Multi-Agent Planning Under Complex Constraints for Deep-Space Probes Group

Yuting Zhao[1,2], Zhaoyu Li[1], Shenying Zhu[1,2], Zixuan Liang[1,2], Rui Xu[1,2]

1. Beijing Institute of Technology, Beijing, 100081, China

E-mail: xurui@bit.edu.cn

2. Key Laboratory of Autonomous Navigation and Control for Deep Space Exploration, Ministry of Industry and Information Technology, Beijing, 100081, China

E-mail: zhaoyuting_bit@163.com

Abstract: With the development of launch technology and the miniaturization of probes, multi-probe systems can be used in space missions, and the demand for on-board planning is rising. Multi-probe planning needs tasks to be allocated properly to increase the total task profit and satisfy the complex time and resource constraints of probes. We propose to solve this problem using a multi-agent system, taking advantage of its negotiation strategy in task allocation to increase the global task profit. And we add complex constraint handling ability to agents, to deal with the more realistic and detailed constraints models in practical engineering. Before planning starts, a dynamic distributed multi-probe group formation pattern is designed to cope with the unstable communication condition between probes. In the multi-agent system, two types of agents are designed: execution agents represent task accomplish abilities of probes and target agents represent target requirements in the task. A novel three-phase multi-agent negotiation planning method is proposed, target agents ask for high-profit actions and execution agents respond depend on constraints and optimal strategy. The last phase of planning satisfies resource constraints through an action-bind strategy, which can handle constraints between overlapped resource consumption and resource generation actions. The experimental results show that the proposed multi-agent planning method can obtain rational plans satisfying complex constraints and increases the task profit compared to the traditional greedy search method.

Key Words: Multi-agent planning, Complex constraints, Deep-space probes

1 INTRODUCTION

A probe group that consists of multiple small probes has higher flexibility and better anti-damage ability than a single probe in a deep-space exploration mission. With the trend of miniaturization of spacecraft and the increase of rocket carrying capacity, multi-probe systems will be a new choice in the future. For instance, ANTS (Autonomic Nano Technology Swarm) project of NASA plans to use 1000 small probes, which are divided into different types of agents according to their functions, to explore the asteroid belt[1,2].

In deep space exploration missions, the probe is far away from the earth, thus there is a long communication delay between the probe and the earth. The traditional remote control and telemetry method makes probes react for a long time and has poor flexibility. Therefore, autonomous planning technology can enable deep-space probes to make decisions autonomously according to the environmental information and mission requirements, reduce the dependence on the ground system, shorten the task response time of probes, and better handle the unknown complex environment in space.

This work is supported by National Nature Science Foundation under Grant 61976020, the Industrial Technology Development Program under Grant JCKY2018602B002, and the Civil Aerospace Technology Research Project of China under Grant MYHT201705.

Existing multi-spacecraft planning problems mainly focus on mission planning of earth observation constellations. The planning methods adopted include evolutionary algorithms [3], swarm optimization [4], iterative repair method [5], multi-agent planning method [6-8], etc. According to the existing research, the multi-agent method is the trend in the future. Although the research on spacecraft planning with multiple deep-space probes is still insufficient, the existing planning methods for earth observation task planning can be used for reference.

In this paper, the multi-probe observation task planning is modeled as a multi-agent task planning problem. Compared with ground planning, on-board task planning for deep-space probe needs to consider the limited computing power and the instability of inter-probe communication. This paper not only pays attention to the specific planning algorithm but also designs the collaborative planning schema between probes, which organizes the collaboration in an orderly way to reduce unnecessary inter-probe communication. Unlike planning methods that only regard probes as agents, this paper establishes two kinds of virtual agents in a leader probe that makes plans for the group. The probes are executing agents, and the space targets that need to be observed are the target agents. The planning process not only optimizes the task allocation of target observing but also deals with complex time and resource constraints.

