ISBN 978-981-14-1684-2
Proceedings of 2019 the 9th International Workshop on Computer Science and Engineering
(WCSE 2019 SUMMER)
Hong Kong, 15-17 June, 2019, pp. 970-976
doi: 10.18178/wcse.2019.06.145

A Control Strategy Algorithm for Finite Alternating Transition Systems

Jinjin Zhang [1,2,+] and Yan Zhang [3]

[1] Department of Computer Science, Nanjing Audit University, Nanjing, P. R. China
[2] Jiangsu Key Laboratory of Audit Information Engineering
[3]Department of Computer Science, Nanjing Forestry University, Nanjing, P. R. China

Abstract. Recently, there has been an increasing interest in the formal analysis and design of control systems. In this area, in order to reduce the complexity and scale of control systems, finite abstractions of control systems are introduced and explored. Amongst, Pola and Tabuada construct finite alternating transition systems as approximate finite abstractions for control systems with disturbance inputs [*SIAM Journal on Control and Optimization*, Vol. 48, 2009, 719-733]. Given linear temporal logical formulas as specifications, this paper provides a control strategy algorithm to find control strategies of Pola and Tabuada's abstractions enforcing specifications.

Keywords: alternating transition systems, finite abstraction, linear temporal logic, control strategy algorithm.

1. Introduction

The formal analysis and design of control systems is one of recent trends in control theory. The formal analysis is concerned with verifying whether a control system satisfies a desired specification, while the purpose of the formal design is to construct a controller for control system so that it meets a given specification. Traditionally, stability and reachability are considered as specifications in the control-theoretic community[1]. Recently, there has been an increasing interest in extending the formal analysis and design by considering more complex specifications [2-8]. In these work, temporal logic [2-5,7,9], regular expressions [6], and transition systems [8] are used to describe specifications. Amongst, temporal logic, due to its resemblance to natural language and the existence of algorithms for model checking, is widely adopted for task specification and controller synthesis in control theory. For example, linear temporal logic (LTL) has been adopted to describe the desired properties of discrete-time linear systems [7] and continuous-time linear systems [9]. In addition, Computation Tree Logic (CTL)[3] and LTL[4,5] are applied to express specifications in the area of mobile robotics.

The formal analysis and design of large-scale control systems is difficult because of the complexity and scale of systems. In order to reduce the complexity and scale, finite abstractions are extracted from these control systems [2,7,8]. Usually, finite abstractions and original systems share properties of interest and the analysis and design of finite abstractions is simpler than that of original control systems. Thus the analysis and design of control systems is often equivalently performed on the corresponding finite abstractions. So finite abstractions are extremely useful in the formal analysis and design.

Much work has been devoted to the construction of finite abstractions of control systems. For instance, Tabuada and Pappas identify critical properties of discrete-time linear systems ensuring the existence of

[+] Corresponding author. Tel.: +86 25 58318607; fax: +86 25 58318604.
E-mail address: jinjinzhang@nau.edu.cn.

finite abstractions [10]. Symbolic models of nonlinear control systems are constructed in [11]. Finite abstractions of hybrid systems are studied in [12,13]. An excellent review of these work may be found in [2].

In the work mentioned above, researchers consider control systems without reference to disturbances. However, as pointed out by B C. Kuo in [14], all physical systems are subject to some types of extraneous disturbances or noise during operation. Recently, Pola and Tabuada extend the above work to control systems affected by disturbances [15,16]. A mathematical structure called alternating transition system is presented as symbolic abstraction of control system with disturbance inputs [15,16]. Under the assumption that control systems are bounded, such abstractions are finite.

In [5,7,8], usual transition systems are adopted as finite abstractions of control systems. Some approaches are presented to construct control strategies of these finite abstractions enforcing specifications. Further, based on such control strategies, controllers of original control systems are generated to meet specifications. So the construction of control strategies of finite abstractions is one of the important steps in the formal design of control systems. However, since Pola and Tabuada's abstractions [15,16] are modeled by alternating transition systems rather than usual transition systems, the approaches provided in [5,7,8] are not suitable for establishing control strategies for Pola and Tabuada's abstractions. To overcome this defect, this paper will present a control strategy algorithm based on Kabanza et al.'s planning algorithm [18] to solve the following control problem: given a finite, non-blocking alternating transition system T and a specification, how to find an initial state and a control strategy of T enforcing the given specification? Clearly, this algorithm can be used to find control strategies for Pola and Tabuada's finite abstractions.

