Article
Supervisory Control of Automated Manufacturing Systems Based on State-Tree Structures

Chan Gu, Junbo Zhao and Zhou He *

School of Electrical and Control Engineering, Shaanxi University of Science and Technology, Xi’an 710026, China; guchan@sust.edu.cn (C.G.); 20061019@sust.edu.cn (J.Z.)
* Correspondence: hezhou@sust.edu.cn

Abstract: The automated manufacturing systems (AMS) can be regarded as a resource allocation system which can be abstracted as discrete event systems (DES) for analysis. Once an effective computational method is put forward to the DES, the corresponding real AMS, connected by networks or sensors, can be indirectly controlled. Supervisory control theory (SCT) of DES is developed with the intention of discovering the maximally permissible supervisor of AMS. The state-tree structure (STS) is an extension of SCT, whose application is mainly to solve the SCT state explosion of complicated AMS. The nonblocking supervisory control is investigated on the foundation of predicates in STS, of which the synthesis focuses on the states of the system but ignores the transition between events. In order to effectively solve the above problems, this paper puts forward a novel method based on state feedback control (SFBC) in the STS framework, by which the relation among events is discussed. First, following the definition of the SFBC for predicates, the algorithms to compute the SFBC under reachability, coreachability, and weak controllability are introduced. Second, it is proved that the nonblocking SFBC obtained by the proposed algorithm is equivalent to the one by computing predicates first, which ensures the satisfactory controllers for the supervisory control problems. To demonstrate the contribution of the proposed approach, which is designed considering the symmetry of the system behavior between the real system and its STS model, three examples are illustrated.

Keywords: discrete event systems; supervisory control of automated manufacturing systems; state-tree structures

1. Introduction

As the Industrial 4.0 is developing rapidly, automated manufacturing systems (AMS) have been extensively employed to achieve managing and manufacturing process automation. In the AMS, the human intervention operation is eliminated, and the management process and technological process are fully automated. To realize the supervision and control of this process, the cooperation of software and hardware equipment is necessary. Therefore, an effective supervisory controller is of vital importance to achieve the smooth operation of the AMS.

The AMS can be regarded as a resource allocation system, and the supervisory controller is responsible for supervising the resource allocation in the control system [1–4]. The trajectories of such systems in space and time are discrete, so it is able to be extracted as a discrete-event system (DES) before investigation [5–13]. In the 1980s, Ramadge and Wonham put forward SCT in their research, which is also regarded as the Ramadge–Wonham (RW) framework [14–16] that offers a mathematical method with the intention of analyzing and controlling DES. On the other hand, for systems with high complexity, the completion of analysis work must be based on large-scale automated model. Based on this, a key obstacle faced by SCT is state explosion due to the sophistication of the controllers. For the purpose of effectively solving this problem, Ma and Wonham proposed a novel DES framework, which is named state-tree structures (STS) [17,18]. The research found a method that can make full use of STS to complete DES modeling. Hierarchical state-tree as well as
the dynamic module, which is named holons (inspired by [19]), can be used to describe its concurrency and hierarchy. Both for the horizontal and vertical level, STS is equipped with the corresponding structure to model the system. In addition, in order to offer a compact representation of state-tree, a binary decision diagram (BDD) is proposed [20]. BDD can lay a good foundation for the optimization of supervisory control calculation. The application of BDD is conducive to the improvement of calculation efficiency, and can also have a positive impact on the representation of complex models in [21,22]. For predicates, in the case of displaying BDD, the number of nodes of BDD can directly determine the complexity of the method used in [17], rather than the state size of the model. In the actual operation process, the size of states is much larger than that of BDD nodes, so the utilization of BDD in the calculation of the supervisors can obtain more effective results.

In the last few years, many scholars have conducted a lot of research on the basis of STS. Some supervisory control (SC) problems with observation are analyzed and discussed, and several subpredicates such as the weakly controllable, coreachable, and observable (WCCO) subpredicate are calculated so as to settle the STS control problem under various situations in [23,24]. In [25], an STS with conditional-preemption matrices (STSM) is put forward, which is a new STS framework and used to solve various priority problems. At the same time, the optimal non-blocking supervisory control is arranged in the STSM framework. In [26], the symmetry of DES, which has parallel components as well as fixed buffer sizes, is studied with the purpose of extracting the invariance of abstract control functions. In order to ensure the solution of the SC problem of real-time systems (RTS) to the greatest extent, the reference [27] proposes a new RTS modeling framework based on STS to guarantee that tasks with multiple periods can be effectively solved. In this study, priority-free conditionally-preemptive scheduling is utilized with the intention of describing and defining the preemption relationship between different items in RTS. The authors in [28] propose a top-down supervisor localization procedure, which creates the pure distributed control architecture of DES, so it plays a unique role in the control design of large-scale systems. A control logic–failsafe mechanism based on STS is proposed in [29] to solve the problem that dangerous flight maneuvers might be executed when unexpected failures or command conflicts happen. Similarly, to solve the complex system, the studies [30–32] develop the decentralized SC of DES by use of STS to manage state explosion, and an independent optimal (maximally permissible) supervisor is prepared by decomposing the specification of the controlled system into several sub-specifications. In addition, the reference [33] gives full play to the role of the nesting method to decompose STS and forms a group of STS nests. So to equip all nests with supervisors, this reference also extends the boundary consistency attributed to ensure that it can cover the supervision control of STS, with the intention of reducing the computational complexity of optimal supervisors. All these conclusions come from the analysis and synthesis of predicates, and the control functions are given by computing the optimal subpredicate, which is represented by BDD. This is of high efficiency to solve the control problems of DES. These studies investigate the computation of supervisors based on predicates representing a suit of state-trees, but lack the discussion of the relation among events under certain constraints.

As is known to all, in [14], a DES is modeled by a controlled automaton, and the formal language, which is produced by the automaton over an alphabet of event, labels the conduction of the DES. The computation of the supervisor is proposed in view of the formal language, which specifies the control strategy. In [17], after standing for predicates by BDD, the specification is given by illegal state-trees, and the supervisor is computed by analyzing if a state-tree satisfies certain conditions. Though the computational complexity of the supervisor is reduced substantially, the relation among events is not discussed. In some cases, two events can occur at one state. In [17], if the state is illegal, the two events are disabled simultaneously, which is not conducive to the analysis of the control functions. To fix this, in this paper, we aim to construct the control functions of controllable events by the means of analyzing the property of the state-feedback control (SFBC) of events for a predicate which is given under reachability, coreachability, and weak controllability,
respectively. First, the definition of the SFBC of events for a given predicate is proposed, and the property of the SFBC for predicates are discussed. Second, the definition and computation of the SFBC under reachability, coreachability, and weak controllability are introduced, respectively. Meanwhile, it is proved that the SFBC obtained by the proposed algorithms in this paper is equivalent to the one by computing the subpredicates first, which implies that the controllers obtained based on SFBC are also equivalent to the one on predicates. In summary, by comparison to the study [17], we provide an approach to achieve the aim of resolving the supervisory control problem based on SFBC, which makes the control strategy clearer while ensuring the computation efficiency. Moreover, this paper lays a foundation for the following work since the SFBC of events (indicating the relation between state-trees and events) is very important to the synthesis of the natural projection, and this leads to the development of other research, such as observability and decentralization. Finally, the contribution is demonstrated by two examples: Transfer Line and Small Factory. In addition, Small Factory, with the specification represented by the SFBC with one event disabled at a state-tree, is described with the intention of clarifying the devotion of this research.

The reminder of this study is structured as below. Section 2 is aimed at introducing the background of the STS model, and Section 3 explains the definition and properties of the SFBC of events for predicates. Section 4 emphatically studies the SFBC under reachability, coreachability, and weak controllability, respectively. Algorithms are introduced to compute the SFBC under certain constraints. Section 5 investigates the nonblocking SFBC for a given STS, which also shows the satisfactory solution for supervisory control problems. Section 6 presents three examples, which are Transfer Line, Small Factory, and Guide Way. Section 7 discusses the advantages and disadvantages of this paper. At last, Section 8 comes to a conclusion of this study.

2. Concepts and Terminologies

2.1. Sts Model

This chapter first analyzes and explains the state-tree structures (STS). The state-tree includes two structures, namely horizontal and vertical structure. Secondly, the holon is a local transition graph (see Section 2.1.2) and is assigned to state-tree or OR superstate. Then, the complete form of STS is described, which is the state-tree as well as its related set consisting of holons.

2.1.1. State-Tree

The four-tuple \((X, x_0, T, E)\) is considered to be a state-tree, where

- \(X\) is a structured state set, of which the number is limited;
- \(x_0 \in X\) is a particular state which is named the root state;
- \(T : X \rightarrow \{\text{and, or, simple}\}\) is the type function;
- \(E : X \rightarrow 2^X\) is the expansion function, where \(x \in X, \emptyset \subset Y \subseteq X\) and \(x \notin Y\);

\[
E(x) = \begin{cases} Y, & \text{if } T(x) \in \{\text{and, or}\} \\ \emptyset, & \text{if } T(x) = \text{simple} \end{cases}
\]

An AND superstate \(x\) is used to stand for the Cartesian product of its component, while the OR superstate is for the disjoint union. In fact, the AND superstate implies that its components can be reached simultaneously, while the OR superstate means the exclusion of its components. Shutting off the branches (but not all) which are rooted by OR superstates generates a sub-state-tree whose function is similar to a subset of states in a generator. Especially, a basic-sub-state-tree ensures that every OR superstate must own one component. \(B(S, T)\) stands for the basic-sub-state-tree set of \(ST\). What is more, as the sub-state-tree can be regarded as a state-tree, “sub-state-tree” is able to be expressed as “state-tree”.
2.1.2. Holon

A Holon \( H \) is described as \( H := (X, \Sigma, \delta, X_0, X_m) \), where

- \( X \) is the non-null state set, organized as the nonintersecting combination of \( X_E \) and \( X_I \), which are the external and internal state set;
- \( \Sigma \) is the event set, which is structured as the nonintersecting combination of \( \Sigma_B \) and \( \Sigma_I \), which are the boundary and internal event set, i.e., \( \Sigma = \Sigma_B \cup \Sigma_I \); the event set can also be seen as \( \Sigma = \Sigma_u \cup \Sigma_c \), where \( \Sigma_u \) is the uncontrollable event set and \( \Sigma_c \) is the controllable event set;
- The transition structure \( \delta : X \times \Sigma \rightarrow X \) is a partial function; this shows the transition between the states in the holon;
- \( X_0 \subseteq X_I \) is the initial state set;
- \( X_m \subseteq X_I \) is the terminal state set.

2.1.3. State-Tree Structures

**Definition 1.** The six-tuple \( (ST, H, \Sigma, \Delta, ST_0, ST_m) \) is considered to be a state-tree structure, where

- \( ST := (X, x_0, T, \mathcal{E}) \) is a state-tree.
- \( H := \{ H^a | T(a) = \sigma \text{ or } H^a = (X^a, \Sigma^a, \delta^a, X^a_0, X^a_m) \} \) is a set of holons which can be matched to OR superstate of \( ST \), in which \( a \) is an OR superstate and \( H^a \) is the matching objective for holon, which is to say, \( H^a \) portrays the dynamics among units of \( a \) as well as the transitions between the upper holon and \( H^a \).
- \( \Sigma := \bigcup_{H \in H} \Sigma^a_I \) is the event set which consists of all events that happens in \( H \), in which \( \Sigma^a_I \) represents the internal event set of \( H^a \).
- \( \Delta : ST(ST) \times \Sigma \rightarrow ST(ST) \) is the transition function, in which \( ST(ST) \) stands for the set of sub-state-trees of \( ST \). This is the global transition function, which is different from \( \delta \).
- \( ST_0 \in ST(ST) \) is the initial sub-state-tree.
- \( ST_m \subseteq ST(ST) \) is the marker sub-state-tree set.

Similarly, \( \Gamma : ST(ST) \times \Sigma \rightarrow ST(ST) \) represents a sub-state-tree of \( ST \) into another sub-state-tree. \( \Gamma(T, \sigma) \) is the source-state-tree that, upon the occurrence of event \( \sigma \), transforms to a sub-state-tree of \( T \).

