A Model of Counterplanning with Variable Object Set

Ying Liu, Dantong Ouyang, Wenxiang Gu and Lei Wang

ABSTRACT

In recent years, the research for the counterplanning and the planning with variable object set has been regarded as difficult problem. In this paper, we effectively combine them under the graphplan framework, and provide a model of counterplanning with variable object set. It can create beneficial objects and delete useless objects, so as to achieve the goals of our agents and block or stop adversarial agents’ goals. The model can be widely used in intrusion detection system, network security, military area, commercial battlefield, and other fields.

INTRODUCTION

Graphplan[1,2] through planning graph analysis, is one of the most well-known methods, which attract much attention and many research studies have been carried out on it. Geib and Goldman in the United States at the DARPA DISCE (2001), pointed out that the intrusion detection system needs the technology of plan recognition to improve defending ability[3].

Plan recognition[4] is the one from the observed action of some agent and deduced the goals of the agent or the planning process. Adversarial plan recognition was first suggested by Geib and Goldman as an addition to the traditional models of keyhole and intended recognition. It has been also independently proposed by Jensen et al. for predicting the opponent’s moves in robotic games[5]. In 1981, Jaime G. Carbonell, a famous professor in Carnegie-Mellon University, developed an approach to reasoning about adversarial behaviour called “counterplanning” that maintains a strategic view of plan conflicts[6]. In 1990, Carol Applegate provided an architecture for adversarial planning[7].

DEFINITIONS AND CONCEPTS

The Definitions of Counterplanning

In the paper [8] we had given some definitions of counterplanning as follows:
Definition 1: In the execution environment, the agent’s goal is blocking our agent’s execution, and destroying our agent’s goal, called adversarial agent.

Definition 2: Adversarial agent for their own benefit to execute the plan, which can destroy our system parts or all, making them unable to run normally, even complete paralysis, such planning called adversarial plan.

Definition 3: Taking an action sequences to stop and destroy performance of the adversarial planning, for our system does not damage or loss to the minimum, such action sequences is called counter plan.

The Planning With Variable Object Set

As is known to all of us, many objects in the problems can be changed dynamically in our real life, such as the security of network, robot control, intelligent games, battle management and so on. This problem was proposed in 1995 by Blum and Furst, and it was partly solved until Wenxiang Gu and his students proposed the planning system CDOGP[9]. They tried to solve the planning problems involving finite object sets and predictable new objects. The main idea is to handle CDOP problems by using the graphplan. In 2009, they extend CDOGP to handle the probabilistic planning problems based on the creating or deleting objects and receive the good results[10,11]. Our paper is also based on graphplan and literature [9] to counter the adversarial plan in adversarial situations.

The Graph Of Counterplanning

McDermott D, et al. thought a plan is devising the sequence of actions for an agent[12].

Valid plan is a plan for a planning problem consists of a set of actions which are not mutex at the same step and specified time steps in which each is to be carried out. A valid plan must make all the problem goals true at the final time step.

Noop action is a special kind of action that does nothing to a proposition at next time step, whose add effects are the same propositions as its preconditions.

The graph of counterplanning is similar to the graphplan, which is a directed, levelled graph with two kinds of nodes which are proposition nodes and action nodes and three kinds of edges as well.

Reduce the Search Space

There are three mutual exclusion relations among planning graph nodes of literature [9]:

- Mutual exclusion relations among objects.
  At proposition level t, two objects of A and B are mutually exclusive in a planning graph, if no valid plan could possibly contain both A and B at that proposition-level.
- Mutual exclusion relations among action nodes.
  Two actions of A and B at action-level t are mutually exclusive if no valid plan could possibly contain both A and B at that action-level.
- Mutual exclusion relations among proposition nodes
  At proposition-level t, two propositions of P and Q are mutually exclusive in a planning graph if no valid plan could possibly contains both P and Q at that proposition-level.
In this paper, we provide a new definition that can find the planning solution more effectively.

**Definition 4**: In the process of searching a valid plan, if the delete effects of some actions delete the goals in the goal set, delete them directly.

**A MODEL OF COUNTERPLANNING**

The model of counterplanning is shown in Fig. 1.

![A model of counterplanning](image)

**Figure 1.** A model of counterplanning.

It is composed of five major components as follows:

- **Execution environment**
  We assume that the execution environment is completely can be observed. According to the direct observations, the agents execute the corresponding actions. To simulate the dynamic environment, and the agent’s reactions, it is necessary to model the decision of these agents. All of the message is from the execution environment, and after the counterplanning executed, the result of the counterplanner return to it.

