A GENERIC DOMAIN CONFIGURABLE PLANNER USING HTN FOR AUTONOMOUS MULTI-AGENT SPACE SYSTEM

Raveesh Kandiyil[1] **and Dr. Yang Gao**[2]

[1]*PhD Student, Surrey Space Centre, Guildford, GU27XH, United Kingdom*
[2]*Senior Lecturer, Surrey Space Centre, Guildford, GU27XH, United Kingdom*

ABSTRACT

Automated planning has been applied to numerous fields such as computer games, industrial robotics and even high-profile missions like planning and scheduling activities for Martian rovers. A current trend among the researchers is to apply automated planning in multiple space systems that work together in a coordinated fashion so as to attain highly complex mission goals. Even though automated planning and scheduling algorithms are mature in industrial scenarios and robotics, little consideration has been given to multiple-agent space applications. In this paper, we describe the development of a domain configurable planner which can be used for different space mission comprising of multiple systems i.e. satellites or rovers. The multiagent planning systems uses agent based modeling techniques, hierarchical task network (HTN) planning and a mixed approach from centralized and distributed planning. The initial results from the prototype planner are also discussed.

Key words: Autonomous Space systems; Multiagent Space systems; HTN planning; BDI agents.

1. INTRODUCTION

Automated planning is the computational process of finding a sequence of actions, given a domain description, an initial state and a set of goals that is to be attained. Planning plays an important role in developing autonomous systems since the autonomicity of the system relies upon how much rational the system is in taking decision. In the past decade space agencies have been developing custom made planners for different space missions, thus spending a huge amount of effort and resources in the overall process. Planning which is already a computationally hard process becomes more complicated when it has to plan for multiple agents and this research work address these problems by developing a domain configurable planner that is generic in nature and can be used for different multiple agent systems, thus reducing the cost and effort. Domain configurable planners can be used for different missions by altering the domain descriptions while keeping the planning engine intact i.e. the mathematical theory behind the planner is same even if it is used for different scenarios. In this paper a proposed architecture for developing such a planning system for multiple-agent autonomous space systems is described. The planner is integrated into a multi-agent architecture where agents such as satellites or rovers are modeled using BDI (Belief-Desire-Intention) [1] or its extension. The planner is made domain configurable for different multi-agent missions by keeping the heuristics and planning algorithm less dependent on the domain knowledge. In terms of acquiring domain knowledge the planner utilizes PDDL, but will be developed in a way that allows pre-existing machine learning techniques to be incorporated in future. Furthermore, the planner is capable of taking responsibility for distributing tasks among agents by using a mixed approach from centralized and distributed planning. This in turn reduces the communication hurdles of the distributed planning and at the same time gives a more robust central planning methodology. It can be compared to a manager who delegates tasks to sub-ordinates, by telling them what to do, but not telling them how to do it. The individual agent planners are given the domain description and a dedicated planner. The multiagent planning system includes a method database (MDB) which stores task decomposition methods and is accessible for all the individual agent planners. The compound task are delivered by the central planner (leader) to sub-planners in a hierarchical fashion after checking the preconditions, whether the resource constraints have been met by each agent. The central planner uses the resource manager (RM) to check such preconditions. Each individual agent then proceeds to planning phase for the execution of its assigned task (goals). The efficiency of the planner depends mainly upon the methods described for task decomposition. The planner discussed in this paper is being implemented in JAVA a mature language for agent modeling and AI planners. The paper also discusses the preliminary results of implementation. Once the planner is fully developed it will be tested and validated on a group of satellites/rovers which are already available in the department test bed facilities. Overall, the novelty of this work has been demonstrated through the development of a domain configurable planner using HTN (Hierarchical Task Network) planning and incorporating re-planning mechanism for multiple agent space missions.