The rest of this paper is organized as follows. In Section 2, we recall the notion of alternating transition system and present the control problem mentioned above in detail. Section 3 recalls some notions and results about Kabanza et al.'s planning algorithm. Based on their algorithm, Section 4 provides a control strategy algorithm and explores the correctness and completeness of this algorithm. Finally, we conclude the paper with future work in Section 5. Because of limitations of space, the proofs of theorems are not provided.

2. Alternating Transition System and Control Problem

Before recalling the notion of alternating transition system, we introduce some useful notations. The symbol $\mathbb{N}$ denotes the set of positive integers.For any set A, A^+ denotes the set of all non-empty finite strings over A, and A^ω represents the set of infinite strings over A. Usually, we put $A^\infty = A^+ \cup A^\omega$. We use $s_{\{A\}}$, σ_A and α_A to denote the elements of A^+, A^ω and A^∞, respectively. If A is known from the context, we will omit the subscript in s_A, σ_A and α_A. For any $s \in A^+$, $s[i]$ and $s[end]$ mean the i-th element and the last element of s, respectively. Given $i \le j$, $s[i,j]$, $s[i,end]$ and $\sigma[i,\infty]$ represent $s[i]s[i+1]\cdots s[j]$, $s[i]s[i+1]\cdots s[end]$ and $\sigma[i]\sigma[i+1]\cdots$, respectively. As usual, $|s|$ means the length of s. For any $\sigma \in A^\omega$, $|\sigma|$ is set to be ∞.

Pola and Tabuada adopt alternating transition systems as models of these control systems with disturbance inputs and their finite abstract systems [15,16].

Definition 1. *An alternating transition system is a tuple $T=(Q,A,B,\rightarrow,O,H)$ consisting of a set of states Q, a set of control labels A, a set of disturbance labels B, a transition relation $\rightarrow \subseteq Q \times A \times B \times Q$, an observation set O, and an observation function $H:Q \rightarrow O$. An alternating transition system is said to be finite if Q, A and B are finite, and is said to be non-blocking if $\{q' : q \xrightarrow{a,b} q'\} \neq \varnothing$ for any $q \in Q$, $a \in A$ and $b \in B$. An infinite sequence $\sigma \in Q^\omega$ is said to be a trajectory of T if and only if for all $i \in \mathbb{N}$ $\sigma[i] \xrightarrow{a_i,b_i} \sigma[i+1]$ for some $a_i \in A$ and $b_i \in B$.*

In the above definition, a transition label is a pair <*a,b*>, where the former is used to denote control input and the latter represents disturbance input. This paper aims to provide an approach to obtain control strategies of Pola and Tabuada's finite abstractions to meet specifications. Formally, we will solve the following control problem:

Problem 1. *Given a finite, non-blocking alternating transition system T and a specification, how to find an initial state and a control strategy of T enforcing the given specification?*

In this paper, the specifications mentioned above will be described by the linear temporal logic $\mathrm{LTL}_{-\mathrm{X}}$ [17]. We recall this logic below.

Definition 2. *Let $\mathbb{P}$ be a finite set of atomic propositions. The linear temporal logic LTL-X($\mathbb{P}$)formula over $\mathbb{P}$ is inductively defined as: $\varphi ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathbf{U} \varphi_2$, where $p \in \mathbb{P}$.*

The operator $\mathbf{U}$ is read as "until" and the formula $\varphi_1 \mathbf{U} \varphi_2$ specifies that φ_1 must hold until φ_2 holds. The semantics of LTL-X($\mathbb{P}$) formulae are defined below.

Definition 3. *Let $\sigma_{\mathbb{P}}$ be any infinite word over $2^{\mathbb{P}}$. The satisfaction of LTL-X($\mathbb{P}$) formula φ at position $i \in \mathbb{N}$ of the word $\sigma_{\mathbb{P}}$, denoted by $\sigma_{\mathbb{P}}[i] \models \varphi$, is defined inductively as follows:*

(1) $\sigma_{\mathbb{P}}[i] \models p$ iff $p \in \sigma_{\mathbb{P}}[i]$;

(2) $\sigma_{\mathbb{P}}[i] \models \neg\varphi$ iff $\sigma_{\mathbb{P}}[i] \models \varphi$ does not hold;

(3) $\sigma_{\mathbb{P}}[i] \models \varphi_1 \wedge \varphi_2$ iff $\sigma_{\mathbb{P}}[i] \models \varphi_1$ and $\sigma_{\mathbb{P}}[i] \models \varphi_2$;

(4) $\sigma_{\mathbb{P}}[i] \models \varphi_1 \mathbf{U} \varphi_2$ iff there is $j \geq i$ s.t. $\sigma_{\mathbb{P}}[j] \models \varphi_2$ and for all $k \in \mathbb{N}$ with $i \leq k < j$, we have $\sigma_{\mathbb{P}}[k] \models \varphi_1$.