2.2. Predicate

Predicate \( P \) described on \( B(ST) \) in \([17]\) is regarded as a function \( P : B(ST) \rightarrow \{0, 1\} \). A predicate is able to be recognized by a set of basic-state-trees, named as \( B_P \), thus

\[
B_P := \{ b \in B(ST) | P(b) = 1 \} \subseteq B(ST).
\]

\( P_1 \leq P_2 \) or \( P_1 \) is stronger compared to \( P_2 \) indicates that \( b \vdash P_1 \Rightarrow b \vdash P_2 \). Quite the opposite, we can find that \( P_2 \) is weaker than \( P_1 \). In other words, \( P_1 \leq P_2 \) means that \( P_b \leq P_1 \Rightarrow P_b \leq P_2 \), where \( b \) satisfies \( P_b \) only. Furthermore, \( P \) holds for \( ST_I \), for example, \( ST_1 \vdash P \text{ iff } B(ST) \subseteq B_P \).

What is more, \( R(G, P) \) is the reachability predicate \([17]\), and it is proposed with the intention of indicating all the basic state-trees reachable form the initial basic state-tree via state-trees satisfying \( P \).

\( R(G, \cdot) \) is monotone, i.e., \( P \leq Q \) implies \( R(G, P) \leq R(G, Q) \). Similarly, \( CR(G, P) \) is the coreachability predicate met with the basic state-trees that is able to reach the marker basic state-tree along state-trees satisfying \( P \). As far as fixed \( G \), \( CR(G, \cdot) : Pred(ST) \rightarrow Pred(ST) \) is a predicate transformer. Clearly, \( CR(G, P) \leq P \) and \( CR(G, \cdot) \) is monotone.

As is analyzed in \([17]\), the weak controllability of predicates is defined instead of the controllability. By ignoring the condition \( P \leq R(G, P) \) of the controllability, the calculation expanse is much declined, which is proved to be reasonable since the weak controllability of a predicate \( P \) implies the controllability of \( R(G, P) \).
Predicate $P$ is weakly controllable if any event that cannot be controlled takes place at state-trees satisfying $P$, then the consequential state-trees also satisfy $P$ [17]. This is able to be written as

$$(\forall \sigma \in \Sigma_u)b \models P \Rightarrow \Delta(b, \sigma) \models P.$$  

Let $CP(P)$ represent the set of weakly controllable subpredicates which are stronger compared to $P$ by

$$CP(P) := \{ K \in \text{Pred}(ST) | K \leq P \& (\forall \sigma \in \Sigma_u)b \models K \Rightarrow \Delta(b, \sigma) \models K \}.$$  

The weakest element $\sup CP(P)$ exists, which indicates that the supremal weakly controllable subpredicate of $P$ exists with $\sup CP(P) = \bigvee \{ K | K \in CP(P) \}$.

Moreover, the nonblockingness of a predicate $P$ is confirmed if $R(G, P)$ is stronger compared to $CR(G, P)$. $P$ is coreachable if $P \leq CR(G, P)$, which indicates the nonblockingness of $P$, as $R(G, P) \leq P \leq CR(G, P)$ generally. Considering the coreachability and weakly controllability, as far as predicate $P$, let $C^2P(P)$ represent the set of the subpredicates of $P$ that are weakly controllable, coreachable, and stronger than $P$ [17], i.e.,

$$C^2P(P) := \{ K \in \text{Pred}(ST) | K \leq P \& K \leq CR(G, K) \& (\forall \sigma \in \Sigma_u)b \models K \Rightarrow \Delta(b, \sigma) \models K \}.$$  

In particular, a (unique) supremal element $\sup C^2P(P)$ exists in $C^2P(P)$, which is given by

$$\sup C^2P(P) = \bigvee \{ K | K \in C^2P(P) \}.$$  

Notice that the reachability subpredicate of $\sup C^2P(P)$ agrees on the reachability subpredicate of the optimal nonblocking controller, which shows the supremal controller of the supervisory control problems.

The control behavior of an STS is carried out by the utilization of state feedback control (SFBC) [17], and it is described by a set of predicates. Let $f : B(ST) \rightarrow \Pi$ be an SFBC for $G$, in which $\Pi := \{ \Sigma' \subset \Sigma | \Sigma_u \subset \Sigma' \}$. For $\sigma \in \Sigma$, clarify a control function (a predicate) $f_\sigma : B(ST) \rightarrow \{ 0, 1 \}$ on the basis of $f_\sigma(b) = 1$ iff $\sigma \in f(b)$. As a result, the synthesis of the supervisor is carried out by $\{ f_\sigma | \sigma \in \Sigma \}$. In consideration of an event $\sigma \in \Sigma$, $f_\sigma(b) = 1$ (respectively, 0) indicates that event $\sigma$ is allowed (respectively, not allowed) to occur at $b$. $f_\sigma$ is defined to be equal to 1 for $\sigma$ with $\sigma \in \Sigma_u$.

The STS $G$ under $f$ is depicted as $G^f = (ST, \mathcal{H}, \Sigma, \Delta^f, P_0^f, P_m^f)$, where $P_0^f = R(G^f, true) \land P_0$, $P_m^f = R(G^f, true) \land P_m$, and $\Delta^f(b, \sigma) = \Delta(b, \sigma)$ if $f_\sigma(b) = 1$. Or else, $\Delta^f(b, \sigma) = \emptyset$. If $R(G^f, true)$ is stronger than $CR(G^f, true)$, the SFBC $f$ is nonblocking. Moreover, if $P$ is weakly controllable and coreachable, it is proved that there does exist a nonblocking SFBC $f$ with $R(G^f, true) = R(G, P)$.

**Example 1.** For a small task which must be performed by two machines $R_1$ and $R_2$, each machine has three states “Idle”, “Work”, and “Breakdown”, and the transition between these states are represented by events $\alpha, \gamma, \beta, \lambda, \mu$, and $\eta$. $\alpha$ stands for a workpiece arriving machine $R_1$ and $\gamma$ for machine $R_2$. The occurrence of $\beta$ signifies the processing complete. $\lambda$ is used to represent the damage of the machine, while $\mu$ (respectively, $\eta$) stands for the repair of machine $R_1$ (respectively, $R_2$). Its STS $G = (ST, \mathcal{H}, \Sigma, \Delta, P_0, P_m)$ is depicted with the state-tree shown in Figure 1 and holons shown in Figure 2.
Figure 1. State-tree of the given STS $G$.

Figure 2. The holons of the given STS $G$.

For the state-tree shown in Figure 1,

- $x_0 = \text{Plant}$ is the root state;
- $E(x_0) = \{R_1, R_2\}$;
- $E(R_1) = \{R_1, 0, R_1, 1, R_1, 2\}$, where $R_1, 0$, $R_1, 1$, and $R_1, 2$ represent the components 0, 1, and 2 of $R_1$, respectively. This means that there are three states in machine $R_1$ which represent the state “Idle”, “Work”, and “Breakdown”;
- $E(R_2) = \{R_2, 0, R_2, 1, R_2, 2\}$, where $R_2, 0$, $R_2, 1$, and $R_2, 2$ represent the components 0, 1, and 2 of $R_2$, respectively, which represent the state “Idle”, “Work”, and “Breakdown” of machine $R_2$.
- $T(\text{Plant}) = \text{AND}$, implying that $R_1$ and $R_2$ work concurrently;
- $T(R_1) = T(R_2) = \text{OR}$, implying that the states $R_1, 0$, $R_1, 1$, and $R_1, 3$ are mutually exclusive, i.e., the system can only reach one state at a time; and
- $T(R_1, 0) = T(R_1, 1) = T(R_1, 2) = T(R_2, 0) = T(R_2, 1) = T(R_2, 2) = \text{SIMPLE}$, which means the exact states for each machine.

For the holons shown in Figure 2, there are two holons $H^{R_1}$ and $H^{R_2}$ in this figure. Take $H^{R_1}$ as the example.

- $X = \{0, 1, 2\}$
- The event set $\Sigma = \{\alpha, \beta, \lambda, \mu\}$;
- $X_0 = \{0\}$ and $X_m = \{1\}$;
- $\delta(0, \alpha) = 1$,
- $\delta(1, \beta) = 0$,
- $\delta(1, \lambda) = 2$, and
- $\delta(2, \mu) = 0$.

In this example, there is not an upper or lower holon for $H^{R_1}$, which implies that the boundary event does not exist. Holon $H^{R_1}$ denotes the transition between the states in machine $R_1$ if one event occurs. Besides, since $R_1$ and $R_2$ are two components of an AND superstate, the holons
$H^{R,1}$ and $H^{R,2}$ are parallel, which means that the shared event ($\beta \text{ or } \lambda$) occurs simultaneously. For the system composed of two machines $R_1$ and $R_2$, the system behavior can be described by the following explanation of the STS model.

- $\Sigma = \Sigma_{R_1} \cup \Sigma_{R_2} = \{a, b, \lambda, \mu, \gamma, \eta\}$,
- $ST_0 = \{R_{1,0}, R_{2,0}\}$, which stands for the initial state-tree, that is, $R_1$ is at state 0 and $R_2$ is at 0;
- $ST_m = \{(R_{1,0}, R_{2,0})\}$, and
- the transition function $\Delta$ is detailed by
  \[
  \Delta(\{R_{1,0}, R_{2,0}\}, \alpha) = \{R_{1,1}, R_{2,1}\},
  \Delta(\{R_{1,0}, R_{2,0}\}, \gamma) = \{R_{1,0}, R_{2,1}\},
  \Delta(\{R_{1,1}, R_{2,0}\}, \alpha) = \{R_{1,1}, R_{2,1}\},
  \Delta(\{R_{1,1}, R_{2,1}\}, \beta) = \{R_{1,0}, R_{2,0}\},
  \Delta(\{R_{1,1}, R_{2,1}\}, \lambda) = \{R_{1,2}, R_{2,2}\},
  \Delta(\{R_{1,2}, R_{2,2}\}, \mu) = \{R_{1,0}, R_{2,2}\},
  \Delta(\{R_{1,2}, R_{2,2}\}, \eta) = \{R_{1,2}, R_{2,0}\},
  \]

\ldots

3. Control Functions for Predicates

Since the synthesis of the given STS is proposed in view of a set of state-trees represented by a predicate, the computation complexity is greatly reduced. However, the relation between state-trees and events remains to be discussed. As for this part, with the intention of investigating this while ensuring the computation efficiency, the state feedback control (SFBC) $f$ of controllable events is introduced on the basis of predicates instead of STS (as in [17,18,20]). A control function set is defined with respect to an individual predicate $P$, which is composed by SFBC $f$ of all controllable events for $P$. To make the definitions clearer, an example is given in detail.

As known in [17,18,20], $f : B(ST) \to \Pi$ is proposed to define the SFBC for $G$, in which

$$\Pi := \{\Sigma' \subseteq \Sigma|\Sigma_u \subseteq \Sigma'\}.$$  

The event $\sigma$ is allowed (respectively, not allowed) to occur at basic-state-tree $b$ if $\sigma \in f(b)$ (respectively, $\sigma \notin f(b)$). As far as event $\sigma$, the predicate $f_\sigma : B(ST) \to \{0, 1\}$ is described by $f_\sigma(b) := 1$ iff $\sigma \in f(b)$. In previous studies, the supervisory control problem is resolved by means of investigating the properties of predicates, which means that the state illegal implies that the related event is ineligible. To avoid this, in this chapter, the SFBC of each event and state-trees is studied on the basis of a given predicate. First, Definition 2 is introduced.

**Definition 2.** Suppose an STS $G := (ST, H, \Sigma, \Delta', P_0, P_m)$ with the controlled STS as $G' := (ST, H, \Sigma, \Delta, P_0', P_m)$ for the closed system supervised by the SFBC $f$. As for the predicate $P$, an SFBC $f$ of $\sigma$ on $P$, denoted by $f_\sigma^P : B(ST) \to \{0, 1\}$, is defined by

$$f_\sigma^P(b) := 1 \text{ iff } P(b) = 1 \& \Delta(b, \sigma)! \& \Delta(b, \sigma) \models P.$$

**Remark 1.** (1) Simply, the SFBC $f$ of an event $\sigma$ on $P$, denoted by $f_\sigma^P$, is a predicate and satisfied by any basic-state-tree $b$ if $\sigma$ is enabled at $b$ and $\Delta(b, \sigma)$ satisfies $P$, simultaneously. Notice that the event $\sigma$ can be both controllable and uncontrollable since there might exist $\sigma \in \Sigma_u \& \Delta(b, \sigma) \not\models P$. In fact, $f_\sigma^P$ is a predicate represented by BDD, which ensures the computational efficiency. Meanwhile, $f_\sigma^P$ shows the relation between state-trees and events for $P$, which makes the computation more fixable and accurate.