2. AUTONOMOUS SYSTEMS AND PLANNERS

Planning and scheduling can be considered as the most important building block for most of the autonomous systems and an very detailed research has been conducted on this topic [3] until now. They are classified into different sections such as classical planning, neo classical planning, domain configurable planners, domain independent and domain dependent planners. Domain configurable planner consists of domain independent planning engine, but the input to the planner is domain specific information encoded in the domain description files. The planner can be operated on different scenarios by changing the domain specific information. This requires an expert who can code the domain knowledge into domain description files which is understandable for the planner. The only two existing domain-configurable planners up to author's knowledge are HTN planners and control rule planners[2]. HTN planning decomposes the high level tasks into subtasks and methods assist in breaking down these compound tasks to primitive tasks[3] whereas in control rule planner, the domain specific information controls the planner during search process by informing whether the current node can be pruned or not from the search space. The control rules are expressed in logical formalism. HTN planning is widely used when it comes to developing domain configurable planner and literatures explaining control rule planners for domain-configurable planners are not known much. A domain-configurable planner cannot be used in every domain for planning because it cannot guarantee efficiency in some domains. Moreover it would be tedious to build such a planner, but we can employ methods to suit the planner for a class of domains for e.g. a planner like the one in our work which can be used for multiple agents systems. It would be a good idea to look at the architecture of the autonomous system where this type of planner can be utilized, it's planning mechanism and what does it mean to be an agent in our system architecture. One of the highly cited and canonical architecture that is agreed upon by the research community is a three layered architecture of autonomous systems[4]. The highest layer i.e. the decision layer is where our planner is operating and the immediate bottom layer is execution layer that takes care of transferring all the execution plans into behavior specifications. This layer will also be monitoring the constraints violations arising from operations. The functional layer which is composed of elementary actions handles the hardware signaling and high speed computation. There are several reasons to use this architecture but first and foremost is that it allows the developer to concentrate on the planner rather than going into the fine details of the hardware

An autonomous space system should be able to plan its own activities, recover from failure and learn in course of time so that it can operate in conditions with less human interactions. These kinds of systems will require complex software running onboard where planning algorithm becomes an integral part of the system so as to justify the actions that are selected by the system. In the last decade several planning mechanism has been developed in industry and academia and these planners can be generally classified into domain specific, domain-independent and domain-configurable planner. The performance of the space based autonomous systems can increase a lot by using agent-based software architectures where individual spacecraft or its subsystems are modeled as agents. Even though, there are several agent based modeling tools available the real potentials of such software tools has not been utilized in space mission planning yet. Our methods proposes development of agent systems where each satellite/rover is represented using a agent that is built using BDI-style architecture. Unlike classical planning theory such as a state-space planning or HTN planning there doesn't exist a single planning theory which can be utilized to produce a robust multiagent plan for a group of agents. Even though multiagent planning in a hard problem when compared to single agent planning there are several advantages it can bring to space missions. The representation of the satellites and rovers as agents will consider them as separates entities with autonomous behavior and at the same time allow them to take part in the overall planning. To be truly autonomous system the agents should also have a good level of autonomy where they can react to the immediate problems arising due to the course of their actions. When the larger tasks is broken down to sub tasks with the help of agents it will increase the robustness and efficiency of the system

The definitions of agents, what they should do and their potential benefits has been clearly laid out in the past few years, yet a proper engineering principle for developing agent architecture has not been found in any literature[5] [6] [7]. Reactive architectures such as subsumption architecture possess intelligence as an emergent property of the system and planning depends upon the dynamic environment in which they operate. This form of planning is good in a dynamic environment, but in course of time several unwanted behaviors can emerge from the system and moreover an agent does not possess any reasoning capability which degrades the purpose of using agents. Intelligent agent needs to exhibit rational behavior and we would like to use deliberative planning in our multiagent planning system for these reasons. Perhaps the highly citied and most popular logical framework for developing agents was proposed by Rao and Georgeff known as BDI agents (Belief-Desire-Intentions). BDI agents have mental attitudes such as belief, desire and intentions which represents the status information, motivation or the rationale behind doing something and the intentions of the agent. The expected utility of BDI agents for multiagent architecture in autonomous systems is tremendously high [8][9]. Agent programming languages such as PRS (procedural reasoning system), JASON, 3APL and JAVA based implementation such as JADE [7] and JACK [10] are available for agent development. Among these JADE is one of the most popular agent development environments and widely maintained open source program. JADE only supports the development of agent system, but the intelligence of each agent solely depends upon how they are developed. Even though BDI agents have mental attitudes and goal

