Journal of Software Engineering and Applications, 2011, 4, 81-85
doi:10.4236/jsea.2011.42009 Published Online February 2011 (http://www.SciRP.org/journal/jsea)

RPBTC: An Implementation Method for Robot Planning

Shan Zhong, Qiuya Chen, Yunfei Zhou, Juan Yuan

Computer Science and Engineering College, Changshu Institute of Technology, Changshu, China.
Email: sunshine620@cslg.edu.cn

Received January 4th, 2011; revised January 13th, 2011; accepted January 20th, 2011.

ABSTRACT

Aiming at the former formalized methods such as Strips, Situation Calculus and Fluent Calculus can not represent the action time and get the action sequence automatically, a novel method based on timed color Petri net—RPBTC was defined. The action time, the precondition and the post-condition of action are formalized in RPBTC based on the Fluent Calculus reasoning rules. An algorism for constructing the RPBTC net system based on bidirectional search strategy is proposed, and through executing the RPBTC net system, the action sequence for reaching the goal can be generated dynamically and the time for the robot reaching the goal also can be obtained. The experiment has proved the method RPBTC as a feasible method for robot planning.

Keywords: *Timed Petri Net, Robot Planning, Reasoning, Action, Goal*

1. Introduction

Reasoning about action [1] of robot is one of the most important areas of Artificial Intelligence. Reasoning about action is mainly to construct the action sequence according to the specific problem, namely, given the initial state, the goal state and the actions robot can do, describing how the robot constructs the action sequence to reach the goal.

Petri net has the strict math model, and it is special suitable for the representation and simulation of the systems characteristic of synchronism, concurrency and dynamics [2-3]. Vittorio [4] introduced a method for the robot planning based on Petri net, formally describing actions and the relations between the actions, but it lacks the formal description for the state including the environment state and robot state. MA Bingxian [5] describes the inner action and the external action, and using two algorisms to reason the action sequence, but can not obtain the action automatically in the constructed hierarchical Petri net system for some given goal.

Therefore, a model for using the timed colored Petri net to represent robot planning based on Fluent Calculus [6-7] rules is introduced, the two axioms such as precondition axiom and state update axiom are converted to the corresponding timed transitions, the places and the arcs between the timed transitions and the places, a bidirectional search strategy is introduced applying to the constructing algorism of RPBTC net system, which is in order to improve the efficiency for robot reasoning and generating the action sequences automatically. Finally, the simulation experiment proves the RPBTC net system is feasible method for robot to reason about action in dynamic environment.

2. Fluent Calculus

As for the Fluent Calculus, all changes to the world can be the result of named actions. A possible state history is a sequence of actions to reach the goal called situation. Fluent is for representation of the atomic properties of the physical world. The function State(s) denotes the state in situation s, which links the two key notions named state and situation.

Precondition axioms are used to formally specify the circumstances under which an action is possible in a state in situation *s*, denoted by predicate *Poss*(*a*, *state*(*s*)) showed in Equation (1), which means at the state *state*(*s*), the action *A*(*x*) can be executed. The $\Pi_A(\bar{x}, z)$ as a pure state axiom about *z* is the conjunction of the conditions under which the action can be happened.

$$Poss\left(A\left(\bar{x}\right)\right) \equiv \Pi_A\left(\bar{x}, z\right) \quad (1)$$

State update axioms defines the effects of an action a as the difference between the state prior to the action, *State*(*s*), and the successor *State*(Do(*a*,*s*)), showed in Equation (2), in which $\Delta(\bar{x}, z)$ is the pure state axiom representing the current state, v^+ (the positive effect) represents the adding states, while v^- (the negative effect) represents the removed states, and *state*(*s*) represents the unchanged states and the unknown states.

$$Poss\left(A(\bar{x}),s\right)\wedge\Delta\left(\bar{x},State(s)\right)\supset$$
$$State\Big(Do\left(A(\bar{x}),s\right)\Big)=State(s)\circ v^+-v^- \qquad (2)$$

3. RPBTC Net System

Definition 1 A RPBTC net system is a 7-tuple system $\sum$ = (*P*, T_v; *F*, *C*, *I*-, *I*+, M_0), which is a timed colored petri net system has the following characteristics:

1)The place $p_i \in P$ represents the states including both the state of robot and the environment state.

2)The timed transition $t \in T_v$ represents the actions which the robot has the ability to do. Any action has the corresponding timed transition.