- (2) $f(P)$ represents the set of eligible events for $P$, i.e., $f(P) = \{\sigma|f_\sigma^P = 1\}$.
- (3) Let $f^P$ denote the SFBC for a predicate $P$ with $f^P = \bigvee_{\sigma \in \Sigma} f_\sigma^P$, which is also a predicate.
- (4) For $f_\sigma^P$ defined as a predicate, $B_{f_\sigma^P}$ stands for the set of state-trees satisfying $f_\sigma^P$.
- (5) $f_\sigma^P(b) = 1$ means that $\sigma$ is enabled at $b$ and $P$ is satisfied by the target state-tree.
For a predicate \( P' \leq P \), the control function on \( P' \) over \( P \) can be denoted as

\[
f_{\sigma}[P'](b) := 1 \text{ if } \sigma \in f(b) \& P' = 1 \& \Delta(b, \sigma) \models P.
\]

Thus, the following proposition is obtained.

**Proposition 1.** For predicates \( P_1 \leq P \) and \( P_2 \leq P \),

\[
f_{\sigma}[P_1 \lor P_2] = f_{\sigma}[P_1] \lor f_{\sigma}[P_2].
\]

**Proof.** With the intention of demonstrating the equation, it is necessary for us to show that \( f_{\sigma}[P_1 \lor P_2] \leq f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \) and \( f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \leq f_{\sigma}[P_1 \lor P_2] \).

1. Show that \( f_{\sigma}[P_1 \lor P_2] \leq f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \). For \( b \models f_{\sigma}[P_1 \lor P_2] \), \( P_1 \lor P_2 \models \Delta(b, \sigma) \models P \Rightarrow b \models (P_1 \lor P_2) \& \Delta(b, \sigma) \models P \), which implies that \( b \models P_1 \lor P_2 \) & \( \Delta(b, \sigma) \models P \Rightarrow (P_1(b) = 1 \& P_2(b) = 1) \& \Delta(b, \sigma) \models P \Rightarrow (P_1(b) = 1 \& \Delta(b, \sigma) \models P) \) or \( (P_2(b) = 1 \& \Delta(b, \sigma) \models P) \) \( b \models f_{\sigma}[P_1 \lor P_2] \). Thus, \( b \models f_{\sigma}[P_1 \lor P_2] \).

2. Prove that \( f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \leq f_{\sigma}[P_1 \lor P_2] \). For \( b \models f_{\sigma}[P_1 \lor P_2] \), \( b \models f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \). This means that \( (P_1(b) = 1 \& \Delta(b, \sigma) \models P) \) or \( (P_2(b) = 1 \& \Delta(b, \sigma) \models P) \) \( b \models (P_1 \lor P_2) \) & \( \Delta(b, \sigma) \models P \Rightarrow b \models f_{\sigma}[P_1 \lor P_2] \). Therefore, \( f_{\sigma}[P_1 \lor P_2] = f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \) as required.

\( \square \)

**Remark 2.** (1) For predicates \( P_1 \) and \( P_2 \), there might exist \( b \models P_1 \) and \( b' \models P_2 \) with \( \Delta(b, \sigma) = b' \). Thus, \( f_{\sigma}[P_1 \lor P_2](b) = 1, f_{\sigma}[P_1](b) = 0 \) and \( f_{\sigma}[P_2](b) = 0 \), implying that \( f_{\sigma}[P_1 \lor P_2] = f_{\sigma}[P_1] \lor f_{\sigma}[P_2] \) is not always true.

(2) Since the SFBC \( f \) of \( \sigma \) is a predicate, propositional logic operators are also defined as

\[
\neg f_{\sigma}[P](b) = 1 \iff f_{\sigma}[P](b) = 0
\]

\[
(f_{\sigma}[P_1] \land f_{\sigma}[P_2])(b) = 1 \iff f_{\sigma}[P_1](b) = 1 \land f_{\sigma}[P_2](b) = 1
\]

\[
(f_{\sigma}[P_1] \lor f_{\sigma}[P_2])(b) = 1 \iff f_{\sigma}[P_1](b) = 1 \lor f_{\sigma}[P_2](b) = 1
\]

**Example 2.** In consideration of an STS \( G = (ST, H, \Sigma, \Delta, P_0, P_m) \), which is depicted in Figure 2, suppose a predicate \( P \) with \( B_P = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 1, R_2, 1\}\} \) and \( \Sigma_{\text{c}} = \{a, \gamma, \mu, \eta\} \). According to Definition 2, it is easy to know that

- \( B_{f_{\sigma}[P]}(b) \) with \( a \) is enabled at \( \{R_1, 0, R_2, 0\} \) and \( \{R_1, 0, R_2, 0\} \) satisfying \( P \).
- \( B_{f_{\sigma}[P]}(b) = \emptyset \) since \( \gamma \) is eligible at \( \{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\} \) but neither \( \{R_1, 0, R_2, 0\} \) or \( \{R_1, 1, R_2, 0\} \) satisfies \( P \).
- Similarly, \( B_{f_{\sigma}[P]}(b) = B_{f_{\sigma}[P]}(b) = \emptyset \).

4. SFBC under Reachability, Coreachability, and Weak Controllability

Aiming to discuss the supervisory control problem on the foundation of SFBC, better control strategies can be proposed since the control behavior obtained by SFBC is relatively flexible. On the other hand, the introduction of nonblocking supervisory control based on SFBC helps to decline the calculation complexity of controllers. In this section, the reachability, coreachability, and weak controllability are defined in view of SFBC. Examples are given to explain the definitions.
4.1. The Reachability and Coreachability

To solve the nonblocking supervisor, the definition and property of the SFBC \( f \) of events under reachability and coreachability are investigated, respectively, in this subsection.

\[ \text{G} = \langle \text{ST}, \mathcal{H}, \Sigma, \Delta, P_0, P_m \rangle \] is given to be a state-tree structure. The SFBC \( \overrightarrow{f} \) under reachability over a predicate \( P \) can be obtained by

- \( b \neq P \Rightarrow (\forall \sigma \in \Sigma) \overrightarrow{f}_P(b) = 0; \)
- \( b \models P \& (\exists \sigma \in \Sigma) \Delta(b_0, \sigma) \models P \Rightarrow (\forall \sigma \in \Sigma) \overrightarrow{f}_P(b_0) = 0; \)
- \( b \models P \& (\exists \sigma \in \Sigma) \Delta(b_0, \sigma) \models P \Rightarrow \overrightarrow{f}_P(b_0) = 1; \)
- \( \overrightarrow{f}_P(b) = 1 \& \Delta(b, \sigma) = b' \& (\exists \sigma' \in \Sigma) f^P_{\sigma'}(b') = \overrightarrow{f}_P(b) = 1; \)
- No other basic state-tree \( b \) satisfies \( (\exists \sigma \in \Sigma) \overrightarrow{f}_P(b) = 1. \)

**Proposition 2.** A predicate \( P \) is reachable if

\[
(\forall b \models P)(\exists \sigma \in \Sigma) f^P_{\sigma}(b) = 1 \Rightarrow \overrightarrow{f}_P(b) = 1.
\]

**Proof.** To prove this proposition, we need to investigate if \( (\forall b \models P)(\exists \sigma \in \Sigma) f^P_{\sigma}(b) = 1 \Rightarrow \overrightarrow{f}_P(b) = 1, b \models R(\text{G}, P). \)

It is easy to know that \( (\forall b \models P)(\exists \sigma \in \Sigma) f^P_{\sigma}(b) = 1 \Rightarrow \overrightarrow{f}_P(b) = 1 \) implies \( b \models P \) and \( f^P_{\sigma_1}(\Gamma(b, \sigma)) = 1. \) \( \overrightarrow{f}_P(\Gamma(b, \sigma)) = 1 \Rightarrow \Gamma(b, \sigma_1) \models P \& \overrightarrow{f}_P(\Gamma(b, \sigma_1)) = 1. \) And so forth, \( \Gamma(b, \sigma_1 \cdots \sigma_n) \models P \) with \( \Gamma(b, \sigma_1 \cdots \sigma_n) = b_0, \) which means that \( b_0 \models R(\text{G}, P). \)

**Proposition 3.** With a given STS \( \text{G} \) and a predicate \( P, \) let \( \overrightarrow{f}_P : B(\text{ST}) \rightarrow \{0,1\} \) be the SFBC under reachability. For the reachability predicate \( R(\text{G}, P), \) the SFBC \( f^R_{\sigma}(\text{G}, P) = \overrightarrow{f}_P. \)

**Proof.** To prove \( f^R_{\sigma}(\text{G}, P) = \overrightarrow{f}_P, \) two steps are necessary.

- Prove \( f^R_{\sigma}(\text{G}, P) \leq \overrightarrow{f}_P. \) If for any state-tree \( b, b \models \overrightarrow{f}_P \Rightarrow b \models R(\text{G}, P) \& \Delta(b, \sigma) \models R(\text{G}, P). \) This implies that there exists \( s = \sigma_1\sigma_2 \cdots \sigma_n, \Delta(b_0, s) = b \models P \) and \( \Delta(b, \sigma) \models P. \) Thus,

\[
\overrightarrow{f}_P(\sigma_1)(b_0) = 1, \\
\overrightarrow{f}_P(\sigma_2)(\Delta(b_0, \sigma_1)) = 1, \\
\overrightarrow{f}_P(\sigma_3)(\Delta(b_0, \sigma_2)) = 1, \\
\cdots \\
\overrightarrow{f}_P(\sigma_n)(\Delta(b_0, \sigma_2 \cdots \sigma_{n-1})) = 1, \\
\overrightarrow{f}_P(b) = 1.
\]

- Prove \( f^R_{\sigma}(\text{G}, P) \leq \overrightarrow{f}_P. \) If for any state-tree \( b, b \models \overrightarrow{f}_P \Rightarrow b \models P \& \Delta(b, \sigma) \models P. \) This implies that there exists \( s = \sigma_0\sigma_1 \cdots \sigma_n \) with \( \Gamma(b, s) = b_0 \models P. \) Considering the definition of reachability predicate, \( b_0 \models R(\text{G}, P), \Delta(b_0, \sigma_0) \models R(\text{G}, P), \Delta(b_1, \sigma_1) \models R(\text{G}, P), \cdots, \Delta(b_n, \sigma_n) = b \models R(\text{G}, P). \)

Therefore, \( \Delta(b_0, \sigma_n) = b \models f^R_{\sigma}(\text{G}, P). \overrightarrow{f}_P \leq f^R_{\sigma}(\text{G}, P) \) is proved.
Then, the proposition is proved to be true. □

Besides, \( P \leq Q \Rightarrow R(G, P) \leq R(G, Q) \) implies \( f^{R(G, P)}_\sigma(b) = 1 \Rightarrow f^{R(G, Q)}_\sigma(b) = 1 \).

According to Proposition 3, if \( P \leq Q \), then \( f^{Q}_\sigma(b) = 1 \Rightarrow f^{Q}_\sigma(b) = 1 \).

Let \( G = (\mathcal{ST}, \mathcal{H}, \Sigma, \Delta, P_0, P_m) \) be a state-tree structure. The SFBC \( \overrightarrow{f} \) under reachability over \( G \) with respect to \( \Sigma \) can be obtained by Algorithm 1.