A word $\sigma_{\mathbb{P}}$ satisfies an LTL-X($\mathbb{P}$) formula φ, written as $\sigma_{\mathbb{P}} \models \varphi$, if and only if $\sigma_{\mathbb{P}}[1] \models \varphi$.

Definition 4. *Let $T=(Q,A,B,\rightarrow,O,H)$ be a finite, non-blocking alternating transition system, $\mathbb{P}$ a finite set of atomic propositions and let $\Pi: Q \rightarrow 2^{\mathbb{P}}$ be a valuation function. For any LTL $L_{\text{-X}}(\mathbb{P})$ formula ψ, an infinite sequence $\sigma \in Q^{\omega}$ is said to satisfy ψ w.r.t Π, written as $\sigma \models_{\Pi} \psi$, if and only if $\Pi(\sigma) \models \psi$, where $\Pi(\sigma) = \Pi(\sigma[1])\Pi(\sigma[2])$....*

If the valuation function Π is known from the context, we often omit the subscript in $\models_{\Pi}$.

3. Kabanza et al.'s Algorithm

To solve Problem 1, we will provide a control strategy algorithm based on Kabanza et al.'s planning algorithm. This section recalls some notions and results about Kabanza et al.'s algorithm. More details can be found in [18].

Kabanza et al. develop their work in the framework of reactive agent. Given a finite set Q of world states, a *reactive agent* is described as a pair $(q_0, succ)$, where $q_0 \in Q$ is an initial world state and *succ* is a transition function. For any world state $q \in Q$, *succ(q)* returns a list $((a_1,d_1,W_1),\cdots,(a_n,d_n,W_n))$, where a_i is an action that is executable in q, d_i is a strictly positive real number denoting the duration of a_i in q, and $W_i \subseteq Q$ is the set of nondeterministic successors resulting from the execution of a_i in q. As usual, if $q' \in W_i$ for some $i \leq n$, then we denote by $q \xrightarrow{a_i} q'$ that q' is a successor of q resulting from the execution of a_i in q.

Definition 5. *A reactive plan is represented by a set of situation control rules (SCRs), where an SCR is a tuple of the form (n,q,a,N) such that n is a number denoting a plan state, q is the world state labeling the plan state n and describing the situation when this SCR is applied, a is the action to be executed in plan state n, and N is a set of integers denoting plan states that are nondeterministic successors of n when a is executed.*

That is, For any q' with $q \xrightarrow{a} q'$, there must be $j \in N$ such that the corresponding world state of plan state j is q'. In the above definition, two kinds of states are referred to: world states and plan states. Each plan state is labeled by a world state and different plan states may be labeled by the same world state. Roughly speaking, these plan states labeled by the same world state q may denote different executive pathes along which the world state q is reached. So, since the actions to be executed in different plan states may not be identical, the choice of the actions in the world state q can be history dependent.

We start the execution of a reactive plan by fetching the SCR corresponding to the initial world state. By convention, this is always the SCR with plan state 1. The corresponding world state describes the current situation before the agent executes any action. At any time, given the current SCR (n,q,a,N), the action a is executed and the SCR matching the resulting situation is determined from the successor plan states in N by getting an SCR (n',q',a',N') such that $n' \in N$. In this case, the current situation is q' and then a' is executed.

Kabanza et al. provide an planning algorithm to construct a reactive plan satisfying an MTL($\mathbb{P}$) formula ψ for the given reactive agent and valuation function Π. The detailed algorithm may be found in [18]. According to Theorem 16 and the observation in Section 7.5 in [18], we have

Theorem 1. *Given a reactive agent $(q_0,succ)$, an LTL-X($\mathbb{P}$) formula ψ and a valuation function Π, if Kabanza et al.'s algorithm returns a reactive plan then this reactive plan satisfies ψ. Moreover, Kabanza et al.'s algorithm can find a reactive plan satisfying ψ if such plan exists.*

4. Control Strategy Algorithm based on Kabanza et al.'s Algorithm

The previous section has provided a brief overview about Kabanza et al.'s planning algorithm. This section will present a control strategy algorithm based on Kabanza et al.'s algorithm. Before providing this algorithm, we introduce the notion of control strategy.