- **Observer**
  Observer is applied to observe the change in the execution environment, the observed information send to the recognizer, which used to identify whether the action is hostile. At the same time, it send information to the strategic planning, to make it more early search the better strategy and respond timely.

- **Recognizer**
  Adversarial plan recognition is important for predicting intentions and future actions of attackers, recognizing unknown attacks, and planning appropriate responses. In many applications, such as computer security and information warfare, the purpose of adversarial plan recognition is to predict possible attacks in order to generate effective counterattacks. In such applications, one needs to know not only how his plans are going to be recognized by the adversary, but also how to recognize adversary plans in order to counterplan[13].

- **Strategic manager**
  Strategic decisions must be made quickly. Any delay could render a decision meaningless because the world could have changed in the meantime. The strategic manager includes a list of actions, temporal information which come from the observer, parameter bindings for variables in the actions (i.e., who, what, when, where), and preconditions of the actions [14]. According to the observed actions
and received information, to simulate the adversarial planning, search for effective tactics, combine it over to carry out the counterplanning effectively.

-Counterplanner

According to the strategy which is from the strategic manager, the counterplanner executes the counterplanning. If the counterplanner is successful, it send the results of counterplanning back to the execution environment, and continue to observe changes of the environment through the observer. If it is failure, send the results back to the strategic manager, and according to situation to choose the strategy again.

**ALGORITHM OF COUNTERPLANNING**

Our paper is based on the [9] to counter the adversarial plan in adversarial situations, and provide the algorithm of the counterplanning.

**Algorithm of Expanding Planning Graphs**

The algorithm of expanding planning graph includes three steps:

Step 1: Transforming the objects in the initial state into object-propositions. All of them are marked with “active”. Started with a plangraph, it only has a single proposition level which contains the initial conditions and object-propositions of initial objects. We assume all of the initial propositions are not mutually exclusive and the proposition-level has been obtained completely.

Step 2: If the level n contains all the goals and no pair of them is exclusive, searching valid plan, else if the proposition level n is exactly the same as the proposition level n-1 and mutex relations are the same, returns no valid plan, else turns to the step3.

Step 3: Instantiating all of the operators which can meet the preconditions to generate the next action-level. Insert all the no-op actions and add the precondition edges between the proposition-level and action-level. Then, we identify the exclusion relation between actions. To create a generic proposition level, we add all the effects of the actions in the previous level, and transform the new objects created by the action in previous level to object-propositions. Then, place them in the next level as proposition-level and connect them via the appropriate add and delete edges. Last, we identify the exclusion relation between propositions.

**The Algorithm of Searching a Valid Plan**

The algorithm for searching a valid plan uses a backward-chaining strategy, like graphplan, in order to make use of the mutual exclusion constraints. It includes two steps as follows:

Step 1: For each goal at time t, select some actions at time t-1 achieving these goals that are not exclusive of any action that has been selected. If some actions delete the goals in goal set, delete them directly. Continue recursively with the next goal at time t. If the recursive call returns failure, try a different action achieving the current goal and so forth. It returns failure once all the actions have been tried. When finishing with all the goals at time t, the preconditions of the selected actions make up the new goal set at time t-1, and then it continues this procedure at time t-1.

Step 2: If the goals at the current level are a subset of the initial conditions, we get the valid plan that is the action set having been selected. If it reruns failure at
the previous level, back up right way. If it still returns failure at the last level t, there is no valid plan, and then go to graph expansion for replanning.

AN EXAMPLE FOR COUNTERPLANNING

Description of the Battle Problem

This is a very simple battle problem. Initially, there is an army A fight to the enemy. In order to win the battle, they carried out the strategic movement. In this example the army is divided into two parts. The purpose what they do is that one part fight continue and stay to hamper the enemy, and the other part around to the back of enemy, then realize the two sides flanked. The given operators are move (move the army) and divide (divide the army). The goal is that active A1 is front of the enemy; Active A2 is behind them. The problem is described as follow:

Initial conditions:{ at A F}
Operators:{move, divide}
Goals:{at active A1 F, at active A2 B}
The definitions of the actions:

(define (operator move)
 :parameters((Army ?A) (place ? from) (place ? to))
 :precondition(: and (:neq ? to ? from) (at ?A from)(active ?A))
 :effect(:and (at ?A ?to )(:not at ?A ?from)))

(define (operator divide)
 :parameters((Army ?A) (Front ?F) (Behind ?B))
 :precondition(: (active ?A ))
 :effect(when( at ?A ?F)
 :and((at ?active A1 ?F)(at ?active A2 ?F)(active ?A1)(active ?A2)(not ?active A))
 when(at ?A ?B)
 :and((at ?active A1 ?B)(at ?active A2 ?B)(active ?A1)(active ?A2) (not ?active A)))
Solving a Problem Using Counterplanning with Variable Object Set

We use the graph expansion algorithm to obtain the plan graph is shown in Fig. 2.