**Algorithm 1:** The SFBC \( \overrightarrow{f} \) under reachability.

**Input:** A predicate \( P \)

**Output:** The SFBC \( f \) of \( \sigma \) under reachability with respect to the predicate \( P \)

1. Initialize \((\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma^p;\)
   - \( b_0 \not\models P \Rightarrow (\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma(b_0) = 0; \)
   - \( b_0 \models P \land (\exists \sigma \in \Sigma) f^p_\sigma(b_0) = 1 \Rightarrow (\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma(b_0) = 0; \)
   - \( b_0 \models P \land (\exists \sigma \in \Sigma) f^p_\sigma(b_0) = 1 \Rightarrow \overrightarrow{f}_\sigma(b_0) = 1; \)
2. repeat
   - If \( \overrightarrow{f}_\sigma^p(b) = 1 \land \Delta(b, \sigma) = b' \land (\exists \sigma' \in \Sigma) f^p_\sigma(b') = 1, \overrightarrow{f}_\sigma(b') = 1; \)
   - until No other basic state-tree \( b \) satisfies \((\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma(b) = 1; \)

**Remark 3.**

1. According to this algorithm, any state-tree satisfying \( \overrightarrow{f}_\sigma \) is reachable from the initial state-tree under reachability with respect to \( \Sigma \).
2. \( \bigvee_{\sigma \in \Sigma} \overrightarrow{f}_\sigma^p = \overrightarrow{f}^R(G, P). \)

**Example 3.**

As given in Example 1, suppose a predicate \( P \) with \( B_P = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\}, \{R_1, 2, R_2, 2\}\} \) and \( \Sigma_r = \{\alpha, \gamma, \mu, \eta\} \). According to Definition 2, it is easy to know that
- \( \{R_1, 1, R_2, 0\} \models P \land (\exists \sigma \in \Sigma) \overrightarrow{f}_\sigma^p(\{R_1, 0, R_2, 0\}) = 1 \Rightarrow \overrightarrow{f}_\sigma^p(\{R_1, 0, R_2, 0\}) = 1; \)
- \( \{R_1, 0, R_2, 0\} \models P \land (\exists \sigma \in \Sigma) \overrightarrow{f}_\sigma^p(\{R_1, 0, R_2, 0\}) = 1 \Rightarrow \overrightarrow{f}_\sigma^p(\{R_1, 0, R_2, 0\}) = 1; \)
- There is no other basic state-tree \( b \) that satisfy \((\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma(b) = 1; \)

Thus, \( \{R_1, 0, R_2, 0\} \models \overrightarrow{f}_\sigma^p \) and \( \{R_1, 0, R_2, 0\} \models \overrightarrow{f}_\sigma^p. \)

It is easy to know that the reachable predicate \( R(G, P) \) is satisfied by \( \{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\} \) and according to Definition 2, \( \{R_1, 0, R_2, 0\} \models \overrightarrow{f}_\sigma^p \) and \( \{R_1, 0, R_2, 0\} \models \overrightarrow{f}_\sigma^p. \)

Thus, the SFBC \( f \) over any event \( \sigma \) obtained by computing \( R(G, P) \) first is equivalent to the \( \overrightarrow{f}_\sigma^p \).

Similarly, the SFBC \( f \) over \( G \) under coreachability with respect to a predicate \( P \) can be obtained by
- \( P \land P_m = \text{false} \Rightarrow (\forall \sigma \in \Sigma) \overrightarrow{f}_\sigma^p(b) = 0; \)
- \( b_m \models P \land (\exists b \models P)(\sigma \in \Sigma) \Gamma(b_m, \sigma) = b \Rightarrow \overrightarrow{f}_\sigma^p(b) = 0; \)
- \( b_m \models P \land (\exists b \models P)(\sigma \in \Sigma) \Gamma(b_m, \sigma) = b \Rightarrow \overrightarrow{f}_\sigma^p(b) = 1; \)
- \( \overrightarrow{f}_\sigma^p(b) = 1 \land (\exists \sigma \in \Sigma) \Gamma(b, \sigma) = b' \land f^p_\sigma(b') = 1 \Rightarrow \overrightarrow{f}_\sigma^p(b') = 1; \)
- No other basic state-tree \( b \) satisfies \( \overrightarrow{f}_\sigma^p(b) = 1. \)

**Remark 4.**

1. According to this statement, any state-tree satisfying \( \overrightarrow{f}_\sigma \) is coreachable from the marked state-tree \( \overrightarrow{f}_\sigma \).
2. \( \bigvee_{\sigma \in \Sigma} \overrightarrow{f}_\sigma^p = \overrightarrow{f}_{CR(G, P)} \).
3. For any basic state-tree \( b \), if there exists a string \( s = \sigma_1 \sigma_2 \cdots \sigma_n \) satisfying \( \Delta(b, s) \leq P_m \), then \( \overrightarrow{f}_\sigma^p(b) = 1. \) This means that there must exist \( s = \sigma_1 \sigma_2 \cdots \sigma_n \) with \( \overrightarrow{f}_\sigma^p(b) = 1, \)
\[ \overleftarrow{\mu}_2^b(\Delta(b, \sigma_1)) = 1, \ldots, \overleftarrow{\mu}_n^b(\Delta(\cdots(\Delta(b, \sigma_1), \cdots), \sigma_n) = 1. \]

Whether a predicate is coreachable or not can be decided by discussing the SFBC as the following proposition.

**Proposition 4.** A predicate \( P \) is coreachable if

\[ (\forall b \leq P)(\exists \sigma \in \Sigma)f^P_b(b) = 1 \Rightarrow \overleftarrow{\mu}_P^f(b) = 1. \]

**Proof.** To prove this proposition, we need to investigate if \( f^P_b(b) = 1 \) implies that there exists \( \sigma \in \Sigma \) such that \( f^P_b(\Delta(\sigma_1, \cdots, \sigma_n)) = 1 \) and \( \Delta(b, \sigma) \models P \).

It is easy to know that \( (\forall b \leq P)(\exists \sigma \in \Sigma)f^P_b(b) = 1 \Rightarrow \overleftarrow{\mu}_P^f(b) = 1 \) implies \( \Delta(b, \sigma) \models P \).

Further, \( \Delta(b, \sigma_1 \cdots \sigma_n) \models P \) with \( b_m \), which means that \( b \models CR(G, P). \)

\[ \square \]

**Proposition 5.** With a given STS \( G \) and a predicate \( P \), let \( \overrightarrow{f}_P : \mathcal{B}(ST) \rightarrow \{0, 1\} \) be the SFBC under coreachability. For the coreachability predicate \( CR(G, P) \), the SFBC \( f^{CR(G, P)}_P = \overrightarrow{f}_P \).

**Proof.** To prove \( f^{CR(G, P)}_P = \overrightarrow{f}_P \), two steps are necessary.

- Prove \( f^{CR(G, P)}_P \leq \overrightarrow{f}_P \). If for any state-tree \( b, b \models f^{CR(G, P)}_P \Rightarrow b \models CR(G, P) \), then \( \Delta(b, \sigma) \models P \).

This implies that there exists \( s \in \sigma \sigma_1 \cdots \sigma_n \) such that \( \Delta(b, \sigma) \models P \). Thus,

\[ \Rightarrow \overrightarrow{f}_P(b) = 1. \]

- Prove \( \overrightarrow{f}_P \leq f^{CR(G, P)}_P \). If for any state-tree \( b, b \models \overrightarrow{f}_P \Rightarrow b \models P \), then \( \Delta(b, \sigma) \models P \).

This implies that there exists \( s \in \sigma \sigma_1 \cdots \sigma_n \) such that \( \Delta(b, \sigma) \models P \). Thus,

\[ \Rightarrow \overrightarrow{f}_P(b) = 1. \]

Therefore, \( f^{CR(G, P)}_P = \overrightarrow{f}_P \) is proved.

\[ \square \]

Let \( G = (ST, \mathcal{H}, \Sigma, \Delta, P_0, P_m) \) be a state-tree structure. The SFBC \( \overrightarrow{f} \) under coreachability over \( G \) with respect to \( \Sigma \) can be obtained by Algorithm 2. An example is given to illustrate the algorithm.

**Example 4.** As given in Example 1, suppose a predicate \( P \) with \( B_P = \{R_{1,0}, R_{2,0}, R_{1,1}, R_{2,0}, \}, \{R_{1,0}, R_{2,1}, \} \), and \( \Sigma = \{\alpha, \gamma, \mu, \eta\} \). According to the definition of coreachable SFBC, it is easy to know that

- \( \{R_{1,0}, R_{2,0}\} \models P \) and \( \Gamma((R_{1,0}, R_{2,0}), \eta) = \{R_{1,0}, R_{2,2}\} \) and \( \{R_{1,0}, R_{2,2}\} \equiv \langle \{R_{1,0}, R_{2,2}\}, \rangle = 0; \)
Although no other basic-state-tree $b$ satisfies $\{R_{1,0}, R_{2,0}\} \models P \& \Delta((R_{1,0}, R_{2,0}), \mu) = \{R_{1,2}, R_{2,0}\} \& \{R_{1,2}, R_{2,0}\} \models P \Rightarrow f^P_P((R_{1,2}, R_{2,0})) = 0$;

- $\{M_{1,0}, M_{2,0}\} \models P \& \Delta((R_{1,0}, R_{2,0}), \beta) = \{R_{1,1}, R_{2,1}\} \& \{R_{1,1}, R_{2,1}\} \models P \Rightarrow f^P_P((R_{1,1}, R_{2,1})) = 0$;
- $f^P_P((R_{1,2}, R_{2,2})) = 1 \& \Delta((R_{1,2}, R_{2,2}), \lambda) = \{R_{1,1}, R_{2,1}\} \& \{R_{1,1}, R_{2,1}\} \models P \Rightarrow f^P_P((R_{1,1}, R_{2,1})) = 0$;
- Similarly, no other basic-state-tree $b$ satisfies $(\forall \sigma \in \Sigma) f^P_P(b) = 1$.

Thus, for any event $\sigma$, there is $f^P_P(b) = 0$.

It is easy to know that the coreachable predicate $\text{CR}(G, P)$ is satisfied by $\{M_{1,0}, M_{2,0}\}$ and according to Definition 2, for any event $\sigma$, there is $f^P_P(b) = 0$.

Thus, the SFBC $f$ over any event $\sigma$ obtained by computing $\text{CR}(G, P)$ first is equivalent to the $f^P_P$.

4.2. Sfbc under Weakly Controllability

The weakest liberal precondition is the predicate transformer $M_\sigma : \text{Pred}(ST) \rightarrow \text{Pred}(ST)$ defined as follows:

$$b \models M_\sigma(P) \iff \Delta(b, \sigma) \models P.$$  

A predicate $P \in \text{Pred}(ST)$ is weakly controllable with respect to $G$ if

$$(\forall \sigma \in \Sigma_u) P \leq M_\sigma(P).$$

To achieve the aim of resolving the nonblocking control problem, the weak controllability should be investigated. Thus, the definition and property of SFBC under weak controllability are given in this part.

**Proposition 6.** A predicate $P$ is weakly controllable with respect to $G$ if

$$(\forall \sigma \in \Sigma_u)b \models P \Rightarrow f^P_P(b) = 1.$$  

**Proof.** Suppose an STS $G := (ST, \mathcal{H}, \Sigma, \Delta^f, P^f_0, P_m)$ with the controlled STS as $G^f := (ST, \mathcal{H}, \Sigma, \Delta^f, P^f_0, P_m)$ for the closed system which is supervised and controlled by the SFBC $f$.

(IF) Assume that the predicate $P \in \text{Pred}(ST)$ is weakly controllable. Thus, $(\forall \sigma \in \Sigma_u) P \leq M_\sigma(P) \Rightarrow (\forall \sigma \in \Sigma_u) b \models P \Rightarrow b \models M_\sigma(P) \Rightarrow (\forall \sigma \in \Sigma_u) b \models P \Rightarrow \Delta(b, \sigma) \models P$. According to the definition, $f^P_P(b) = 1$. Therefore, $(\forall \sigma \in \Sigma_u)b \models P \Rightarrow f^P_P(b) = 1$ is proved.
Additionally, the SFBC of any event under controllability is denoted by 

\[
\text{Algorithm 3: The SFBC of any event under controllability}
\]

**Input:** A predicate \( P \) and \( (\forall \sigma \in \Sigma) f^P_{\sigma} \).

**Output:** The SFBC \( f \) with respect to any event for under controllability.

1. Initialize \( i = 0, K_0 = \neg P \) and \( (\forall \sigma \in \Sigma) f^{K_0}_{\sigma} = f^P_{\sigma} \);
2. repeat
3. \( K_{i+1} = K_i \);
4. If \( b \models K_0 \) & \((\exists \sigma \in \Sigma) \Delta(b', \sigma) = b \) & \( f^{K_0}_{\sigma}(b') = 0 \) & \( (\forall \sigma' \in \Sigma) \Delta(b', \sigma') = b' \), then \( f^{K_{i+1}}_{\sigma}(b') = 0 \) & \( K_{i+1} = K_{i+1} \lor P_b \);
5. \( i = i + 1 \);
6. until \( f^{K_{i+1}}_{\sigma} = f^{K_i}_{\sigma} \);
7. Compute the SFBC \( \underline{f}^{\sigma \sup \mathcal{CP}(P)} \) according to Algorithm 1.

