Towards Trustworthy AI for Autonomous Systems

Hadrien Bride[2], Jin Song Dong[1,2], Zhé Hóu[2(✉)], Brendan Mahony[3], and Martin Oxenham[3]

[1] School of Computing, National University of Singapore, Singapore, Singapore
[2] Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, Australia
z.hou@griffith.edu.au
[3] Defence Science and Technology Group, Edinburgh, Australia

Abstract. Trust remains a major challenge in the development, implementation and deployment of artificial intelligence and autonomous systems in defence and law enforcement industries. To address the issue, we follow the verification as planning paradigm based on model checking techniques to solve planning and goal reasoning problems for autonomous systems. Specifically, we present a novel framework named Goal Reasoning And Verification for Independent Trusted Autonomous Systems (GRAVITAS) and discuss how it helps provide trustworthy plans in uncertain and dynamic environment.

1 Introduction

Planning is a central and hard computer science problem that is essential in the development of autonomous systems. Many existing solutions require a controlled environment in order to function correctly and reliably. However, there are situations where adaptive autonomous systems are required to run for a long period of time and cope with uncertain events during the deployment. Our work is motivated by the requirements of next generation autonomous underwater vehicles (AUV) in law enforcement and defence industries. Particularly, we are currently developing a decision making system suitable for an AUV designed to stay underwater for up to 6 months with very limited communication with the outside world. The AUV is expected to carry out survey missions on its own and report details of its surveillance at semi-regular intervals. During the mission, the AUV may encounter underwater currents, deep ocean terrain, fishing boats, objects and places of interest, hostile vehicles etc., each of which may affect its ability to achieve its goals. The AUV must be able to decide which goals to pursue when such uncertain events occur and plan tasks to achieve the goals in an agile manner.

In the face of uncertain events in execution, planning becomes an even harder problem. In this case, the agent's goal may be affected and thus both selecting a new goal and re-planning are necessary. This generally follows a *note-assess-guide* procedure, where *note* detects discrepancies, *assess* hypothesises causes for

J. Sun and M. Sun (Eds.): ICFEM 2018, LNCS 11232, pp. 407–411, 2018.
https://doi.org/10.1007/978-3-030-02450-5_24

discrepancies, and *guide* performs a suitable response. Differing from classical planning where the goal is fixed, when a discrepancy is detected, it is often necessary to change the current goal. Goal reasoning is about selecting a suitable goal for the planning process. There have been various formalisms that attempt to solve planning problems in a dynamic environment, including hierarchical planning methods, such as hierarchical task networks (HTN) [3] and hierarchical goal networks (HGN) [8], and goal reasoning systems such as the Metacognitive Integrated Dual-Cycle Architecture (MIDCA) [2].

Although some of the above formalisms have been successfully applied to solve real life problems, the verification aspect of the problem remains to be addressed. Usually planning is solved by heuristic search, but this approach does not confer a sufficient level of trust. The correctness, safety, and security issues of autonomous systems are particularly important in mission-critical use cases such as our AUV example. To tackle this problem, we turn to formal methods, which have been used to solve planning problems in the literature. For example, Giunchiglia et al. proposed to solve planning problems using model checking [4] and Kress-Gazit et al.'s framework translates high-level tasks defined in linear temporal logic (LTL) to hybrid controllers [5].

Following the above ideas, in this short paper we introduce a new system called Goal Reasoning And Verification for Independent Trusted Autonomous Systems (GRAVITAS). This novel planning and goal reasoning framework has the ability to produce verifiable and explainable plans for autonomous systems. It is build upon the model checker Process Analysis Toolkit (PAT) [9], which is a self-contained tool that supports composing, simulating and reasoning about concurrent, probabilistic and timed systems with non-deterministic behaviours. The benefits of the proposed approach notably include the capacity to formulate inconsistency and incompatibility of plans as reachability/LTL properties and the ability to verify them on the fly. For instance, when a new goal is generated during execution, we can check whether the new goal conflicts with existing goals, and select the subset of goals that are compatible with each other. Finally, we can also verify the planning model itself, such that a given planning model does not output plans that may lead to undesired events.

2 Planning and Goal Reasoning via PAT

The plan and goal reasoning problems to be solved are expressed and formally defined as Goal task networks (GTNs) – an extension and unification of hierarchical task networks and hierarchical goal networks [8]. GTNs explicitly models the hierarchy among tasks and goals in ways that generally mirrors well the hierarchical structure of many real-world planning applications. This hierarchy can then be used during the planning phase following the well know *divide and conquer* scheme. Due to this, GTNs planners are much more scalable and performant than classical planners in practice.

In GRAVITAS, the verification and resolution of plan and goal reasoning problems expressed as GTNs is based on their translation to CSP# – one of the input language of PAT. This translation is fully automated and notably considers the

autonomous system capabilities as well as its environments. The translated CSP# code models all the elementary actions that the autonomous system can perform together with their effects on its environment. Furthers, it also considers resource constraints and goal reasoning (e.g., prioritization of goals). To do so, it assigns economic values to both its resources and its goals in order to leverage economic reasoning. By doing so, we leverage PAT optimisation features to formulate plans that incentivises the completion of goals providing the most rewards while compromising with the resources they require to be completed. These economic notions therefore lead to the formulation of highly cost-effective plan. Additionally, when multi-agents missions are considered, they provide further benefits as market-based mechanisms [1] can be leveraged to obtain greater collaboration among agents as well as to optimise resources and tasks allocation.