Definition 6. *Let $T=(Q,A,B,\rightarrow,O,H)$ be a finite, non-blocking alternating transition system. For any function $f:Q^{+}\rightarrow A$, we say f is a control strategy of T. For any $q\in Q$ and $f:Q^{+}\rightarrow A$, the outcomes $Out_T^n(q,f)$ $(n\in\mathbb{N})$ and $Out_T(q,f)$ of f from q are defined as follows:*

$Out_T^n(q,f)=\{s\in Q^n:\ s[1]=q \text{ and } \forall 1\le i<n\exists b_i\in B(s[i]\xrightarrow{f(s[1,i]),b_i}s[i+1])\}$,

$Out_T(q,f)=\{\sigma\in Q^{\omega}:\ \sigma[1]=q \text{ and } \forall i\in\mathbb{N}\exists b_i\in B(\sigma[i]\xrightarrow{f(\sigma[1,i]),b_i}\sigma[i+1])\}$.

Furthermore, we define $Out_T^{+}(q,f)$ and $Out_T^{\infty}(q,f)$ as

$Out_T^{+}(q,f)=\bigcup_{n\in\mathbb{N}}Out_T^n(q,f)$ *and* $Out_T^{\infty}(q,f)=Out_T^{+}(q,f)\cup Out_T(q,f)$.

If alternating transition system T is known from the context, we often omit the subscripts in $Out_T^n(q,f)$, $Out_T(q,f)$, $Out_T^{+}(q,f)$, and $Out_T^{\infty}(q,f)$. Given a finite, non-blocking alternating transition system T, an LTL$_{-X}(\mathbb{P})$ formula ψ and a valuation function Π, we want to find an initial state q and a control strategy f of T so that $\sigma\vDash\psi$ for all $\sigma\in Out(q,f)$. An algorithm, which is used to find such initial state and control strategy, is presented in Algorithm 1 below.

(1) **input**: T, ϕ and $\prod$, where $T=(Q,A,B,\longrightarrow,O,H)$
(2) Construct a transition function $succ_T$ from T
(3) **for all** $q_0\in Q$ **do**
(4) Adopt Kabanza et al.'s algorithm to find a reactive plan RP_{kab} of $(q_0,succ_T)$ enforcing ϕ w.r.t. $\prod$
(5) **if** reactive plan RP_{kab} is found **then**
(6) RP=SimplyReactivePlan(RP_{kab}) /*See Algorithm 2 */
(7) f_{RP}=FunctionStrategy(RP) /*See Algorithm 3 */
(8) Return q_0 and f_{RP}
(9) **end if**
(10) **end for**
(11) Return false

Algorithm. 1: Control Strategy Algorithm

In Algorithm 1, steps (2), (6) and (7) are needed to be further refined. We illustrate them in turn.

Definition 7. *Let $T=(Q,A,B,\rightarrow,O,H)$ be a finite, non-blocking alternating transition system and $A=\{a_1,a_2,\cdots,a_k\}$. The transition function $succ_T$ w.r.t T is defined as: for any $q\in Q$, we set $succ_T(q)=((a_1,1,W_1),(a_2,1,W_2),\cdots,(a_k,1,W_k))$, where $W_i\triangleq\{q'\in Q:q\xrightarrow{a_i,b}q'$ for some $b\in B\}$ for $i=1,2,\cdots k$.*

By Definition 1, for any finite, non-blocking alternating transition system $T=(Q,A,B,\rightarrow,O,H)$, each set W_i mentioned above is finite and non-empty. Thus for any $q\in Q$, $(q,succ_T)$ is a reactive agent. Clearly, due to the finiteness of Q, A, and $\rightarrow$, the function $succ_T$ may be obtained using a simple algorithm. We leave it to interested reader. Before refining steps (6) and (7), we provide some notions and result below.