The rest of the paper is organized as follows. Section 2 describes the multi-probe task planning problem for deep-space observation. Section 3 proposes a dynamic multi-probe group formation pattern. Section 4 develops the TPMANP algorithm (Three-Phase Multi-Agent Negotiation Planning). Finally, section 5 presents the evaluation of the TPMANP algorithm.

2 PROBLEM DESCRIPTION

This section presents the description of the multi-probe planning problem in deep-space observation tasks, including the task requirements and probe capabilities descriptions.

2.1 Task Requirements Description

In an observation task with multiple probes, probes can cooperate with each other to observe more targets in limited time. Consider the probes are far away from the earth, on-board planning is a vital technique to improve the efficiency of the task execution. The relative positions of probes are changing along with the whole task, so the communication links between probes are not stable. Thus, this is a multi-agent planning problem under unstable communication conditions, which means the message exchanges between agents are affected. The task requirements are defined as follows:

$$Task = \{name, time, Tar\} \tag{1}$$

where $name$ is the identification of the task, there may be more than one task during the flight of the probes; $time$ means the required start time and the end time of the task; $Tar=\{tar_1,tar_2,...,tar_n\}$ is a target set including all the targets in this observation task, each target has its own name, position and priority. Each target may need one or several times observation, depending on the task background.

$$tar = \{name, position, pri\} \tag{2}$$

2.2 Probe Capability Description

A probe is an agent that makes plans for tasks and executes plans. The capability of a probe is defined as follow:

$$\begin{aligned} Probe = \{&name, TimeCons, ResourceCons, ObACT\_set,\\ &DataACT\_set\} \end{aligned} \tag{3}$$

We simplify the model of the probe, mainly focus on the observation activities $ObACT = \{id, Rate, t_s, t_e\}$ and data transformation activities $DataACT = \{id, Rate, t_s, t_e\}$. Where id is the action identifier, t_s is the start time of the action and t_e is the end time of the action. Actions in sets $ObACT\_set$ and $DataACT\_set$ are durative actions that change the resource amount at a stable rate $Rate$.

$TimeCons$ denotes the time constraints of the probe. There are two types of time constraints in this problem. The first one is that a probe can only observe one target at a time unless the probe is able to observe different targets in the same attitude. The second is that there should be a time duration for preparation between two observation activities.

$ResourceCons$ denotes the resource constraints of the probe, we take the memory storage as an example. The amount of the rest storage is changing along with the executions of observations and data transformations. The storage amount should always be kept in an interval. In many spacecraft planners, a resource consumption action and a resource production action don't overlap, but we build a more realistic model that allows different types of actions to be executed concurrently.

A plan is aiming at meeting the task requirements under the complex constraints. Due to the limited probe resources, it is acceptable that a few targets are not observed, and the number of observed targets is the optimization goal of the problem. When some targets have to be abandoned to satisfy constraints, we choose the low priority targets.

3 DAYNAMIC MULTI-PROBE GROUP FORMATION PATTERN

There are mainly two structures of multi-agent planning systems: centralized and distributed. Centralized systems need an agent to plan for the whole group, so the central agent has a heavy computing load. Distributed systems have a heavy communication load [9]. To solve these problems, we propose a structure that combines these two structures: a distributed system with a center. The agents in the system can do part of the job individually and a central agent does the left to avoid the communication overload.

Communication is important for forming a group. While the communication condition between probes is not stable due to the changing relative positions and the uncertain environmental effect. Therefore, we propose a dynamic multi-probe group formation pattern that depends on the prediction of the communication links. Each probe is capable of planning and communication. Once a probe receives a task request from the earth, it begins to predict the persistence of links and then chooses probes among which communication links are stable. This temporary group is responsible for the task, and will then make a plan for the task and execute it.