Since tasks and goals are both translated into processes in CSP#, it is straightforward to check properties for tasks and goals using PAT. For instance, using we can verify that the proposed plans respect predefined safety and liveness properties (e.g., the autonomous system does not collide with obstacles, the autonomous system has the ability to join the recovery area).

3 A Trustworthy Framework for Planning and Goal Reasoning

Compared with traditional AI techniques, the planning and goal reasoning methods in our work are realised by model checking, which is an automated reasoning technique that has been successfully applied in formal verification tasks. Hence an advantage of our approach is that we can use model checking to verify correctness, safety, and security properties of the underlying model.

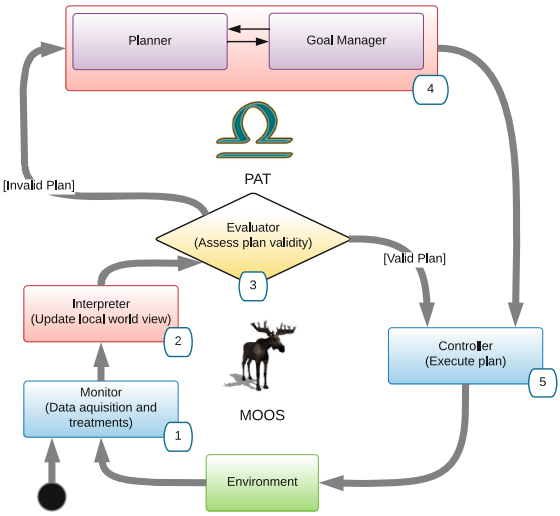

Fig. 1. Overall workflow of GRAVITAS.

To demonstrate the strengths of such approach we are developing *Goal Reasoning And Verification for Independent Trusted Autonomous Systems* (GRAVITAS) – a fully automated system enabling unmanned agents such as AUVs to autonomously operate with a high level of trust in a dynamic environment.

GRAVITAS follows a cyclic pattern composed of four main phases: Monitor, Interpret, Evaluate and Control. Figure 1 is a UML activity diagram of the overall control flow of GRAVITAS.

The main operative cycle of GRAVITAS begins with the Monitor (1). This component perceives the environment through the signal processing and fusion of the raw outputs of available sensors. It is also in charge of processing this data in order to provide information such as the estimated position and speed of the agent to the Interpreter (2). Once the Interpreter (2) receives the required information, it updates the agent's local model of the system and its environment. This formally defined local model is then forwarded to the Evaluator (3) – a component in charge of assessing the validity of the previously established plan with respect to pre-defined specifications. If the Evaluator assesses the plan to be valid, the Controller (5) is tasked with executing the plan. Alternatively, if the Evaluator (3) finds the plan invalid e.g. an uncertain event creates inconsistencies in the previously established plan and the mission requirements, a new plan needs to be formulated. The formulation of a new plan is accomplished by the joint operation of the Planner and Goals Manager components (4). After a new plan is formulated, the Controller (5) is tasked with executing this plan. This step involves processing based on control theory [6] which we do not discuss here.

In the developed framework, the components in the lower loop in Fig. 1 are orchestrated via the Mission Oriented Operating Suite [7] (MOOS) – a middleware mainly in charge of the communication. The main computational workload of the Evaluator (3), The Planner and Goal Manager (4) components are powered by PAT. Note that although conceptually the planner and the goal manager are two separated components, in our implementation they are concretized as a single PAT model. Also, note that, to achieve high efficiency in real-life applications, we use a hybrid approach to implement planning and goal reasoning: the PAT model performs high-level goal reasoning and planning, and we implement an external actuator to derive a low-level plan from a high-level plan, the former will then be sent to hardware for execution.

References

1. Clearwater, S.H.: Market-Based Control: A Paradigm for Distributed Resource Allocation. World Scientific, Singapore (1996)
2. Cox, M.T., Alavi, Z., Dannenhauer, D., Eyorokon, V., Munoz-Avila, H., Perlis, D.: MIDCA: a metacognitive, integrated dual-cycle architecture for self-regulated autonomy. In: AAAI, pp. 3712–3718 (2016)
3. Erol, K., Hendler, J.A., Nau, D.S.: UMCP: a sound and complete procedure for hierarchical task-network planning. In: AIPS, vol. 94, pp. 249–254 (1994)

4. Giunchiglia, F., Traverso, P.: Planning as model checking. In: Biundo, S., Fox, M. (eds.) ECP 1999. LNCS (LNAI), vol. 1809, pp. 1–20. Springer, Heidelberg (2000). https://doi.org/10.1007/10720246_1
5. Kress-Gazit, H., Fainekos, G.E., Pappas, G.J.: Temporal-logic-based reactive mission and motion planning. IEEE Trans. Robot. **25**(6), 1370–1381 (2009)
6. Lee, E.B., Markus, L.: Foundations of optimal control theory. Technical report, Minnesota University Minneapolis Center for Control Sciences (1967)
7. Newman, P.M.: MOOS-mission orientated operating suite (2008)
8. Shivishankar, V.: Hierarchical goal network planning: formalisms and algorithms for planning and acting. Ph.D. thesis, Department of Computer Science, University of Maryland College Park (2015)
9. Sun, J., Liu, Y., Dong, J.S., Pang, J.: PAT: towards flexible verification under fairness. In: Bouajjani, A., Maler, O. (eds.) CAV 2009. LNCS, vol. 5643, pp. 709–714. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02658-4_59