At the first step, the proposition-level 1 is obtained by the initial conditions and object-propositions of initial objects. Because the goal doesn’t appear in this level, we expand the plan graph to level 2. At this step we can instantiate the operators: move and divide, and add their add effects and delete effects in the next proposition-level, and mark the mutual exclusion relations among them. In the proposition-level 2, we don’t get the goal, so expand the graph go on until the goal appears at proposition-level 3, and then searching for valid plan.

The goals of this problem are \{At A1 F\} and \{At A2 B\}. In the searching for valid plan, we find the action(Move A1) is delete the goal \{At A1 F\}, according to the definition 4 and the mutual exclusion relations, we select the noop action to support the goal, instead of the action(Move A1). And the preconditions of them form a set of subgoals\{At A1 F, Active A1, At A2 F\}. The subgoals’ support action is divided, and its preconditions are the subset of initial conditions in level 1. Then,

![Graph of counterplanning for the battle problem](image)

**Figure 2.** A graph of counterplanning for the battle problem.

we can get a valid plan as follows:
Time step1: Divide A;
Time step2: Move A2.

In this graph, Active A1 and Active A2 are the newly created objects, and the Active A is the deleted object.

CONCLUSIONS

In this paper, we provide a model of counterplanning, and give an algorithm of counterplanning with variable object set. The model can observe the other agent’s actions, analyse the goal of opposite side, simulate adversarial planning, and firmly carry out the counterplanning. It can be widely used in intrusion detection system, network security, man-machine competition, military area, commercial battlefield, and other fields.
This research was supported by the "Twelfth five-year" science and technology research of education department of Jilin under Project Number: [2015] no. 572 and [2015] no.563.

REFERENCES

[1] Blum and M. Furst.: Fast planning through planning graph analysis. In Proc.14th Int. Joint Conf. AI, (1995)1636-1642.
[2] Blum and M. Furst.: Fast planning through planning graph analysis. Artificial Intelligence, 90 (1997)281-300.
[3] C. Geib and R. Goldman.: Plan recognition in intrusion detection systems. in DARPA Information Survivability Conference and Exposition (DISCEX II’01), (2001) 46-55.
[4] Wenxiang Gu, Minghao Yin, Li Xu, et al.: Intelligent Planning and Plan Recognition [M]. Beijing: Science Press, (2010)173-174.
[5] R. Jensen, M. Veloso, and M. Bowling.: Obdd-based optimistic and strong cyclic adversarial planning. In Proceedings of ECP, (2001).
[6] J.G. Carbonell.: Counterplanning: A strategy -based model of adversary planning in real-world situations. Artificial Intelligence, 16 (1981)295-329.
[7] C. Applegate, C. Elsaesser, D. Sanborn: An architecture for adversarial planning. IEEE Trans. Systems Man Cybernet. 20 (1990) 186-294.
[8] Ying Liu, Tiejun Shao,Wenxiang Gu.: Research on a response planner model. Journal of northeast normal university. 40(2008)30-33.
[9] Wenxiang Gu, Zengyu Cai, Xinmei Zhang.: Extending graphplan to handle the creating or destroying objects panning. ICMIC2005. Guangzhou, China (2005).
[10]Wen-xiang Gu, Chang-Fang Shao, Ming-Hao Yin, Jin-Li Li, Bing-Li.: Probabilistic planning for creating or destroying objects. IJCAI (2009).
[11]Wenxiang Gu, Ying Liu: Recognition and Opposition for Adversarial Planning [M]. Beijing: Science Press, (2016)234-240.
[12]McDermott D, et al.: Planning: What is, What it could be, An introduction to the Special Issue on Planning and Scheduling. Artificial Intelligence, (1995)1-16.
[13]Braynov, S.: Adversarial planning and plan recognition: Two sides of the same coin. Secure Knowledge Management Workshop, New York. (2006a).
[14]W. Swartout, DARPA workshop on planning, AlMag. 9(1988).