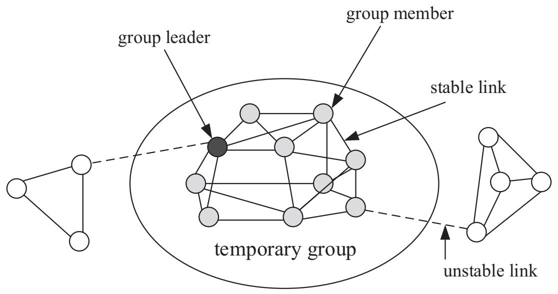

Fig 1. Dynamic multi-probe group formation pattern

As shown in Fig. 1, the probe that receives the task is the leader of the temporary group, links between group members are stable in a time duration, which ensures enough time for planning. The other probes that are not in the group have no links with the group members or the link is unstable, which means the link will not exist long enough. Group leader sends task requests to the members so that the group can choose a new leader to finish the planning if the leader probe fails during the planning processes.

Before planning, there are some preparation works that accomplish by each probe to decrease the calculation load of the leader probe. After receiving task requests, each group member calculates its possible observation actions and data transformation actions according to the target positions and the probe orbit parameters. Each action has a time duration as defined in section 2.2, and the duration can be modified during planning to satisfy time or resource constraints. These actions will be sent to the leader probe to build virtual execution agents.

4 PLANNING ALGORITHM

The TPMANP algorithm is divided into three phases based on multi-agent cooperation. We build two types of virtual agents in the leader probe. A target agent (TA) stores observation actions of the target and manage the actions it stores. An execution agent (EA) represents a probe, which receives observation actions from target agents, stores all the possible data transformation actions the probe has and manages the actions. As depicted in Fig 2, the first phase is greedy task allocation in which each target agent evaluates the observation actions it has and sends the best action to the corresponding task execution agent. The second phase is agent negotiation in which two kinds of agents send messages to each other to negotiate the allocation of observation actions. Meanwhile, execution agents check the time constraints of probes. In the third phase, resource constraints handling algorithms are designed to check the data storage constraints of execution agents. At the same time, execution agents select data transmission actions according to the situation of constraints satisfaction and abandon the observation tasks that violate constraints.

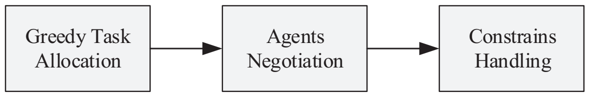

Fig 2. Three phases of TPMANP algorithm

4.1 Greedy Task Allocation

Target agents perform greedy task allocations. The observation actions calculated by probes are sent to the leader, and each TA stores actions that observe the target it represents. Fig 3 illustrates this process. In this paper, we take the duration of an action i as an evaluation criterion $Value_i$, because the longer the duration is, the more information the probe can get from a target.

$$Value_i = etime_i - stime_i \tag{4}$$

We could also set other parameters as criteria, such as the start time of an action if we want targets to be observed as early as possible.

Each TA sorts actions according to the values of them, and then sends the best action to corresponding EA. If the target needs to be observed by more than one action, and these actions have time constraints between them, the TA has to deal with constraints first before sending actions to EAs. For example, we do not want a target to be observed by different probes at the same time, therefore the TA has to check and adjust the time duration of the actions to make sure they won't overlap each other.

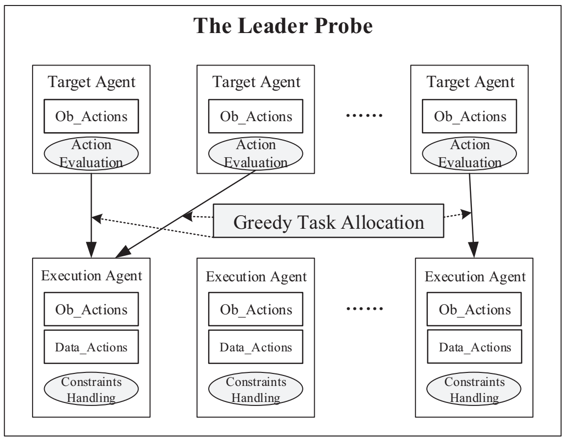

Fig 3. Greedy task allocation process

4.2 Agents Negotiation

Negotiation is the core part of this planning method, the aim is to optimize the observation tasks allocated in EAs to satisfy time constraints and to observe high priority targets as many as possible. Observation actions sent from TAs are observation tasks for EAs. The value of an observation task is different from an observation action. We have to consider the priority of the target and the time duration of the task. We do not calculate task values, but compare different tasks according to their target priority, if two tasks have the same priority, then compare the time durations of tasks, the longer task is more valuable.