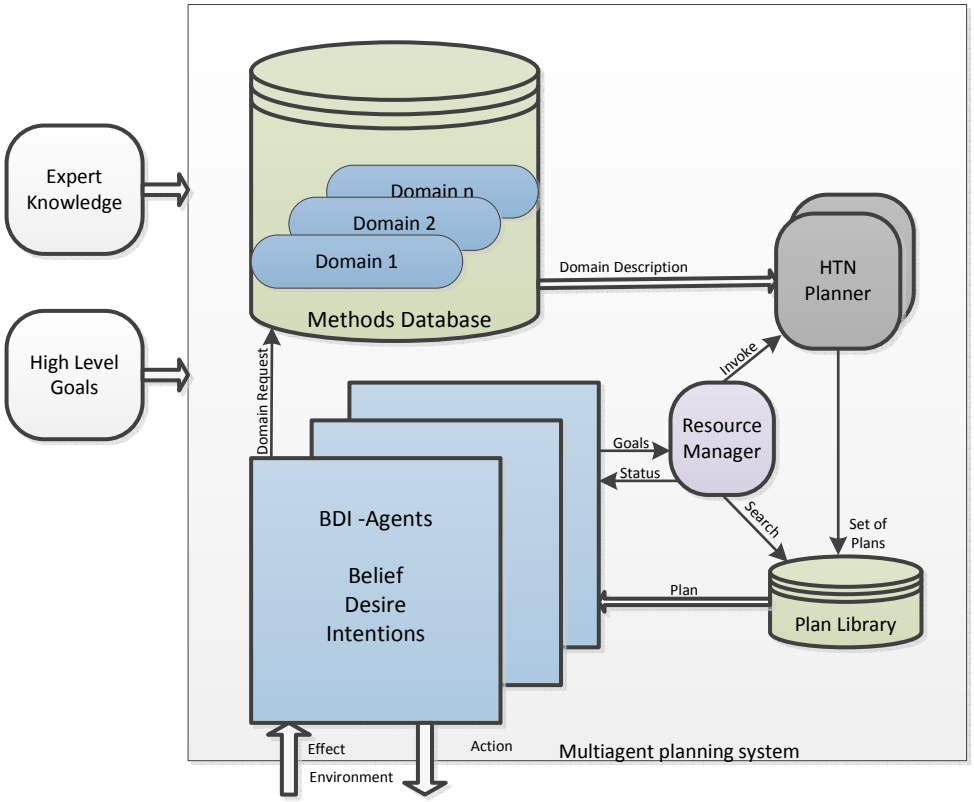

Figure 1. Multiagent planning system

oriented behavior they lack reasoning capability and no look ahead features. A very interesting tool JADEX [11] which can acts as a reasoning engine on the top of BDI jade agents can rectify the above mentioned problems and has proven to be successful in previous years. Combining these techniques it is possible to create a group of agents that are modeled using JADE with all the agents having BDI architecture and a reasoning engine JADEX on the top

3. PROPOSED SYSTEM ARCHITECTURE

The architecture of the system can be explained by the above figure and reviewing the BDI agent architecture. BDI agents have beliefs, desire and intentions as mental attitudes represented in possible world states. The intentions are the subset of the belief and desire that the agent has committed so as to attain the goals based on their world view. Each of the BDI agents in this multiagent system has a dedicated HTN planner which can be directly called by the agent during planning process. The planner in turn finds a set of action and those plans are stored in the plan library for execution. The very good similarity in the HTN planning and BDI style execution favors us in integrating a dedicated planner for each agent in the system. The highly dynamic nature of the BDI agents and the guaranteed solution of the HTN planner can thus produce a robust multiagent planning system[12] which is the two most important feature in space missions.

A high level representation of the domain is input to the multiagent planner with the help of a human expert and this makes the planner efficient and fast in respective domain. The high level goals are also input to the planner such as taking an image for a particular point of interest or attitude maneuvering and these goals are further broken down into sub goals by the planner. The sub goals are then assigned to different agents by a specific agent(central agent) that is selected by an algorithm described in [13]. The planning system uses a mixed approach from centralised and distributed planning. The agents don't have a fixed central planner, instead when a new high level goal arrives an agent is selected using a random agent selection algorithm. The random agent selection algorithm is required only in homogeneous agent system. If a mission involves heterogeneous agents where a specific agent has a different capability or can act as central planner, the agents don't need to run a random election algorithm since the central planner is selected by the user before planning process starts. In these cases the agents need to just execute the plans delivered from the central planner or can use dedicated planner if sub goals are delivered from the

central planner. In the former case, once the central agent is selected by the algorithm it break downs the high level goals into sub goals and is distributed to specific agent for further planning and execution with the help of Resource Manager(RM). Resource Manager has two units, first that helps the central agent to find out which agent is capable of doing each subtasks and invokes the appropriate planner, second that maintains a set of search algorithms which can help the agents to find previous successful plan and it also keeps the status of operation of each agents for e.g. if they are free at a specific moment or if the plan failed at a particular instance etc. The BDI agents are also updated about the actions of other agents by the resource manager. The planner initially finds a multiagent plan with the help of sub agents and starts executing it even if the plan is not completely found. The execution of actions can produce some effects in the enviroment or some agents might need to wait until the results of certain action are available. Most of the time there are cases when multiple effects are produced for a single action. These problems are solved by using an interleaved planning and execution technique and it also helps in identifying plan failure. The intermediate and successful plans are stored in the plan library which can be used in future.