**Proposition 7.** For a predicate \( P \), the SFBC \( f \) under controllability obtained by computing the predicate \( R(G, \sup \mathcal{CP}(P)) \) first is equivalent to \( \underline{f}^{\sigma \sup \mathcal{CP}(P)} \) by Algorithm 3, i.e., the diagram commutes.

**Proof.** To prove that the diagram commutes, we need to show that \( \underline{f}^{\sigma \sup \mathcal{CP}(P)} \) obtained by Algorithm 3 is equivalent to the \( f^{R(G, \sup \mathcal{CP}(P))} \). Thus, if or any \( \underline{f}^{\sigma \sup \mathcal{CP}(P)}(b) = 1 \Rightarrow f^{R(G, \sup \mathcal{CP}(P))}(b) = 1 \), the proposition is true. Since \( \underline{f}^{\sigma \sup \mathcal{CP}(P)}(b) = 1, b \models P \) & \( (\sigma \in \Sigma) \Delta(b, \sigma) \models P \) and

\[
\begin{align*}
\underline{f}^{\sup \mathcal{CP}(P)}(b_h) &= 1, \\
\underline{f}^{\sup \mathcal{CP}(P)}(b_{n-1}) &= 1, \\
\ldots, \\
\underline{f}^{\sup \mathcal{CP}(P)}(b_0) &= 1.
\end{align*}
\]

Thus, it is easy to know \( b \models P \) & \( (\sigma \in \Sigma) \Delta(b, \sigma) \models P \), which means that \( b \models \sup \mathcal{CP}(P) \).

Additionally,

\[
\begin{align*}
\underline{f}^{\sup \mathcal{CP}(P)}(b_h) &= 1 \Rightarrow b_h \models P \& (\sigma \in \Sigma) \Delta(b_h, \sigma) = b \models P \Rightarrow b_h \models \sup \mathcal{CP}(P), \\
\underline{f}^{\sup \mathcal{CP}(P)}(b_{n-1}) &= 1 \Rightarrow b_{n-1} \models P \& (\sigma \in \Sigma) \Delta(b_{n-1}, \sigma_{n-1}) = b_{n-1} \models P \Rightarrow b_{n-1} \models \sup \mathcal{CP}(P), \\
\ldots
\end{align*}
\]

By the definition of the reachability predicate, \( b_0 \models R(G, \sup \mathcal{CP}(P)), b_1 \models R(G, \sup \mathcal{CP}(P)), \ldots, b_n \models R(G, \sup \mathcal{CP}(P)) \). Thus, \( f^{R(G, \sup \mathcal{CP}(P))}(b) = 1 \) is obvious. Until now, the proposition is proved. \( \square \)
Note that the diagram in Figure 3 shows the relation between the predicate \( P \) and the SFBC over the reachability predicate \( f^{R(G, \sup \mathbb{CP}(P))} \). According to Proposition 7, for a predicate \( P \), \( f^{R(G, \sup \mathbb{CP}(P))} \) can be obtained by computing \( R(G, \sup \mathbb{CP}(P)) \) first and then SFBC over \( R(G, \sup \mathbb{CP}(P)) \). Similarly, \( f^{R(G, \sup \mathbb{CP}(P))} \) can also be obtained if \( f^P \) is computed first. The proof of this diagram in Figure 3 shows the feasibility of Algorithm 3.

\[
\begin{align*}
P & \xrightarrow{\neg P, \ R(G, \ast)} R(G, \sup \mathbb{CP}(P)) \\
& \xrightarrow{\ast} R(G, \sup \mathbb{CP}(P))
\end{align*}
\]

Figure 3. The relationship between the SFBC and the weakly controllable predicate.

Example 5. As given in Example 1, suppose a predicate \( P \) with \( B_P = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\}, \{R_1, 2, R_2, 2\}\} \) and \( \Sigma_c = \{\alpha, \gamma, \mu, \eta\} \). According to Algorithm 3, it is easy to know that for \( B_P = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\}, \{R_1, 2, R_2, 2\}\} \),

- Since \( K_0 = \neg P, \{\{R_1, 1, R_2, 2\}, \{R_1, 2, R_2, 0\}, \{R_1, 0, R_2, 2\}\} \models K; 
- For \( \{R_1, 1, R_2, 1\}, (\exists \sigma \in \Sigma_a) \Delta(b', \sigma) = b, K_1 = K_0 \land (\forall \sigma' \in \Sigma) f^\eta_{K_1}(b') = 0; 
- For \( \{R_1, 2, R_2, 0\}, (\exists \sigma \in \Sigma_a) \Delta(b', \sigma) = b, K_1 = K_0 \land (\forall \sigma' \in \Sigma) f^\eta_{K_1}(b') = 0; 
- And for \( \{R_1, 0, R_2, 2\}, (\exists \sigma \in \Sigma_a) \Delta(b', \sigma) = b, K_1 = K_0 \land (\forall \sigma' \in \Sigma) f^\eta_{K_1}(b') = 0; 

Thus, for any event \( \sigma \), \( f^\eta_{K_1}(\{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}\}) = 1 \) and \( f^\eta_{K_2}(\{\{R_1, 1, R_2, 0\}\}) = 1 \).

It is easy to know that \( sup \mathbb{CP}(P) \) is satisfied by \( \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\}, \{R_1, 2, R_2, 2\}\} \) and \( R(G, sup \mathbb{CP}(P)) \) is satisfied by \( \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}, \{R_1, 0, R_2, 1\}\} \). According to Definition 2, for any event \( \sigma \), \( f^\eta_{R(G, sup \mathbb{CP}(P))}(\{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}\}) = 1 \) and \( f^\eta_{R(G, sup \mathbb{CP}(P))}(\{\{R_1, 0, R_2, 0\}\}) = 1 \).

Thus, the SFBC \( f \) over any event \( \sigma \) obtained by computing \( R(G, sup \mathbb{CP}(P)) \) first is equivalent to the \( f^\eta_{sup \mathbb{CP}(P)} \).

5. Nonblocking SFbc for Sts

According to Theorem 3.3 in [17], a nonblocking SFBC \( f \) for \( G \) with \( R(G, true) = R(G, P) \) exists if \( P \) is weakly controllable and reachable, where \( G^f := (ST, \mathcal{H}, \Sigma, \Delta^f, P^f_0, P^f_m) \) and \( P^f_0 = P \land P_0 \). Notice that our sup\( C^2 \mathbb{CP}(P) \) is the optimal supervisor that is obtained instead of nonblocking, where sup\( C^2 \mathbb{CP}(P) \) is the supremal element of \( C^2 \mathbb{CP}(P) \). Furthermore, \( R(G, sup \mathbb{CP}(P)) \) is the supremal element of \( sup \mathbb{BC}(P) \), where

\[
sup \mathbb{BC}(P) := \{ K \in \text{Pred}(ST) | K \leq P \land K \text{ weakly controllable} \land K \text{ nonblocking} \}
\]

This means that both controllers agree on their reachable subpredicates and sup\( C^2 \mathbb{CP}(P) \) is another optimal nonblocking controller. \( R(G, sup \mathbb{CP}(P)) \) shows the final controlled behavior. Therefore, the control functions for \( R(G, sup \mathbb{CP}(P)) \) are necessary to be discussed.

Proposition 8. For a predicate \( P \), \( \forall K \in C^2 \mathbb{CP}(P) \) if

- \( b \models K \Rightarrow (\sigma \in \Sigma_a) f^K_{\hat{K}}(b) = 1; \) and
- \( b \models K \Rightarrow (\exists \sigma \in \Sigma) \exists \hat{K} f^K_{\hat{K}}(b) = 1. \)

Proof. The proposition can be proved by the following two steps:

1. If \( b \models K \Rightarrow (\exists \sigma \in \Sigma) \exists \hat{K} f^K_{\hat{K}}(b) = 1, \) \( K \) is coreachable (i.e., \( K \leq CR(G, K) \)). For any \( b \models K \), we need to prove \( b \models CR(G, K) \).
According to the definition of the coreachability predicate, if \( b_m \models K, b_m \models CR(G, K) \), obviously.

2. If \( b \models K \), there must exist \((\exists \sigma \in \Sigma)^{(f)}_{\sigma}(b) = 1\). This implies that \((\exists \sigma \in \Sigma)\Gamma(b', \sigma') = b\) and \( b' \models K \& (\exists \sigma_1 \in \Sigma)^{(f)}_{\sigma_1}(b') = 1\). Iteratively, a string \( s = \sigma_1 \sigma_2 \cdots \sigma_n \) can be obtained, which makes \( \Gamma(b_m, s) = b \) hold. Thus, \( b \models CR(G, K) \).

It is proved that if \( b \models K \Rightarrow (\exists \sigma \in \Sigma)^{(f)}_{\sigma}(b) = 1 \), \( K \) is coreachable.

2. If \( b \models K \Rightarrow (\exists \sigma \in \Sigma)^{(f)}_{\sigma}(b) = 1, K \) is weakly controllable. To prove this, \((\forall \sigma \in \Sigma_0)K \leq M_\sigma(K)\) should be discussed. For any \( b \models K \), \((\forall \sigma \in \Sigma_0)^{(f)}_{\sigma}(b) = 1\), which means that \( \Delta(b, \sigma') \) \& \( \Delta(b, \sigma) \models K \). Thus, \( b \models M_\sigma(K) \). Thus, \( K \) is proved to be weakly controllable. Therefore, the proposition is proved to be true.

**Algorithm 4:** The SFBC \( f \) under controllability and coreachability.

**Input:** A predicate \( P \)

**Output:** The SFBC \( f \) with respect to any event

1. Initialize \( i = 0, \neg K_i = P \), and \((\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^P_{\sigma} \);
2. repeat
   3. \( i = i + 2; \)
   4. Let \( j = 0 \) and \( K_j = K_i; \)
   5. for \( f^K_{j+1} \neq f^K_{j+1} \) do
      6. \( K_{j+1} = K_j; \)
      7. if \( b \models K_j \& (\exists \sigma \in \Sigma_0)\Delta(b', \sigma') = b \& f^{-K_j}_{\sigma}(b) = 0 \& (\forall \sigma' \in \Sigma)\Delta(b'', \sigma') = b', f^{-K_j}_{\sigma}(b') = 0 \& K_{j+1} = K_{j+1} \vee P_b; i = j + 1; \)
      8. let \( f^{-K_j}_{\sigma} = f^{-K_j}_{\sigma} \);
   9. Compute the coreachable SFBC \( f^{-K_j+1}_{\sigma} \) for each \( \sigma \) as follows:
      10. \( b_m \models \neg K_{j+1} \Rightarrow (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+1}}_{\sigma} = 0 \& \neg K_{j+2} = f^{K_{j+1}}_{\sigma} \);
      11. \( b_m \models \neg K_{j+1} \& b \models \neg K_{j+1} \& (\exists \sigma \in \Sigma)\Gamma(b_m, \sigma) = b \Rightarrow (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+1}}_{\sigma} = 0 \& \neg K_{j+2} = P_m \vee P_b; \)
      12. \( f^{-K_{j+1}}_{\sigma}(b') = 0 \& (\exists \sigma \in \Sigma)\Gamma(b', \sigma') = b \Rightarrow (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+2}}_{\sigma}(b) = 0 \& \neg K_{j+2} = P_m \vee P_b; \)
      13. \( f^{-K_{j+2}}_{\sigma}(b') = 0 \& (\exists \sigma \in \Sigma)\Gamma(b', \sigma') = b \Rightarrow (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+2}}_{\sigma}(b) = 0 \& \neg K_{j+2} = P_m \vee P_b; \)
      14. \( f^{-K_{j+2}}_{\sigma}(b') = 0 \& (\exists \sigma \in \Sigma)\Gamma(b', \sigma') = b \Rightarrow (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+2}}_{\sigma}(b) = 0 \& \neg K_{j+2} = P_m \vee P_b; \)
      15. No other basic-state-tree \( b \) satisfies \((\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+2}}_{\sigma}(b) = 1 \).
16. \( (\forall \sigma \in \Sigma)^{(f)}_{\sigma} = f^{K_{j+2}}_{\sigma} \);

Compute the SFBC under reachability \( f^{K_{j+2}}_{\sigma} \) according to Algorithm 1.