Definition 8. *Let $T=(Q,A,B,\rightarrow,O,H)$ be a finite, non-blocking alternating transition system, $q\in Q$ and let $succ_T$ be the transition function w.r.t T. A reactive plan of $(q, succ_T)$ is said to be a reactive plan of T.*

Definition 9. *Let $RP=\{(1,q_1,a_1,N_1),(2,q_2,a_2,N_2),\cdots,(k,q_k,a_k,N_k)\}$ be a reactive plan. For any finite sequence $s\in\{1,2,\cdots,k\}^{+}$, if $|s|>1$ and $s[i+1]\in N_{s[i]}$ for all $i<|s|$, then s is said to be a finite path of RP. For any two pathes s_1 and s_2 of RP, if $s_1[1]=1$ and $s_1[end]=s_2[1]=s_2[end]$, then the pair (s_1, s_2) is said to be a reachable cycle of RP.*

Now we refine steps (6) and (7). These two steps aim to get a control strategy from a reactive plan.

Step (6): *In this step, given a reactive plan RP, we will simplify it in this way: for any (i,q_i,a_i,N_i) in RP, if there exist $j_1,j_2,\cdots,j_m\in N_i$ with $m>1$ and $q_{j1}=q_{jn}$ for all $n\le m$, then we remain one of them and remove*

others from N_i. Thus for any (i,q_i,a_i,N_i) in the simplified reactive plan and for any world state q, there exists at most one plan state $j \in N_i$ with $q_j=q$. Formally, Step (6) is refined in Algorithm 2.

In this algorithm, the lines (3)-(11) is used to find a reachable cycle (*prefix*,*suffix*). Amongst, we adopt DijKstra's algorithm [19][20] to find the shortest pathes of *RP* from *i* to *i* and from 1 to *i* (see lines (4) and (6)). By the completeness of DijKstra's algorithm [19][20], *prefix* and *suffix* must can be found in this algorithm if the given reactive plan may generate trajectory.

Suppose that $RP = \{(1, q_1, a_1, N_1), (2, q_2, a_2, N_2), \cdots, (k, q_k, a_k, N_k)\}$
(1) SimplifyReactivePlan(RP){
(2) $note = 0$
(3) **while** $i \leq k$ and $note = 0$ **do**
(4) *suffix=shortest_ path(i,i)*
(5) **if** *suffix*$\neq \emptyset$ **then**
(6) *prefix=shortest_ path(1,i)*
(7) **if** $prefix \neq \emptyset$ **then**
(8) $note = 1$;
(9) **end if**
(10) **end if**
(11) **end while**
(12) **for all** $(i, q_i, a_i, N_i) \in RP$
(13) **for all** $j_1, j_2, \cdots, j_m \in N_i$ with $m > 1$ and $q_{j_1} = q_{j_2} = \cdots = q_{j_m}$
(14) **if** for some $l \leq m$, there exists $n < |prefix|$ such that i=*prefix[n]* and j_l=*prefix[n+1]* **then**
(15) $N_i = N_i - \{j_1, \cdots, j_{l-1}, j_{l+1}, \cdots, j_m\}$ /*Remove $j_1, \cdots, j_{l-1}, j_{l+1}, \cdots, j_m$ from N_i */
(16) **else if** for some $l \leq m$, there exists $n < |suffix|$ such that i=*suffix[n]* and j_l=*suffix[n+1]* **then**
(17) $N_i = N_i - \{j_1, \cdots, j_{l-1}, j_{l+1}, \cdots, j_m\}$ /*Remove $j_1, \cdots, j_{l-1}, j_{l+1}, \cdots, j_m$ from N_i */
(18) **else if**
(19) $N_i = N_i - \{j_2, j_3, \cdots, j_m\}$ /*Remove $j_2, j_3, \cdots, j_m$ from N_i */
(20) **end if**
(21) **end for**
(22) **end for**
(23) Return RP}

Algorithm.2: Simplifying Reactive Plan *RP*

Suppose that *RP* may generate trajectory and the reachable cycle (*prefix*,*suffix*) has been found. The lines (12)-(22) aim to simplify the reactive plan *RP* based *prefix* and *suffix* so that the simplified reactive plan may generate trajectory. Since *prefix* is the shortest path from 1 to *prefix[end]*, it is clear that there do not exist *i,j*<|*prefix*| such that $i \neq j$ and *prefix[i]=prefix[j]*. So, for the line (14) in Algorithm 2, there exists at most one natural number l such that *l*≤*m*, *i=prefix[n]* and j_l=*prefix[n+1]* for some *n*<|*prefix*|. Similar argument holds for the line (16). We provide a simple example below to illustrate Algorithm 2.