3.1. Domain Description

The domain description files stores the hand coded recipes for decomposing the compound tasks into primitive tasks. These files are stored in the method database from where the agents can select the domain description files that are relevant for each compound tasks to be decomposed. An excerpt from a PDDL [14] file for scenario which describes the operators for taking image from a satellite is shown below.

Primtive Tasks

```
((:operator(!warm_up)((at ?x))(())())
(:operator (!camera_on )((at ?y))()())
(:operator (!orient_camera )((at ?y))((at ?y))(( at
?z)))
(:operator (!take_picture)((at ?y))()())
(:operator (!antenna_on)((at ?y))()())
(:operator (!communication_start)((at ?y))()())
(:operator (!transfer)((at ?y))()())
(:operator (!communication_off )((at ?y))()())
(:operator (!antenna_off )((at ?y))()())
(:operator (!camera_off )((at ?y))()()))
```

Methods

```
(:method (get_image ?y) method name
((at ?x) (not (at ?y))) precondition for methods
((!warm_up ) (camera_on ?y) subtasks
(!orient_camera ?y)(!take_picture ?y)
(communicate) (!antenna_off ?y)
(!camera_off ?y) ))
```

3.2. Agent Modeling

The BDI model of the agent is described using the same technique as JADEX agents. Each agent is described in XML and stored in a Agent Description File (ADF)[11]. The agent description file contains all the static properties of the agent and the plans for the agents are produced from the HTN planner externally. The executable plans are then stored in agent's plan library so that instead of planning from the scratch a successful plan can be utilized in future given the precondition are always satisfied. The agents can also store reactive plan for handling different scenarios making it faster in operations and decreasing time for planning. The standard format for agent description in XML is shown below.

agent definition
<agent id="satellite" >
set of beliefs
<beliefs>
<belief name="currentpointing" class="orient">
<fact >new orient() </fact>
</belief>
</beliefs>
set of goals
<goals>
<achievegoal name="PointOfInterest">
<parameter name="targetPOI" class="orient"/>
</achievegoal>
set of plans
</goals>
<plans>
<plan name= "Strategy">
<body>new Strategicplanner()</body>
</plan>
</plans>
</agent>

4. RESULTS

The multiagent planner is in the very early stages of the development and an initial result from the prototype planner is presented here. Currently no GUI has been developed for the prototype. The results shown here is for two satellites which have to point in a same direction to take a 3D image of a particular point of interest. The first satellite plan produced a sequence of actions starting from warming up the camera to communication for transferring image and the second satellite plan produced an sequence for synchornising with the first satellite and orienting the camera in the direction of the point mentioned by first satellite. At this moment there is no guarantee that the plan are free from temporal conflicts since none of the temporal information are encoded neither in the domain description nor externally. The next stage of the research is to impart temporal knowledge since feeding tempo-