Algorithm 1 Agents Negotiation

```
for each EA in Agents do
  for each ObAction in EA do
    bool c = Time Constraints Processing();
     if c==false then
     Send Message to TA();
     TA Add Best ObAction to Corresponding EA();
     end if
  end for
end for
```

The whole negotiation process starts from the first to the last EA by the probe id. As shown in Fig 4, the detailed algorithm is as follows:

1. The planner sorts all the observation tasks of this EA by task start time and check the time constraints between neighboring tasks.

2. If there are two neighboring tasks $task_i$ and $task_{i+1}$ do not satisfy the constraint $stime_{i+1} - etime_i \geq ptime$, where $ptime$ is preparation time, reserve the more valuable one and delete another from EA.

3. EA sends messages to corresponding TA if a task is deleted, the TA selects the most valuable action if any and sends the action to the EA that executes the action as a new task. If no task is deleted in this EA, then go to the next EA. Sign the EA as constraint checked after step 3.

4. If the EA that receives a new task is not constraint checked, put the new task into the task list directly. If the EA is checked, insert the new task to the plan which means we have to shorten the time duration of the task or its lower value adjacent task to satisfy time constraints.

5. Go to the next EA and repeat steps 1 to 4 until all EAs are signed as checked.

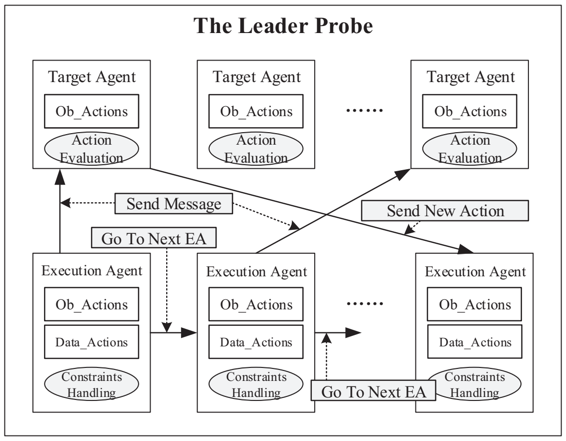

Fig 4. Agent negotiation process

4.3 Constraints Handling

There are two kinds of constraints we consider in this problem: time constraints and resource constraints. Time constraint defines the temporal distance between two adjacent observation actions which is the preparation time of the probe between two observation actions as shown in Fig. 5.

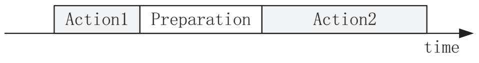

Fig. 5. Time constraints definition

Resource constraints in a probe are mainly memory storage constraints and battery energy constraints. We can handle them in the same method because they have the same essence. Some actions consume resource while the other produce resource, the amount of resource change depends on the consumption (production) rate and time duration of the action. The plan of the problem has to make sure the resource amount keeps between upper and lower limits. As shown in Fig 6, there are two actions, Acion1 and Action2 with different rates k, $k_1>0$ means Action1 produces resource, $k_2<0$ means Action2 consumes resource, the upper limit is resource r=C, C is a constant, and the lower limit is r=0.