Example 1. *Consider the reactive plan* $RP = \{(1,q_1,a_1,\{2\}),(2,q_2,a_2,\{1,4\}),(3,q_3,a_3,\{1\}),(4,q_1,a_4,\{3\})\}$. *We adopt Algorithm 2 to simplify RP. It is easy to check that both suffix and prefix found in this algorithm are "121". For the SCR* $(2,q_2,a_2,\{1,4\}) \in RP$, *since both plan states 1 and 4 are labeled by* q_1 *and prefix=121, plan state 4 is removed from {1,4}. One may easily examine that the simplified reactive plan is* $\{(1,q_1,a_1,\{2\}),(2,q_2,a_2,\{1\}),(3,q_3,a_3,\{1\}),(4,q_1,a_4,\{3\})\}$.

In the above example, for the plan states 3 and 4 in the simplified reactive plan, there does not exist path from plan state 1 to these states, although such pathes exist for the original reactive plan. Thus a natural question arises: whether the simplification provided in Algorithm 2 may result in that the simplified reactive plan can not generate trajectory although the original reactive plan can do so. The following result reveals that this situation can not arise.

Theorem 2. *Let* $RP = \{(1,q_1,a_1,N_1),(2,q_2,a_2,N_2),\cdots,(k,q_k,a_k,N_k)\}$ *be a reactive plan. If RP generates trajectory, then so does the simplified reactive plan generated by Algorithm 2.*

Theorem 3. *Let T=(Q,A,B,→,O,H) be a finite, non-blocking alternating transition system, Π a valuation function, φ an LTL-X($\mathbb{P}$) formula, and let* $RP=\{(1,q_1,a_1,N_2),\cdots,(k,q_k,a_k,N_k)\}$ *be a reactive plan of T. We adopt Algorithm 2 to simplify RP. Then we have*

(1) For any (i,q_i,a_i,N_i) *in the simplified reactive plan and for any* $q\in Q$, *there exists at most one plan state* $j\in N_i$ *with* $q_j=q$.

(2) If RP satisfies φ then the simplified reactive plan also satisfies φ.

Suppose that $RP=\{(1,q_1,a_1,N_1),(2,q_2,a_2,N_2),\cdots,(k,q_k,a_k,N_k)\}$
FunctionStrategy(RP){
(1) **input**: s /*s is an array denoting a sequence of world states*/
(2) SeqOfPS[1]=1 /*SeqOfPS is an array denoting a sequence of plan states*/
(3) **if** $s[1]\neq q_1$ **then**
(4) Return a_1
(5) **end if**
(6) $i=2$
(7) **while** $i\leq|s|$ **do**
(8) k =SeqOfPS$[i-1]$
(9) **if** $s[i]=q_j$ for some $j\in N_k$ **then**
(10) SeqOfPS$[i]=j$
(11) $i=i+1$
(12) **else**
(13) Return a_1
(14) **end if**
(15) **end while**
(16) k =SeqOfPS$[i-1]$
(17) Return a_k}

Algorithm.3: Producing Control Strategy f_{RP}

Step (7). Next, we refine Step (7) in Algorithm 1. In Algorithm 2, a control strategy will be obtained from the simplified reactive plan. If the simplified reactive plan obtained by performing Algorithm 2 satisfies formula ϕ then it can generate a control strategy f_{RP} so that $\sigma\models\phi$ for all $\sigma\in Out(q_1,f_{RP})$. Algorithm 3 shows how to produce a control strategy from a simplified reactive plan.

The following theorem illustrates the correctness and completeness of Algorithm 1.

Theorem 4. *Given a finite, non-blocking alternating transition system T=(Q,A,B,→,O,H), LTL-X($\mathbb{P}$) formula φ and valuation function Π.*

(1) if control strategy algorithm returns a state q_0 *and a control strategy* f_{RP}, *then* $\sigma\models\phi$ *for any* $\sigma\in Out(q_0,f_{RP})$;

(2) if φ is total and there exists a state q of T and a control strategy $f:Q^+\to A$ *such that* $\sigma\models\phi$ *for all* $\sigma\in Out(q_1,f)$, *then the control strategy algorithm can find an initial state* q' *and a control strategy* $f':Q^+\to A$ *so that* $\sigma\models\phi$ *for all* $\sigma\in Out(q_1,f')$.