3) For any arc $f \in F$, the value on the arc such as $I_-(f)$ and $I_+(f)$ defines the change rules between the place and the transition, the specific value is decided by the relations between the action and the precondition of the action, and also the relations between the action and the successor states of the action.

4) *C* is the colored set, defining the possible token color in the place and the appearance color in the transition.

5) M_0 as the initial state represents the initial state of the agent and the environment.

Definition 2 The corresponding relations between the precondition axiom of Fluent Calculus and the RPBTC net is showed as follows: the preconditions of action *A*(*x*) are represented by places, the specific value of the state is represented as the token in the place, the executing time for the action is using the @ + *t* attached to the transition.

Definition 3 The relation between state update axiom and the RPBTC net is showed as follows: the representation of the action is showed in the definition 2, the successor state of the actions such as $State(s)\circ v^+-v^-$ are using the place to represent, the specific value in the place is represented by specific token, v^+ denotes the add state, the specified value of v^+ can be represented by token from the transition to the place, v^- denotes the removed the state from the place to the transition. *State*(*s*) represents the unchanged state, namely, the representation of the framework problem, the token from the place goes back to the same place.

Definition 4 Given a action sequence $\sigma=\{t_1,\cdots,t_n\}$ for a robot to reach the goal, then the backward action sequence can be represented as $\sigma=\{t_n,\cdots,t_1\}$.

Definition 5 The bidirectional search strategy is the combination of the forward search strategy and the backward search strategy, the action sequence from the initial state M_0 to some middle state Mmid is $\sigma=\{t_1,\cdots,t_i\}$, and the action sequence from the goal state M_g to the same middle state M_{mid} is $\sigma=\{t_n,\cdots,t_i\}$, then the forward search action sequence σ is connected to the reversed backward search action sequence, finally, the repeated action sequence for reaching the same middle state is removed, then the action sequence for the bidirectional search strategy is as: $\sigma_{goal}=\{t_1,\cdots,t_n\}$.

4. The Constructing Algorism for RPBTC Net System

Given the initial state, the goal state of the system and the actions robot can do, a bidirectional RPBTC net system can be constructed using the bidirectional search strategy in definition 5, in which a action in the RPBTC net system has both the forward transition t and backward transition $t_{\_Reverse}$. As for the place representing robot state, there are two places such as *pi* and $p_{i\_reverse}$, representing the forward search place and backward search place respectively. As for the representation of the environment state, there is only one place p_k in the forward search and backward search processes, finally, the RPBTC net system can be constructed.

The input of algorism: the place set *Pre* in the precondition axiom and the place set Post of successor state update axioms, the initial state and the goal state. The output of algorism: the bidirectional RPBTC net system for robot planning. *Pre* is used to represent the place set of precondition axioms of specific planning instance, and *Post* is used to represent the place set of successor states.

Step 1 Define *P*, T_v, *F*, *C*, I_-, I_+, M_0 as set, *P* is the collection of places, $p_{Action\ Sequences} \in P$ stores action sequence, $p_{Goal} \in P$ stores the goal of agent, *T* is the collection of timed transition, *F* is the collection of arc, I_- and I_+ are the arc set, M_0 is the initial marking converting from initial state, for the convenient, the subset *F1* of *F* is defined.

Define *i* as a counter, the initial value of *i* is 1

$$P = P \cup p_{Action\ Sequences} \cup p_{Goal}$$

Step 2 Loop the implementation of the following part until all the preconditions and successor states of the action have already been constructed, and when the i-th action is executing,

$T = T \cup \{t_i\} \cup \{t_{i\_Reverse}\}$; //Action and its executing time is represented by transition adding @+