According to Theorem 3.3 in [17], a nonblocking SFBC \( f \) for \( G \) with \( R(G^f, true) = R(G, P) \) exists if \( P \) is weakly controllable and coreachable, where \( G^f := (ST, \mathcal{H}, \Sigma, \Delta^f, P^f_0, P^f_m) \) and \( P^f_0 = P \wedge P_0 \). Since \( P \) is weakly controllable, \( R(G, P) \) is controllable. Thus, the nonblocking SFBC \( f \) under controllability and coreachability shows the control behaviors of the supervisor \( P \), which can be computed according to Algorithm 4.

**Remark 5.** According to this algorithm, the nonblocking SFBC \( f \) for \( G \) with \( R(G^f, true) = R(G, P) \) can be obtained by the following:

1. For predicate \( P \), compute the SFBC \( f \) under weak controllability by analyzing the SFBC of the uncontrollable events, iteratively, to ensure that the uncontrollable event is not disabled;
2. Then, compute the SFBC under coreachability by revising the SFBC of any events which cannot lead the system to the marked state-tree.
3. If the SFBC of any event is not revised, go to Step 4; if not, back to Step 1;
4. Finally, compute the SFBC under reachability by revising the SFBC of any events which cannot reached by the initial state-tree.

So far, so good. However, we still need to answer an important question. Notice that the $supC^2P(P)$ is the optimal supervisor with $R(G, supC^2P(P)) = R(G, supNBCP(P))$ and a nonblocking $f$ with $R(G^f, true) = R(G, supC^2P(P))$ exists. The nonblocking SFBC $f$ of events can be obtained by

$$f_{\sigma}^{R(G, supC^2P(P))}(b) := 1 \text{ iff } R(G, supC^2P(P))(b) = 1 \& \Delta(b, \sigma) \& \Delta(b, \sigma') \models R(G, supC^2P(P)).$$

So the question is, does $supC^2P(P)$ own the same control effect over $G$ as $supNBCP(P)$?

**Proposition 9.** For a predicate $P$, the SFBC $f$ wrt. $supC^2P(P)$ obtained by computing the predicate $supC^2P(P)$ first is equivalent to $f_{\sigma}^{supC^2P(P)}$ by Algorithm 4, i.e., the diagram in Figure 4 commutes.

$$P \xrightarrow{\neg [P] R(G, P) R(G, \cdot)} R(G, supC^2P(P))$$

$$f \downarrow \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \downarrow f$$

$$\xrightarrow{f^{supC^2P(P)}_G} \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \xrightarrow{f} R(G, supC^2P(P)).$$

**Figure 4.** The relationship between $f_{\sigma}^{supC^2P(P)}$ and $R(G, supC^2P(P))$.

**Proof.** To prove that the diagram commutes, we need to show that $f_{\sigma}^{supC^2P(P)}$ obtained by Algorithm 4 is equivalent to the $f_{\sigma}^{R(G, supC^2P(P))}(b) = 1 \Rightarrow f_{\sigma}^{R(G, supC^2P(P))}(b) = 1$, the proposition is true. Since $f_{\sigma}^{supC^2P(P)}(b) = 1$, $b \models P \& (\sigma \in \Sigma) \Delta(b, \sigma) \models P$ and

$$(\exists s = \sigma_0\sigma_2 \cdots \sigma_n) f_{\sigma}^{supC^2P}(b) = 1 \& \Delta(b, s) = b_m,$$

$$(\exists s' = \sigma_0'\sigma_2' \cdots \sigma_n') f_{\sigma'}^{supC^2P}(b_0) = 1 \& \Delta(b_0, s') = b.$$

Thus, it is easy to know $b \models P \& (\sigma \in \Sigma) \Delta(b, \sigma) \models P \& \Delta(b, s) = b_m$, which means that $b \models supC^2P(P)$. Additionally,

$$f_{\sigma_n}^{supC^2P(P)}(b_n) = 1 \Rightarrow b_n \models P \& (\sigma' \in \Sigma) \Delta(b_n, \sigma_n) = b \models P \& \Delta(b_n, s) = b_m \Rightarrow$$

$$b_n \models supC^2P(P),$$

$$f_{\sigma_{n-1}}^{supC^2P(P)}(b_{n-1}) = 1 \Rightarrow b_{n-1} \models P \& (\sigma'_{n-1} \in \Sigma) \Delta(b_{n-1}, \sigma_{n-1}) = b_{n} \models P \& \Delta(b_{n}, \sigma_n) = b \Rightarrow$$

$$b_{n-1} \models supC^2P(P),$$

$$\cdots$$

$$f_{\sigma_0}^{supC^2P(P)}(b_0) = 1 \Rightarrow b_0 \models P \& (\sigma_0' \in \Sigma) \Delta(b_0, \sigma'_0) = b_1 \models P \& \Delta(b_0, \sigma'_0) = b_1 \Rightarrow$$

$$b_0 \models supC^2P(P).$$

By the definition of the reachability predicate, $b_0 \models R(G, supC^2P(P))$, $b_1 \models R(G, supC^2P(P))$, $\cdots$, $b_n \models R(G, supC^2P(P))$, $b \models R(G, supC^2P(P))$. Thus, $f_{\sigma}^{R(G, supC^2P(P))}(b) = 1$ is obvious. Until now, the proposition is proved. \(\square\)
Notice that the supC^2P(P) is the optimal supervisor that is coreachable instead of nonblocking. For NBCP(P), R(G, supC^2P(P)) = R(G, supNBCP(P)) is proved, which implies that for any event σ, f_0^(R(G, supC^2P(P))) = f_0^(R(G, supNBCP(P))). Thus, the nonblocking SFBC f for G with R(G', true) = R(G, P) can be obtained by Algorithm 4.

**Example 6.** As given in Example 1, suppose a predicate P' with B_P' = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}\}. According to Algorithm 4, it is easy to know that for B_P' = \{\{R_1, 0, R_2, 0\}, \{R_1, 1, R_2, 0\}\},

1. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking. For each
   \( \{\{R_1, 0, R_2, 0\}\} \)

   - For i = 0, \( K_0 = \neg \sigma' = \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \);
   - For \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   Compute the coreachable SFBC \( f^K \) for each NBCP nonblocking:

   a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

      - For \( \neg K_1, \{\{R_1, 0, R_2, 0\}\} \models f^K_\sigma \) for each NBCP nonblocking:

        a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   b. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   c. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   d. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   e. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   f. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   g. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).

   h. Compute the reachable SFBC \( f^K \) for each NBCP nonblocking:

      a. \( \{\{R_1, 0, R_2, 0\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         a. \( \{\{R_1, 0, R_2, 1\}, \{R_1, 1, R_2, 2\}\} \) \( \models f^K_\sigma \) for each NBCP nonblocking:

         b. \( (\forall b) \models K_0(b) \) \( (\exists \sigma \in \Sigma_u) f^K_\sigma(b) = 1 \Rightarrow K_1 = \neg \sigma \) \( \models f^K_\sigma \).
• Proposition 1 shows the relation between the SFBC for two subpredicates of \( P \).
• Propositions 6 and 7 show the reasonability of Algorithm 3, which is proposed to compute the SFBC under reachability, and Example 3 explains each step of Algorithm 1;
• Propositions 4 and 5 show the reasonability of Algorithm 2, which is proposed to compute the SFBC under weak controllability, and Example 4 explains each step of Algorithm 2;
• Propositions 2 and 3 show the reasonability of Algorithm 1, which is proposed to compute the SFBC under coreachability, and Example 3 explains each step of Algorithm 1;
• Proposition 1 shows the relation between the SFBC for two subpredicates of \( P \);
• Propositions 4 and 5 show the reasonability of Algorithm 2, which is proposed to compute the SFBC under weak controllability, and Example 5 explains each step of Algorithm 2;
• Propositions 6 and 7 show the reasonability of Algorithm 3, which is proposed to compute the SFBC under weak controllability and coreachability, and Example 6 explains each step of Algorithm 4; the computation of Algorithm 4 requires the results of Algorithms 1–3, and by Algorithm 4, the control functions of the supervisory control problem can be obtained.

Overall, nine propositions and four algorithms are proposed, and the relation between these propositions and algorithms are displayed in Table 1. As shown in this table, we have the following:

• Definition 1 is proposed to define the SFBC of an event for a predicate and Example 2 illustrates this definition;
• Proposition 1 shows the relation between the SFBC for two subpredicates of \( P \);
Table 1. Propositions, algorithms, and examples in this paper.

| Propositions/Definitions | Algorithms | Examples |
|--------------------------|------------|----------|
| Definition 1: SFBC of an event for predicate | | Example 2 |
| Proposition 1: property of SFBC | Algorithm 1 | Example 3 |
| Proposition 2: reachability on SFBC; Proposition 3: the relation between reachability subpredicate of P and the SFBC | | |
| Proposition 4: coreachability on SFBC; Proposition 5: the relation between coreachability subpredicate of P and the SFBC | Algorithm 2 | Example 4 |
| Proposition 6: weak controllability on SFBC; Proposition 7: the relation between weak controllability subpredicate of P and the SFBC | Algorithm 3 | Example 5 |
| Proposition 8: weak controllability and coreachability on SFBC; Proposition 9: the relation between weak controllability and coreachability subpredicate of P and the SFBC | Algorithm 4 | Example 6 |

6. Examples

6.1. Transfer Line

Chapter 4 in [14] proposes an industrial “transmission line” composed of two machines $TM_1$, $TM_2$, and a testing unit $TU$, which is connected by buffers $B_1$ and $B_2$, as is depicted by Figure 5. If the workpiece is accepted after $TU$ test, it can be released; otherwise, it will be returned to $B_1$ and reprocessed by $TM_2$. Because of this, the structure has the function of “material feedback”. The specification can help $B_1$ and $B_2$ avoid underflow as well as overflow. The capacities of $B_1$ and $B_2$ are assumed to be 1. The controlled events are 1, 3, and 5, while the uncontrolled events are 2, 4, 6, and 8.

As its STS model is depicted in Figure 5, $TL$ is an AND superstate and its components $TM_1$, $TM_2$, $TU$, $B_1$, and $B_2$ are parallel, implying that the system work at these states simultaneously. $TM_1$ is an OR superstate and its components $TM_{1_1}$ and $TM_{1_2}$ are exclusive, which means only one of $TM_{1_1}$ and $TM_{1_2}$ can be reached at a time. So as $TM_2$, $TU$, $B_1$, and $B_2$. The occurrence of events promotes the transition between these states. For the shared events, such as 2, if 2 occurs, the states $TM_{1_1}$ and $B_{1_0}$ transfers to $TM_{1_2}$ and $B_{1_1}$ simultaneously. Other events can always occur if the system reaches the trigger states. For example, event 1 can occur once $TM_1$ stays at $TM_{1_0}$. Thus, to display the behavior of the system, $\Delta$ can be given as follows:

\[
\Delta(\{TM_{1_0}, TM_{2_0}, TU_{0}, B_{1_0}, B_{2_0}\}, 1) = \{TM_{1_1}, TM_{2_0}, TU_{0}, B_{1_0}, B_{2_0}\}
\]
\[
\Delta(\{TM_{1_1}, TM_{2_0}, TU_{0}, B_{1_0}, B_{2_0}\}, 2) = \{TM_{1_0}, TM_{2_0}, TU_{1}, B_{1_0}, B_{2_0}\}
\]
\[
\Delta(\{TM_{1_0}, TM_{2_0}, TU_{1}, B_{1_0}, B_{2_0}\}, 3) = \{TM_{1_0}, TM_{2_1}, TU_{0}, B_{1_0}, B_{2_0}\}
\]
\[
\Delta(\{TM_{1_0}, TM_{2_1}, TU_{0}, B_{1_0}, B_{2_0}\}, 4) = \{TM_{1_0}, TM_{2_0}, TU_{0}, B_{1_0}, B_{2_1}\}
\]
\[
\Delta(\{TM_{1_0}, TM_{2_1}, TU_{0}, B_{1_0}, B_{2_0}\}, 1) = \{TM_{1_1}, TM_{2_1}, TU_{0}, B_{1_0}, B_{2_0}\}
\]
\[\ldots\]
The specification of the transfer line is given by two holons ($B_1$ and $B_2$). Based on the SFBC of each event, the control functions of 1, 3, and 5 can be given by Figure 6. For this example, $B_P = \{\{TM_1_0,TM_2_0,TU_0,B_1_0,B_2_0\}, \{TM_1_1,TM_2_0,TU_0,B_1_0,B_2_0\}, \{TM_1_0,TM_2_1,TU_0,B_1_0,B_2_0\}, \{TM_1_0,TM_2_0,TU_1,B_1_0,B_2_0\}, \{TM_1_0,TM_2_0,TU_0,B_1_1,B_2_0\}, \{TM_1_0,TM_2_0,TU_0,B_1_0,B_2_1\}\}$. According to Algorithm 4, the nonblocking SFBC of the events can be obtained.

$$f_1^P(\{TM_1_0,TM_2_0,TU_0,B_1_0,B_2_0\}) = 1,$$
$$f_2^P(\{TM_1_1,TM_2_0,TU_0,B_1_0,B_2_0\}) = 1,$$
$$f_3^P(\{TM_1_0,TM_2_0,TU_0,B_1_1,B_2_0\}) = 1,$$
$$f_4^P(\{TM_1_0,TM_2_1,TU_0,B_1_0,B_2_0\}) = 1,$$
$$f_5^P(\{TM_1_0,TM_2_0,TU_0,B_1_0,B_2_1\}) = 1,$$
$$f_6^P(\{TM_1_0,TM_2_0,TU_1,B_1_0,B_2_0\}) = 1,$$ and
$$f_8^P(\{TM_1_0,TM_2_0,TU_1,B_1_0,B_2_0\}) = 1.$$

Therefore, the control functions of the controllable events 1, 3, and 5 can be given by Figure 6.
Figure 6. Control functions of 1, 3, and 5.