Most of the existing multi-agent planners are domain-independent, and the time and resource constraints are not the main problems they focus on. Few of multi-agent planners deal with these constraints, such as [9], [11] and [12]. One of the traditional methods to handle time constraints in planning is to build an STN (Simple Temporal Network) [13-15] that models durative actions and time constraints as a network. Shortest path algorithms are used in the STN problem to propagate constraints and find the shortest duration of each action. Nevertheless, in our problem, we are not interested in the shortest duration and do not need to build constraints between all the time points. We use a more efficient method to handle time constraints. We propose Time Ordered inTerval Adjustment (TOTA) algorithm. The algorithm sorts all the durative actions in a timeline by start time and check adjacent actions in pairs. If time duration between a pair of actions is not long enough, adjust the time points of actions to meet the request. The strategy of changing time points depends on the value of the action and the requirements of the task. There are three types of strategies as presented in Fig. 7: changing the end time point of the former action, changing the start time point of the later action, or changing these two points proportionally.

The last step of planning is to check all the data transmission actions of each probe. Whether the data transmission actions (DA) and observation actions (OA) already in the plan violate the resource constraints. We use the data storage constraint to explain the method and battery energy constraints can be handled in the same way. The data amount R of each probe should satisfy constraint $0 \le R \le R_{max}$ at any time, R_{max} is the maximum data storage capacity.

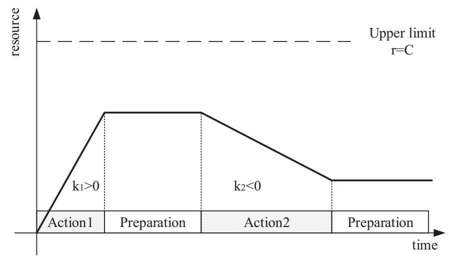

Fig. 6. Resource constraints definition

We propose an Action-Bind Algorithm (ABA) to deal with resource constraints. We divide actions into two types, resource consumption action and resource production action. In this problem, an OA is a resource production action that increases the amount of R. Bind a DA with all the OAs that happened before its start time and has not banded by other DAs. In this way, a DA is responsible for the OAs bound to it and the resource amount calculated from bound action pairs before it. If the resource amount violates the constraint, adjust the time interval of current DA and OAs.

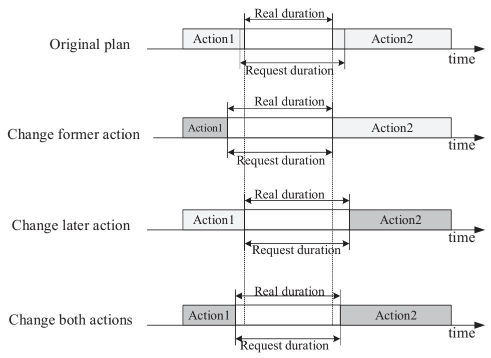

Fig.7. Three strategies of TOTA algorithm

Due to the characteristic of this problem, the time interval of an action can only be shorter, which means it can start later or end earlier but cannot be changed in the opposite way. When the storage is empty, current DA has nothing to be responsible to, abandon this DA. Check the DA-OAs pairs chronologically, we finally achieve a plan with OAs and DAs that satisfy time and resource constraints.

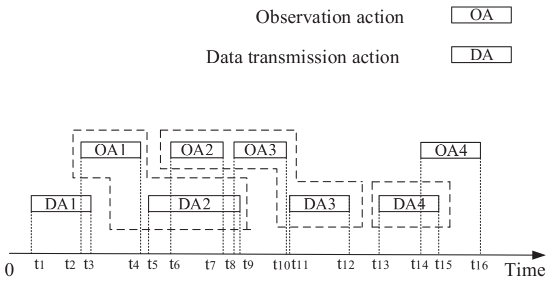

Fig. 8. Action-bind model

5 RESULTS AND DISCUSSION

To test our planning algorithms, we set up a scenario with five probes and two ground stations. The duration of the testing scenario is 24 hours. The preparation duration of a probe is 300 seconds. The observation action resource produce rate is 1MB/s, the resource consumption rate of data transmission actions is 3MB/s. The storage capacity is 5000MB. The target amount change from 5 to 100. One hundred targets are evenly divided into four different priorities.