$$F1 = F \cup \left\{\begin{array}{l}\left(p_{ik,}t_i\right),\left(t_i,p_{Action\ Sequences}\right),\left(p_{Action\ Sequences},t_i\right),\\ \left(t_i,p_{Action\ Sequences}\right),\left(t_i,p_{Goal}\right),\left(p_{Goal},t_i\right),\\ \left(p_{ik},t_{i\_Reverse}\right),\left(t_{i\_Reverse},p_{Action\ Sequences}\right),\\ \left(p_{Action\ Sequences},t_{i\_Reverse}\right)\end{array}\right\}$$

```
if $p_{ik}$ appears in the precondition place set $Pre$ and not in the successor state update place set $Post$
  if $p_{ik}$ is a environment state then
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik}$;
      $F = F1$;
  else if $p_{ik}$ is the state place of robot then
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik} \cup p_{ik\_Reverse}$;
      $F = F1 \cup \{p_{ik\_Reverse}, t_{i\_Reverse}\}$;
  end if
if $p_{ik}$ is an environment place then
  if $p_{ik}$ appears in the post conditions set $Post$ and not in the precondition place set $Pre$
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik}$;
      $F = F1 \cup \{(t_{i\_Reverse}, p_{ik})\} - \{(p_{ik}, t_{i\_Reverse})\}$;
  else if $p_{ik}$ is the state place of robot then
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik} \cup p_{ik\_Reverse}$;
      $F = F1 \cup \{(t_{i\_Reverse}, p_{ik})\} \cup \{(p_{ik\_Reverse}, t_{i\_Reverse})\} - \{(p_{ik}, t_{i\_Reverse})\}$;
   end if
else if $p_{ik}$ appears both in the precondition place set $Pre$ and in the successor state place set $Post$
  if $p_{ik}$ is the environment state place then
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik}$;
      $F = F1 \cup \{(t_{i\_Reverse}, p_{ik})\}$;
  else if $p_{ik}$ is the state place of robot then
      $P = P \cup p_{Action\ Sequences} \cup p_{Goal} \cup p_{ik} \cup p_{ik\_Reverse}$;
      $F = F1 \cup \{(t_{i\_Reverse}, p_{ik})\} \cup \{(p_{ik\_Reverse}, t_{i\_Reverse})\}$;
  end if
end if
```

Step 3 $i := i + 1$, if all the actions have been iterated for once, then go to the step 4, else go the step 2

Step 4 After the precondition place set and the successor place set have been constructed, according to the initial states of system, the token is added to the relative place then the initial marking M_0 is obtained. In the following part, the RPBTC model of the entire system will be constructed in the example.

4.1. Simulation Experiment

The robot example is showed as **Figure 1**. The round represents the robot. There are four rooms such as R401, R402, R403, R404, and alley. Every neighbor room is connected by a door, and the door also connects the neighbor room and alley, for example, D34 connects room R304 and room R303, and DA3 connects Alley and room R303. The goal of the robot is reaching the goal room from the initial room, and for reaching the given goal, the robot have some actions that he can do. The actions which the robot can do are as follows:

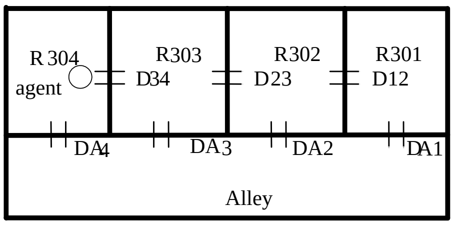

Figure 1. Robot planning in office scene.

1) The action *Go*(*x*) means robot goes to the door *x*. The executing time for the action is 2 units. The precondition axioms and the state update axioms will be described showed in Equations (3) and (4)

$$Poss(Go(x), z) \triangleq Position\_start(r, x') \wedge Connects(x', r, \_) \quad (3)$$

$$\begin{aligned} &Poss(Go(x), s) \wedge \text{Holds}\ (Position(r, x'), \text{s}) \\ &\supset State(Do(Go(x), s)) \triangleq Connects(x', r, \_) \\ &\circ Position\_start(r, x) - Position(r, x') \end{aligned} \quad (4)$$

2) The action *Enter*(*r*) represents the robot go to the room *r*, the executing time for the action is 1 unit, The precondition axiom and the state update axioms are showed as Equation (5) and Equation (6):

$$\begin{aligned} &Poss(Enter(r), z) \triangleq Connects(x, r, r') \\ &\wedge Position\_start(r', x) \wedge Closed(x) \end{aligned} \quad (5)$$

$$\begin{aligned} &Poss(Enter(r), s) \wedge \text{Holds}\ (Position(r', x), \text{s}) \\ &\supset State(Do(Enter(r), s)) \triangleq Connects(x, r, r') \\ &\wedge Closed(x) \circ Position\_start(r, x) \\ &-Position\_start(r', x) \end{aligned} \quad (6)$$

3) The action *Open*(*x*) represents opening a door, and the executing time for the action is 1 unit. The precondition axiom and the state update axiom are showed as Equation (7) and Equation (8), respectively.