Remark 6. It is easy to find that the control functions obtained on the basis of SFBC are the same as the ones obtained on the predicates in Chapter 4 in [17]. In [17], the supremal weakly controllable and coreachable subpredicate needs to be calculated by analyzing the states of the system, and then the control functions are obtained in view of this subpredicate. For this example, the supremal subpredicate is \(\{\{TM_1 \_0, TM_2 \_0, TU_0, B_1 \_0, B_2 \_0\}, \{TM_1 \_1, TM_2 \_0, TU_0, B_1 \_0, B_2 \_0\}, \{TM_1 \_0, TM_2 \_0, TU_0, B_1 \_1, B_2 \_0\}\). In view of the subpredicate and the transition function \(\Delta\), the control functions can be obtained by studying the enabled events of each state. However, by using Algorithm 4, we only need to focus on the SFBC of the uncontrollable events, which are disabled. Then, the control functions can be obtained directly, which helps to improve the calculation efficiency to a certain extent. The use of SFBC makes the computation process of the supervisory control problem more flexible since we can disable the event by letting \(f_\sigma(b) = 0\), such as the following example.

6.2. Small Factory

This example shows how to achieve the aim of resolving the control problem of the Small Factory based on SFBC. First, the STS model is shown in Figure 7, where the BUF contains a buffer with one slot. The two states of SM_1 are represented as 0 and 1, where 0 stands for “idle” and 1 for “working”. In the initial state, the buffer is empty. SM_1 can work from the infinite input bin (Event 1), after which M1 completes the work cycle (Event 2). The operation mechanism of SM_2 is basically the same. Odd numbers represent controllable events and even numbers represent uncontrollable ones. After that, BUF needs to complete the following two tasks: first, when event 2 takes place, the workpiece will be placed in the buffer; secondly, when event 3 takes place, the workpiece is transferred to SM_2.
The overflow and underflow can be taken care of here. If the capacity of the buffer is one, to achieve this, we need to (1) disable event 3 at state 0 of the holon BUF and (2) disallow event 2 to occur at state 1 of the holon BUF. (1) is represented by the holon BUF, and (2) can be written as \( f_2^2(\{\text{SM}_1, \text{BUF}_1\}) = 0 \). The procedure to resolve this problem is displayed as follows.

1. \( B_p = \{\{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}, \{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}\} \)

2. \( f_2^2(\{\text{SM}_1, \text{BUF}_1\}) = 0 \) implies that \( f_2^2(\{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}) = 0 \) and \( f_2^2(\{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}) = 0 \)

3. For event 1, \( f_2^3(\{M_1, M_2, \text{BUF}_0\}) = 1 \), \( f_2^3(\{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}) = 1 \), \( f_2^3(\{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}) = 1 \), and \( f_2^3(\{\text{SM}_1, \text{SM}_2, \text{BUF}_0\}) = 1 \)

4. According to Algorithm 4, the nonblocking SFBC \( f \) can be got by

- Under weak controllability, since \( f_2^3(\{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}) = 1 \) & \( \Gamma(\{\text{SM}_1, \text{SM}_2, \text{BUF}_1\}) \)

- Under coreachability,
\(f_2^{-K_2}(\{SM\_1\_1, SM\_2\_0, BUF\_0\}) = 1 \& \{SM\_1\_1, SM\_2\_0, BUF\_0\} \models \neg K_2;\)
\(\{SM\_1\_1, SM\_2\_0, BUF\_0\} \models \neg K_2 \& f_1^{-K_1}(\{SM\_1\_0, SM\_2\_0, BUF\_1\}) = 1 \Rightarrow f_1^{-K_2}(\{SM\_1\_0, SM\_2\_0, BUF\_0\}) = 1 \& \{SM\_1\_0, SM\_2\_0, BUF\_0\} \models \neg K_2;\)
\(\{SM\_1\_1, SM\_2\_0, BUF\_0\} \models \neg K_2 \& f_4^{-K_1}(\{SM\_1\_1, SM\_2\_1, BUF\_0\}) = 1 \Rightarrow f_4^{-K_2}(\{SM\_1\_1, SM\_2\_1, BUF\_0\}) = 1 \& \{SM\_1\_1, SM\_2\_1, BUF\_0\} \models \neg K_2;\)
\(\{SM\_1\_0, SM\_2\_0, BUF\_1\} \models \neg K_2 \& f_4^{-K_1}(\{SM\_1\_0, SM\_2\_1, BUF\_1\}) = 1 \Rightarrow f_4^{-K_2}(\{SM\_1\_0, SM\_2\_1, BUF\_1\}) = 1 \& \{SM\_1\_0, SM\_2\_1, BUF\_1\} \models \neg K_2;\)
\(\{SM\_1\_1, SM\_2\_1, BUF\_1\} \models \neg K_2 \& f_2^{-K_1}(\{SM\_1\_1, SM\_2\_1, BUF\_0\}) = 1 \Rightarrow f_2^{-K_2}(\{SM\_1\_1, SM\_2\_1, BUF\_0\}) = 1 \& \{SM\_1\_1, SM\_2\_1, BUF\_0\} \models \neg K_2;\)

and no other \(\sigma \in \Sigma\) \(f^{-K_2}_\sigma (b) = 1;\)

- It is easy to find that the SFBC \(f^{-K_2}\) satisfies the weakly controllability and coreachability. The iteration stops.
- Under reachability, similar to the calculation of coreachability, \(f^{-K_2}_\sigma\) can be computed.

\(\vec{f}_1^{-K_2}(\{SM\_1\_0, SM\_2\_0, BUF\_0\}) = 1;\)
\(\vec{f}_3^{-K_2}(\{SM\_1\_0, SM\_2\_0, BUF\_1\}) = 1;\)
\(\vec{f}_2^{-K_2}(\{SM\_1\_1, SM\_2\_0, BUF\_0\}) = 1;\)
\(\vec{f}_1^{-K_2}(\{SM\_1\_0, SM\_2\_1, BUF\_0\}) = 1;\)
\(\vec{f}_4^{-K_2}(\{SM\_1\_1, SM\_2\_1, BUF\_0\}) = 1;\)
\(\vec{f}_4^{-K_2}(\{SM\_1\_0, SM\_2\_1, BUF\_1\}) = 1;\)
\(\vec{f}_2^{-K_2}(\{SM\_1\_1, SM\_2\_1, BUF\_1\}) = 1;\)

and no other \(\sigma \in \Sigma\) \(\vec{f}_\sigma^{-K_2}(b) = 1;\)

Therefore, the control functions of 1 and 3 can be given by \(B_1 = \{SM\_1\_0, SM\_2\_0, BUF\_0, SM\_1\_1, SM\_2\_1, BUF\_0\}\) and \(B_3 = \{SM\_1\_0, SM\_2\_0, BUF\_1\}\), which is depicted in Figure 8.

![Figure 8. Control functions of 1 and 3.](image-url)
trees are not allowed to occur. In this paper, the specification is given by SFBC, which makes the calculation easier since we can analyze the SFBC of the uncontrollable events instead of analyzing the exclusive states. For this example, the specification can be given by $f_2^2(\{SM_1, BUF_1\}) = 0$. The computation will focus on $f_2^2(\{SM_1, BUF_1\})$ since the SFBC of the uncontrollable event 2 equals 0. Then, we can analyze the control functions with the transition function $\Delta$. However, in [17], $\{SM_1, BUF_1\}$ is deleted to ensure that the uncontrollable event will not occur. The calculation of the supremal subpredicate is given by keeping the reasonable states and deleting unreasonable ones. This will also narrow the system behavior. Such as for the supervisory control problem under partial observation, after the natural projection, the states are combined and thus one state might exists in different combination. Computing the supervisor only by deleting the states will reduce the permissive controlled behavior. In this paper, we can keep the states but delete the illegal events, which contributes to a more permissive supervisor.

6.3. Guide Way

A and B are two stations on a single track guideway. There are four sections between A and B, as shown in Figure 9, with the symbol “·” representing the guideway between two sections and “*” showing that the section junctions install spotlights. If there are two vehicles $V_1$ and $V_2$ running on this guideway, the use of the spotlights prevents collision by ensuring that $V_1$ and $V_2$ never travels on the same section simultaneously.

![Figure 9. Guideway](image)

Its STS model is shown in Figure 10, with the odd number standing for the controllable events and the even number for the uncontrollable events. In Figure 10, holon $V_1$ (respectively, $V_2$) stands for the behavior of vehicle $V_1$ (respectively, $V_2$). Take $V_1$ as an example. Since there are four sections between A and B, six stations are represented by six states (0 to 5) in holon $V_1$. Additionally, “·” implies the guideway between two sections, thus the five “·” can be represented as follows:

- event 11 in Figure 10 stands for the first “·”, which means $V_1$ passing the guideway between A and station 1;
- event 13 in Figure 10 stands for the second “·”, which means $V_1$ passing the guideway between station 1 and 2;
- event 10 in Figure 10 stands for the third “·”, which means $V_1$ passing the guideway between station 2 and 3;
- event 15 in Figure 10 stands for the forth “·”, which means $V_1$ passing the guideway between station 3 and 4;
- event 12 in Figure 10 stands for the fifth “·”, which means $V_1$ passing the guideway between station 4 and B.

Thus, to prevent collision, the specification can be represented by the set of exclusive state-trees $\{V_1, V_2\}$, $\{V_1, V_2\}$, $\{V_1, V_2\}$, $\{V_1, V_2\}$. This means that

$$f_{21}(\{V_1, V_2\}) = 0, f_{11}(\{V_1, V_2\}) = 0,$$

$$f_{23}(\{V_1, V_2\}) = 0, f_{13}(\{V_1, V_2\}) = 0,$$

$$f_{20}(\{V_1, V_2\}) = 0, f_{10}(\{V_1, V_2\}) = 0,$$

$$f_{25}(\{V_1, V_2\}) = 0, f_{15}(\{V_1, V_2\}) = 0.$$

To compute the supervisor, we can focus on $f_{20}(\{V_1, V_2\}) = 0$ and $f_{10}(\{V_1, V_2, V_2\}) = 0$ since 20 and 10 are not allowed to be disabled. Thus, according to Algorithm 4, the control function is shown as Figure 11. This means that 1) event 11 occurs when $V_1$ is at state 0 and $V_2$ is at state 0, 2, 3, 4, or 5; 2) event 13 occurs when $V_1$ is at state 1 and...
$V_2$ is at state 0, 4, or 5; 3) event 15 occurs when $V_1$ is at state 3 and $V_2$ is at state 0, 1, or 5; 4) event 21 is allowed to occur when $V_2$ is at state 0 and $V_1$ is at state 0, 2, 3, 4, or 5; 5) event 23 is allowed to occur when $V_2$ is at state 0 and $V_1$ is at state 0, 2, 3, 4, or 5; and 6) event 25 is allowed to occur when $V_2$ is at state 0 and $V_1$ is at state 0, 1, or 5.