Table1. Parameters of tests

Test ID	Targets IDs in task request	High priority targets	Second priority targets	Third priority targets	Lowest priority targets
1	1-5	0	5	0	0
2	1-10	0	5	5	0
3	1-20	5	5	5	5
4	1-50	10	15	15	10
5	1-100	25	25	25	25

Table 2. Result under different target numbers

Test ID	Target number in task request	Observed target amount in final plan	Planning time (*ms*)
1	5	5	0.719
2	10	10	0.781
3	20	20	0.861
4	50	49	2.304
5	100	93	2.637

Tests were performed on a 3.6GHz Intel i7 computer with 12 GB of memory. The test results are shown in Table 2. The target number changes from 5 to 100. We test tasks under each target number 10 times and record the average planning time. As the target number grows, the amounts of initial available OAs and DAs are increasing, so is the possibility of constraint violations. Due to the limited ability of probes and the strategy of the planner, there are some targets unable to get OAs in the plan, so the observed target number in the final plan is possibly less than the number in the task request. Table 3 shows the planning result of a task with twenty targets. From Table 3 we can see that the plan result satisfies time and resource constraints in the task requests.

We compare the TPMANP algorithm with a pure greedy algorithm without negotiation. Both algorithms use the same time constraints and resource constraints handling method. The results are illustrated in Fig.9, it is clear to see that the negotiation increases the percentage of final observed targets in the plan, thus optimizes the task profit. The reason is that our planning algorithm is able to find a substitution through the negotiation when an OA violates any constraint. While the pure greedy selection algorithm chooses all the best OAs at once and has no choice but to delete actions when the current plan violates constraints.

Table3. Planning result of a task with twenty targets

Target ID	Action type	Start time(*s*)	End time(*s*)	Probe ID
11	OA	27713	27776	1
2	OA	80109	80129	1
1	DA	38853	38874	1
1	DA	80223	80229	1
7	OA	430	450	2
8	OA	5199	5219	2
13	OA	11316	11336	2
5	OA	17034	17054	2
3	OA	22946	22966	2
12	OA	33438	33458	2
1	OA	39015	39035	2
16	OA	49123	49143	2
20	OA	49866	49886	2
15	OA	56261	56281	2
18	OA	71384	71404	2
2	DA	22771	22797	2
1	DA	38838	38851	2
2	DA	68463	68490	2
1	DA	80210	80217	2
9	OA	2277	2297	3
10	OA	7741	7761	3
17	OA	28468	28488	3
4	OA	39651	39671	3
2	DA	18470	18484	3
1	DA	30158	30165	3
2	DA	56120	56127	3
19	OA	22202	22237	4
14	OA	60294	60314	4
2	DA	35133	35144	4
2	DA	72803	72809	4
6	OA	16347	16367	5
1	DA	22203	22209	5

We simulate the resource change according to the task plans of five probes. The amount of the data is changing along the time, and the values are within the storage constraint. All DAs in the plan is necessary because our algorithm only add

DA in plan when there is data left in the storage. We avoid adding any meaningless action to the plan, thus reduce the burden on the control system. The method avoids random search in the planning, so it is theoretically faster than evolutionary algorithms or swarm optimization, therefore more suitable for on-board planning.