$$\begin{aligned} &Poss(Open(x), z) \triangleq Closed(x) \\ &\wedge Position\_start(r, x) \wedge Key(x) \end{aligned} \quad (7)$$

$$\begin{aligned} &Poss(Open(x), s) \wedge \text{Holds}\ (Closed(x), s) \\ &\supset State(Do(Open, s)) \triangleq Position\_start(r, x) \\ &\wedge Key(x) \circ Closed(x) - Closed(x) \end{aligned} \quad (8)$$

4.2. The Constructing RPBTC Net System for Office Robot Example

The initial state for the robot is that the robot is at the door “d34” of the room “304”, it owns the key of door “d34”, door “dac” and door “da4”. The door “d34” is closed. The connecting state between the door and the room is showed in **Figure 1**, using the place *Position_start*, *Key*, *Closed* and *Connects* to represent the initial state are as follows: M_0 = {*Position_start* (“304”,“d34”), *Key*{“d34”}, *Closed*{“dac”,“da4”,“d34”}, *Connects*{(“da1”,“301”, “alley”), (“da2”,“302”,“alley”), (“d23”,“302”,“303”), (“d34”,“303”,“304”), (“da2”,“alley”,“302”), (“da3”,“alley”,“303”), (“da4”,“alley”,“304”)}}.

The goal for robot is in room “301”, so the four places to represent the goal marking is: M_g = {*Position_start* (“301”,_), D, D, D }, in which D is any value. “_” is a variable, and when it is appointed to “d12”, then the place *Goal_Reverse_Position* represents the token value.

Using the constructing algorism of bidirectional search RPBTC net system for the office instance, the modeling figure is showed in **Figure 2** using CPNTools [8].

The simulation result is showed in **Figure 2**. The forward action sequence is showed in place *Action Sequeces*, and the backward action sequence is showed in the place *Action Sequence Reverse*.

The forward search action sequence in place *Action Sequence* is showed as: {open(d34), enter(303), go(da3), enter(alley)}. The place *Action Sequence Reverse* contains the backward search action sequence as follows:

{enter (301), go (da1), enterReverse (alley), go_reverse (da3)}.

After the reversed backward action sequence and the forward search action sequence are connected, then according the definition 3, the final action sequence is showed as follows: {open(d34), enter(303), go(da3), enter(alley), go(da1), enter(301)}

The action sequence represents a successful plan from the initial state to the goal state, and the left “time” marked the executing time of the action sequence is 8 units, which is the overall time for robot to reach the goal.

5. Conclusions

The precondition and state update axioms of Fluent Calculus are used in this paper, compared with Fluent Cal-