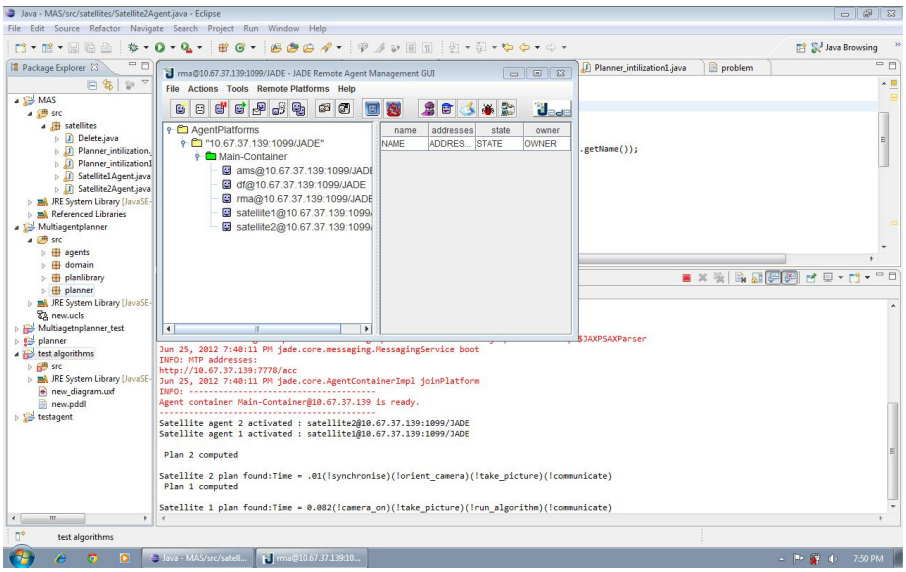

Figure 2. Different plans found for two satellites

ral information into the domain knowledge will help in avoiding temporal conflicts as much as possible during initial planning process. Another possibility of using BDI agent's desire to guide planning is being investigated for future work.

5. RELATED WORK

Previous works about modeling satellites as agents system can be found in several literatures and especially in [13] where the developers have modeled each subsystem of a satellite as agents and also used HTN planning. In our work we are using BDI agent modeling so as to make the agents more rational and also can invoke the planner depending upon specific goal criteria. Moreover the planning system is made domain configurable by making the planning engine least dependent on the domain it works. Another work which employs a state-based HTN planner in the BDI agent architecture is done in [15] but our planning system distributes the sub goals to the individual planners before planning is initiated and this speeds up the planning process.

ACKNOWLEDGMENTS

We would like to thank the open source JADEX community at Technical University of Hamburg, Germany and SHOP planner developers at University of Maryland, United States for providing the technical details on their websites.

REFERENCES

[1] Rao, A.S. & Georgeff, M.P.(1995). BDI Agents: From Theory to Practice. *In Proceedings of the first international conference on multi-agent systems (ICAMS-05)*, pp 312-319.

[2] Nau, D.S.(2007),Current trends in automated planning. *Artificial Intelligence magazine*. **28**(4),43-58.

[3] Nau, D.S., Ghallab, M. & Traverso, P. (2004), *Automated Planning: Theory & Practice*, Morgan Kaufmann Publishers Inc, USA, pp 229-345.

[4] Gat, E.(1998), On Three-Layer Architectures. *Artificial Intelligence and Mobile Robots*, MIT Press, USA.

[5] Wooldridge, M., & Jennings, N.R.,(1995), Intelligent agents: Theory and practice. *Knowledge Engineering Review*.**10**,115-152.

[6] Rao, A.S., & Georgeff, M.P.,(1995), *Formal models and decision procedures for multi-agent systems*, Technical report, Australian Artificial Intelligence Institute, Melbourne, Australia.

[7] Bellifemine, F.L., Caire, G., & Greenwood, D., (2007), *Developing Multi-Agent Systems with JADE*, Wiley, UK.

[8] Jennings, N.R., (2001), An agent-based approach for building complex software systems. *Communications of the ACM*, **44**(4), 35-41.

[9] Weiss, G., (1999), *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, MA, USA.

[10] Evertsz, R., et al. (2003), Implementing Industrial Multi-agent Systems Using JACK. *International Workshop on. Programming Multiagent Systems PROMAS 2003*, pp 18-48.

[11] Pokahr, A., Braubach, L., & Lamersdorf, W., (2005), Jadex: A bdi reasoning engine. *Multi-Agent Programming*, Springer Science+Business Media Inc., USA, pp 149-174.

[12] de Silva, L., & Padgham, L., (2004) , A comparison of BDI based real-time reasoning and HTN based planning.*Proceedings of the 17th Australian joint conference on Advances in Artificial Intelligence*, pp 1167–1173.

[13] Amigoni, F., Gualandi, S., Menotti, D., & Sangiovanni, G., (2010), A multiagent architecture for controlling the palamede satellite.*Web Intelligence and Agent Systems*, **8**(3), 269-289.

[14] Mcdermott, D., et al (1998), *PDDL - The Planning Domain Definition Language*, Yale Center for Computational Vision and Control.

[15] Walczak, A., et al(2006), *Augmenting BDI Agents with Deliberative Planning Techniques*, In The Fifth International Workshop on Programming Multiagent Systems PROMAS-2006, Hakodate, Japan.