5. Conclusion and Future Work

Pola and Tabuada have introduced finite abstractions for control systems Σ with disturbance inputs [15,16]. However, since these finite abstractions are modeled by finite, non-blocking alternating transition systems rather than usual transition systems, the approaches provided in [5][7][8] are not suitable for finding control strategies for Pola and Tabuada's abstractions. To overcome this defect, this paper presents a control strategy algorithm based on Kabanza et al.'s planning algorithm (see Algorithm 1). This control strategy algorithm can be used to find an initial state and a control strategy of finite, non-blocking alternating transition system enforcing an given LTL-X formula. The correctness and completeness of this algorithm are explored. We demonstrate that this algorithm is correctand is complete w.r.t total LTL-X formulas (see Theorem 4). But it is still an open problem: whether Theorem 4 holds for all LTL-X formulas. We will explore this problem in further work.

6. Acknowledgments

This work received financial support of the National Natural Science of China (No. 61602249) and the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (No. 17KJB520012).}

7. References

[1] L. Habets and J. H. van Schuppen, A control problem for affine dynamical systems on a full-dimensional polytope, *Automatica*, 2004, 40: 21-35.

[2] R. Alur, and T. A. Henzinger, Discrete abstractions of hybrid systems, in Proceedings of *The IEEE*, 2000, 88(7): 971-984.

[3] M. Antoniotti and B. Mishra, Discrete Event Models + Temporal Logic = Supervisory Controller: Automatic Synthesis of Locomotion Controllers, *Proc. of IEEE International Conference on Robotics and Automation,* Nagoya-shi, Japan, 1995, pp. 1441-1446.

[4] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, Temporal logic motion planning for mobile robots, *Proc. of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, 2005, pp. 2020-2025.

[5] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, Hybrid controllers for path planning: A temporal logic approach, *Proc. of 44th IEEE Conference on Decision and Control and 8th European Control Conference (CDC-ECC'05)*, vol.5, Seville, Spain, 2005, pp. 4885-4890.

[6] X. Koutsoukos, P. Antsaklis, J. Stiver, and M. Lemmon, Supervisory control of hybrid systems, in Proceedings of *the IEEE,* 2002, 88: 1026-1049.

[7] G. Pola, A. Girard, and P. Tabuada, Approximately bisimilar symbolic models for nonlinear control systems, *Automatica*, 2008, 44: 2508-2516.

[8] P. Tabuada and G. J. Pappas, Finite bisimulations of controllable linear systems, *Proc. of the 42nd IEEE Conference on Decision and Control*, Hawaii, USA, 2003, pp. 634-639.

[9] M. Kloetzer and C. Belta, A Fully Automated Framework for Control of Linear Systems from LTL Specifications, *IEEE transactions on automatic control*, 2008, 53: 287-297.

[10] P. Tabuada and G. J. Pappas, Linear time logic control of discrete-time linear systems, *IEEE Transactions on Automatic Control,* 2006, 51(12): 1862-1877.

[11] G. Pola and P. Tabuada, Symbolic models for nonlinear control systems affected by disturbances, *Proc. of 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 251-256.

[12] R. Alur and D. L. Dill, A theory of timed automata, *Theoretical Computer Science*, 1994, 126: 183-235.

[13] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, The algorithmic analysis of hybrid systems, *Theoretical Computer Science*, 1995, 138: 3-34.

[14] B C. Kuo, *Automatic control systems*, Prentice-Hall, New York, 1975.

[15] B. Lacerda and P. U Lima, Linear-Time Temporal Logic Control of Discrete Event Models of Cooperative Robots, *Journal of Physical Agents*, 2008, 2: 53-61.

[16] G. Pola and P. Tabuada, Symbolic models for nonlinear control systems: Alternating approximate bisimulations, *SIAM Journal on Control and Optimization*, 2009, 48: 719-733.

[17] E. A. Emerson, Temporal and modal logic, in *Handbook of theoretical computer science: formal models and semantics*, J. van Leeuwen, Ed. Amsterdam, The Netherlands: North Holland/MIT Press, 1990, pp.995-1072.

[18] F. Kabanza, M. Barbeau, and R. St-Denis, Planning control rules for reactive agents, *Artificial Intelligence*, 1997, 95: 67-113.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed, Cambrideg, MA and New York: MIT Press and McGraw-Hill Book Company, 2001.

[20] E. W. Dijkstra, A note on two problems in connexion with graphs, *Numerische Mathematik,* 1959, 1: 269-271.