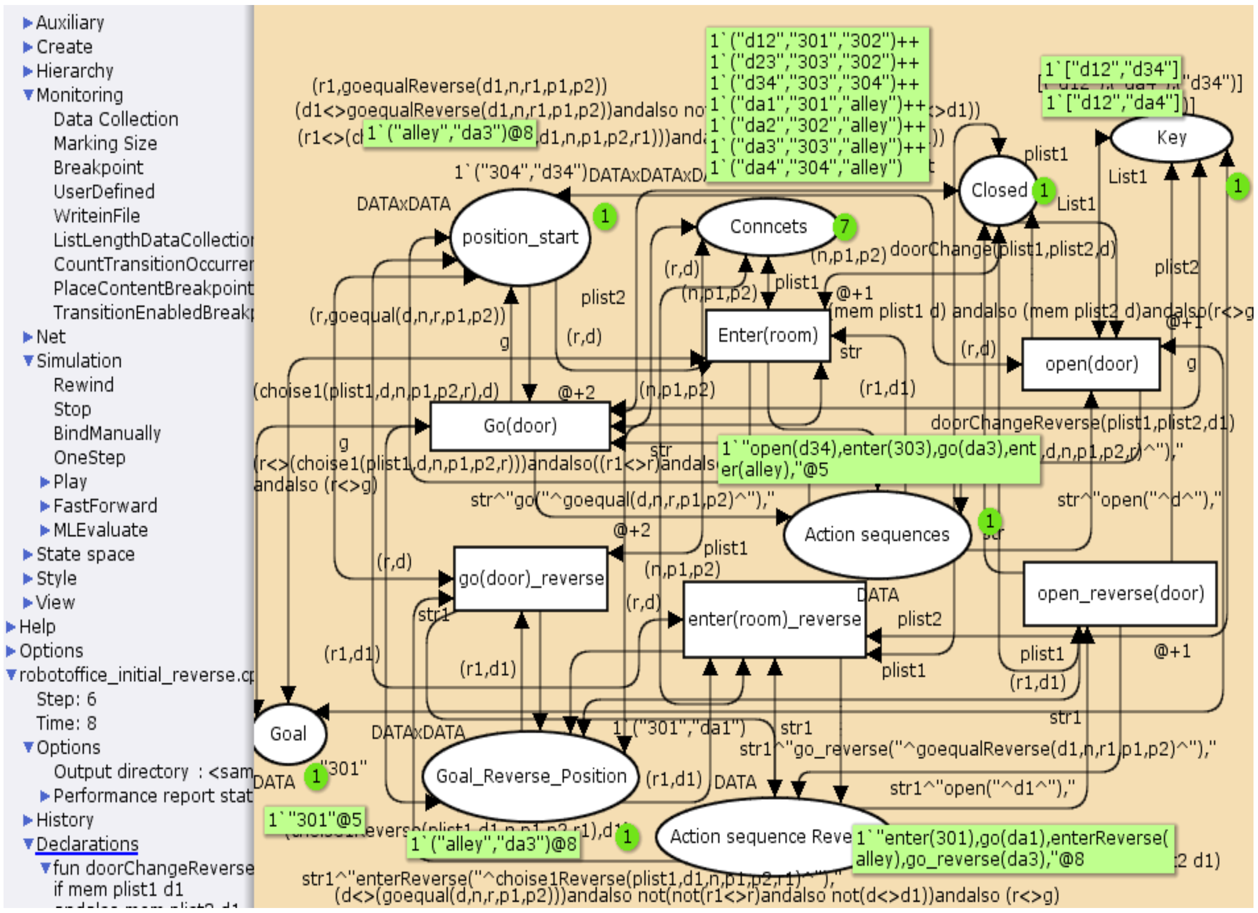

Figure 2. The simulation figure for the office scene.

culus, RPBTC net system add the representation for the action time, can satisfy the high demand for the real time, secondly, Fluent Calculus using the directional search strategy and backdate mechanism to reach the goal, but RPBTC net system use the bidirectional strategy, so it has the higher efficiency. Finally, RPBTC net system can dynamically and automatically simulate the action sequence reaching the goal, it also can according the time demand to choose the best action sequence. The method in this paper has improved the efficiency and timing.

The next work is to use hierarchical petri net to represent and implement robot planning, to avoid the problem of state explosion problem when the system scale and state become larger.

REFERENCES

[1] J. McCarthy and P. J. Hayes, "Haves Some Philosophical Problems from the Standpoint of Artificial Intelligence," B. Meltzer and D. Michie, Eds., *Machine Intelligence*, Vol. 4, 1969, pp. 463-502.

[2] K. Jensen, L. M. Kristensen and L. Wells. "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems," *Software Tools for Technology Transfer*, Vol. 9, No. 3-4, 2007, pp. 213-254. doi:10.1007/s10009-007-0038-x

[3] P. F. Palamara, V. A. Ziparo, L. Iocchi, *et al.*, "A Robotic Soccer Passing Task Using Petri Net Plans," *Proceedings of 7th International Conference on Autonomous Agents and Multiagent Systems*, Estoril, 12-16 May 2008, pp. 1711-1712.

[4] V. A. Ziparo and L. Iocchi, "Petri Net Plans," *Fourth International Workshop on Modelling of Objects, Components, and Agents*, Turku, 26 June 2006, pp. 267-290.

[5] B. X. Ma, Z. H. Wu and Y. L. Xu, "Research on Multi-Agent Planning with Petri Nets," *Computer Engineering*, In Chinese, Vol. 32, No. 14, 2006, pp. 4-6.

[6] Y. Jin and M. Thielscher, "Iterated Belief Revision, Revised," *Artificial Intelligence*, Vol. 171, No. 1, 2007, pp. 1-18. doi:10.1016/j.artint.2006.11.002

[7] S. Sardina, G. de Giacomo, Y. Lesperance, *et al*., "On the Semantics of Deliberation in Indigolog-From Theory to Implementation," *Annals of Mathematics and Artificial Intelligence*, Vol. 41, No. 2-4, 2004, pp. 259-299. doi:10.1023/B:AMAI.0000031197.13122.aa

[8] CPN Tools. http://www.daimi.au.dk/CPNTools/