Applications (ICMLA-2006), Orlando, Florida, USA, pp. 69–74 (2006)
12. Qin, Z.: Naive Bayes classification given probability estimation trees. In: The Proceedings of ICMLA-06, pp. 34–39 (2006)
13. Qin, Z., Lawry, J.: Decision tree learning with fuzzy labels. Information Sciences 172(1-2), 91–129 (2005)

14. Rapoport, A., Chammah, A.M., Orwant, C.J.: Prsoner's Dilemma: A Study in Conflict and Cooperation. Uni. of Michigan Press, Ann Arbor (1965)
15. Smith, V.L.: An experimental study of competitive market behavior. Journal of Political Economy 70, 111–137 (1962)
16. http://en.wikipedia.org/wiki/El_Farol_Bar_problem
17. http://big5.newone.com.cn/download/new_zszq.exe
18. http://bbs.jjxj.org/thread-69632-1-7.html

Author Index