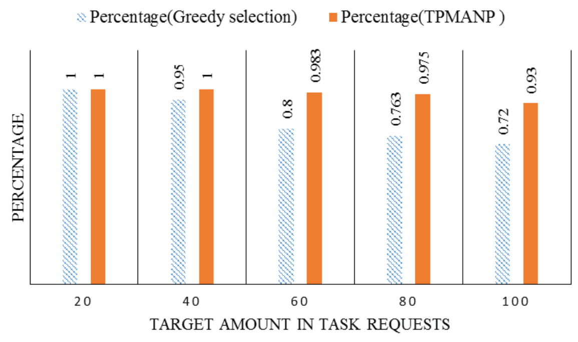

Fig. 9. Percentages of final observed targets

6 CONCLUSIONS

In this paper, we present a multi-agent planning system under complex constraints for a small probes group in deep-space exploration tasks. We propose a dynamic multi-probe group formation pattern that adapts to the dynamic change of communication links between probes. Based on the temporary stable probe group that forms according to the pattern, we introduce a three-phase planning algorithm using multi-agent negotiation. In order to deal with complex constraints in the problem, we design a time constraint handling algorithm TOTA and a resource constraint handling algorithm ABA. The test results indicate that our planning algorithm can make a reasonable plan for a multi-probe observation task.

Our future works will focus on a more complex background that communication conditions change during the planning. Moreover, a better way to optimize the profit of the task needs further research. After we determine the observation and data transmission actions, we can also consider making a more detailed plan for subsystems of a probe.

REFERENCES

[1] F.C. Liu, Application of artificial intelligence in spacecraft, Flight Control & Detection, 1 (1), 016-025, 2018.

[2] W.Truszkowski. H.L. Hallock, C. Rouff, J. Karlin, J. Rash, M. Hinchey, Swarms in space missions. Springer, London, 207-221, 2010.

[3] A. Globus, J. Creawford, J. Lohn, R. Morris, Scheduling earth observing fleets using evolutionary algorithms: problem description and approach. Proceeding of the 3rd International NASA Workshop on Planning and Scheduling for Space, 27-29, 2002.

[4] L. Jian, C. Wang, Resource planning and scheduling of payload for satellite with particle swarm optimization. Proceedings of SPIE - The International Society for Optical Engineering, 67951E-67951E-6, 2007.

[5] Mohammed, L. John, SpaceCAPS: Automated Mission Planning for the TechSat 21 Formation-Flying Cluster Experiment, Fifteenth International Florida Artificial Intelligence Research Society Conference, 2002.

[6] P.O. Skobelev, E.V. Simonova, A.A. Zhilyaev, V.S. Travin, Application of multi-agent technology in the scheduling system of swarm of earth remote sensing satellites, Procedia Computer Science, 103(C): 396-402, 2017.

[7] J. Bonnet, M.P Gleize, E. Kaddoum, S. Rainjonneau, Rapid and adaptive mission planner for multi-satellite missions using a self-adaptive multi-agent system, The 67th International Astronautical Congress, 2016.

[8] H.C. Hao, J. Wei, Y.J. Li, Z.Q. Yua, Research on agile satellite dynamic mission planning based on multi-agent, Journal of National University of Defense Technology, 01, 53-59, 2013.

[9] A. Torreño, E. Onaindia, A. Komenda, M. Štolba, Cooperative multi-agent planning: a survey, Acm Comput Surv, 50, 1-32, 2017.

[10] J. Kvarnström. Planning for loosely coupled agents using partial order forward-chaining, International Conference on Automated Planning and Scheduling, ICAPS 2011, Freiburg, Germany, June, 2011.

[11] J. Aldinger, B. Nebel, Interval based relaxation heuristics for numeric planning with action costs, KI 2017: Advances in Artificial Intelligence, Cham, 2017.

[12] J. Frank , A. Jónsson, R. Morris, D. E. Smith, Planning and scheduling for fleets of earth observing satellites, Proceedings of Sixth Int. symp on Artificial Intelligence Robotics Automation & Space, 307, 2001.

[13] M. Wilson, T. Klos, C. Witteveen, B. Huisman. Flexibility and decoupling in Simple Temporal Networks, Artificial Intelligence, 214(9), 26-44, 2014.

[14] J. K. Mogali, S. F. Smith, Z. B. Rubinstein, Distributed decoupling of multi-agent simple temporal problems, International Joint Conference on Artificial Intelligence, 2016.

[15] C.Pralet, G. Verfaillie, Time–dependent Simple Temporal Networks: Properties and Algorithms, RAIRO - Operations Research, 47(2), 173-198, 2013.