![Guideway Diagram](image)

**Figure 10.** Guideway.

**Remark 8.** The computation of the control functions in this paper is more efficient than that of [17]. In this paper, we only need to focus on the SFBC of the uncontrollable events since if there exists an SFBC equal to 0, the control functions must be not controllable. For this example, $f_{20}(\{V_1=3, V_2=2\}) = 0$ and $f_{10}(\{V_1=2, V_2=3\}) = 0$. Since 20 and 10 are not allowed to be disabled, we need to make the SFBC of the controllable events (23 and 13) equal 0. Thus, just one step is needed to compute the control functions. However, in [17], the controllable predicate should be studied first by analyzing the uncontrollable events and then computing the control functions of the controllable events. The computation of the controllable predicate is omitted, which helps to improve the computational efficiency.
7. Discussion

In order to construct an efficient supervisor for complex automated manufacturing systems (AMS), supervisory control theory is put forward on the foundation of STS. STS can complete the control and modeling of hierarchy and concurrency of discrete-event systems (DES), and manage the state explosion encountered in the complex DES [14]. However, the relation between state-trees and events was not clearly defined during the course of investigating the supervisory control problem, which is of paramount significance in DES. In order to achieve this goal, an approach on the foundation of state feedback control (SFBC) is proposed in this paper, which constructs the control behavior directly by analyzing the SFBC of events. Thus, for a given AMS modeled by STS $G$, the requirements of users denoted by the specification can be modeled as follows:

1. The specification is explicitly saying that the dangerous situation (to be prohibited) is whenever the system is in a certain state. This specification can be denoted by the illegal state-trees. Such as, in Example 1, if the two machines, $R_{1,1}$ and $R_{2,1}$, are not allowed to occur at the same time. The specification can be written as $\{R_{1,1}, R_{2,1}\}$ with $B\neg P = \{\{R_{1,1}, R_{2,1}\}\}$, and the SFBC of events wrt. $P$ can be obtained according to Definition 2. Then, the supervisor (shown by the control functions) can be computed by Algorithm 4.

2. The specification can be put into the STS model as a holon. For example, in Small Factory, the specification is written as holon $BUF$, which implicitly describes the control part: not allowing 3 to occur at state 0. Moreover, to avoid the buffer overflow, the event 2 is not allowed to occur at state 1. Thus, the illegal state-tree is $\{SM_{1,1}, BUF_{1}\}$ and $B_P = \{\{SM_{1,0}, SM_{2,0}, BUF_{0}\}, \{SM_{1,1}, SM_{2,0}, BUF_{0}\}, \{SM_{1,0}, SM_{2,0}, BUF_{1}\}, \{SM_{1,0}, SM_{2,1}, BUF_{1}\}, \{SM_{1,0}, SM_{2,1}, BUF_{0}\}\}$. The control functions of the illegal state-trees are shown in Figure 11.

Figure 11. Control functions of 11, 13, 15, 21, 23, and 25.
The SFBC of events with respect to $P$ can be obtained according to Definition 2. Then, the supervisor (shown by the control functions) can be computed by Algorithm 4.

3. Moreover, the specification can be directly denoted by the SFBC which means some events disabled at state-trees. For example, in Small Factory, to avoid the buffer overflow, the event 2 is not allowed to occur at state 1. Thus, the SFBC of 2 can be written as $f^2_P(\{SM_1_1, BUF_1\}) = 0$. The SFBC of events with respect to $P$ can be achieved according to Definition 2. Then, the control functions can be computed by Algorithm 4.

For the given specification $P$, the supervisory control problems can be solved by the proposed propositions and algorithms in this paper. By Algorithm 1, the SFBC for $P$ under reachability can be computed and then, the SFBC under coreachability can be achieved by Algorithm 2. To solve the control problem, SFBC under controllability should be studied. Since the weak controllability implies controllability and the computation of the SFBC under weak controllability is more efficient, Algorithm 3 is proposed. As it is known that the controlled system should be nonblocking and controllable, Algorithm 4 is given to pursue the SFBC under coreachability and weak controllability since the nonblocking can be replaced by coreachability and controllability by weak controllability. As shown in examples, to compute the control functions, the SFBC under weak controllability and coreachability should be computed by using Algorithm 1 and Algorithm 2, iteratively. Then, the control functions can be obtained by using Algorithm 3. The reasonability of the algorithms can be proved by the nine propositions in this paper. During the computation process, only the SFBC of uncontrollable events needs to be considered, and the control functions can be obtained directly by analyzing the SFBC. This helps to improve the computational efficiency.

Moreover, this paper lays a foundation for the following work since the SFBC of events is very important to the development of other research. For example, the deployment of sensors and detectors to detect certain manufacturing processes may be too costly or difficult for technical reasons, which leads to the partial observability of AMS. To model and control these AMS, the definition of natural projection represented by predicates should be proposed. The SFBC of events with respect to predicates helps the analysis of the occurrence of events for the natural projection, which helps the achievement of more permissive system behavior under partial observation. Such as the example in Section 6.3, the symbol “!” represents detectors installed, implying that for $V_1(V_2)$, the events 11, 10, 15, and 12 (resp, 21, 20, 25, and 22) are observable and event 13 (respectively, 23) is unobservable. To compute the control functions, the natural projections should be performed first. Therefore, some states will be combined, such as $\{V_1_1, V_2_1\}$ and $\{V_1_2, V_2_1\}$ can be seen as a new state, represented by $P_1$, since 13 is unobservable as shown in [23]. The same is true for $\{V_1_1, V_2_1\}$ and $\{V_1_1, V_2_2\}$, represented by $P_2$. By using the proposed algorithms in this paper, $f^{P_1}_{13}(\{V_1_1, V_2_1\}) = 0$ does not imply that $f^{P_2}_{13}(\{V_1_1, V_2_1\}) = 0$, which is different from the previous work. The state-tree $\{V_1_1, V_2_1\}$ is not deleted in this situation and only 13 is disabled. Thus, more permissive supervisor can be obtained under partial observation.

The above works are based on the relation between predicates and events and the supervisor is given by control functions that are represented by states that controllable event can occur. All these works ignore the structure of the STS model, which leads to a situation that all these results cannot be used in the next computation. If we can study the supervisory control problem by analyzing the STS model, the computation can focus on the holons related to the specification. The expected supervisor can be also represented by STS, which can be used directly in other computation if necessary. This helps to improve the efficiency of the computation and the practicality of the theory.
8. Conclusions

Considering that all behaviors in an automated manufacturing system (AMS) are discrete in space and time, the system can be seen as a discrete event system (DES), which ensures a valid representation of the AMS and effectively performs subsequent computations. In this study, to solve the control problem of AMS modeled by state-tree structures (STS), the supervisor of AMS is studied on the basis of state feedback control (SFBC). For the control requirements modeled by the specification, the properties of the SFBC are well investigated. Then, with the purpose of studying the supervisory problem on the foundation of SFBC, the algorithms to compute the SFBC under reachability, coreachability, and weak controllability are introduced. It is proved that the nonblocking SFBC obtained by the proposed algorithm is equivalent to the one obtained by computing predicates first, which ensures satisfactory controllers for the supervisory control problems. Finally, with the intention of demonstrating the contribution of the proposed approach, which is designed considering the symmetry of the system behavior between the real system and its STS model, three examples are illustrated. Consequently, for any specification, which can be represented by SFBC, the solution to the supervisory control problems is well investigated. However, since all these results are represented by predicates which cannot be used in the next computation, the future work will be devoted to solving the problem in view of the structure of STS models.

Author Contributions: Conceptualization, C.G.; Data curation, J.Z.; Formal analysis, C.G.; Funding acquisition, C.G.; Methodology, C.G.; Project administration, Z.H.; Software, J.Z.; Supervision, C.G.; Validation, Z.H.; Visualization, Z.H.; Writing—original draft, C.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was sponsored in part by the National Natural Science Foundation of China Under Grant Nos. 62003201 and 61803246.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Li, Z.W.; Zhou, M.C. Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach; Springer Science Business Media: Berlin/Heidelberg, Germany, 2009.
2. Revelliotis, S.; Nazeem, A. Deadlock avoidance policies for automated manufacturing systems using finite state automata. Form. Methods Manuf. 2018, 169–196.
3. Fantini, M.P.; Zhou, M.C. Deadlock control methods in automated manufacturing systems. IEEE Trans. Syst. Man Cyber. 2004, 34, 5–22. [CrossRef]
4. Wu, N.Q.; Zhou, M.C. Deadlock resolution in automated manufacturing systems With robots. IEEE Trans. Autom. Sci. Eng. 2007, 4, 474–480. [CrossRef]
5. He, Z.; Ma, Z.; Li, Z.W.; Giua, A. Parametric transformation of timed weighted marked graphs: Applications in optimal resource allocation. IEEE/CAA J. Autom. Sinica 2020, 8, 179–188. [CrossRef]
6. He, Z.; Li, Z.W.; Giua, A. Performance optimization for timed weighted marked graphs under infinite server semantics. IEEE Trans. Autom. Control 2018, 63, 2573–2580. [CrossRef]
7. He, Z.; Ma, Z.; Tang, W. Performance safety enforcement in strongly connected timed event graphs. Automatica 2021, 128, 109605. [CrossRef]
8. He, Z.; Li, Z.W.; Giua, A. Cycle time optimization of deterministic timed weighted marked graphs by transformation. IEEE Trans. Control Syst. Technol. 2017, 25, 1318–1330. [CrossRef]
9. He, Z.; Li, Z.W.; Giua, A. Optimization of deterministic timed weighted marked graphs. IEEE Trans. Autom. Sci. Eng. 2017, 14, 1084–1095. [CrossRef]
10. He, Z.; Ma, Z.Y. Performance safety enforcement in stochastic event graphs against boost and slow attacks. Nonlinear Anal. Hybrid Syst. 2021, 41, 101057. [CrossRef]
11. Yan, L.Y. State-Based Control of Discrete-Event Systems with Observational Abstraction. Master’s Thesis, University of Toronto, Toronto, ON, Canada, 2011.
12. Wang, X.; Li, Z.W.; Wonham, W.M. Dynamic multiple-period reconfiguration of real-time scheduling based on timed DES supervisory control. *IEEE Trans. Ind. Inf.* 2015, 12, 101–111. [CrossRef]
13. Wang, X.; Khemaissia, I.; Khalgui, M.; Li, Z.W.; Wonham, W.M. Dynamic low-power reconfiguration of real-time systems with periodic and probabilistic tasks. *IEEE Trans. Autom. Sci. Eng.* 2014, 12, 258–271. [CrossRef]
14. Wonham, W.M. Supervisory Control of Discrete-Event Systems. 2015. Available online: http://www.control.toronto.edu/DES (accessed on 1 July 2015).
15. Wonham, W.M.; Ramadge, P.J. On the supremal controllable sublanguage of a given language. *SIAM J. Control Optim.* 1987, 25, 637–659. [CrossRef]
16. Wang, X.; Li, Z.W.; Wonham, W.M. Dynamic multiple-period reconfiguration of real-time scheduling based on timed DES supervisory control. *IEEE Trans. Ind. Inf.* 2015, 12, 101–111. [CrossRef]
17. Wang, X.; Khemaissia, I.; Khalgui, M.; Li, Z.W.; Wonham, W.M. Dynamic low-power reconfiguration of real-time systems with periodic and probabilistic tasks. *IEEE Trans. Autom. Sci. Eng.* 2014, 12, 258–271. [CrossRef]
18. Wonham, W.M.; Ramadge, P.J. On the supremal controllable sublanguage of a given language. *SIAM J. Control Optim.* 1987, 25, 637–659. [CrossRef]
19. Ma, C.; Wonham, W.M. Control of state tree structures. In Proceedings of the 11th Mediterranean Conference on Control and Automation, Rhodes, Greece, 18–20 June 2003.
20. Harel, D. Statecharts: A visual formalism for complex systems. *Sci. Comput. Program.* 1987, 8, 231–274. [CrossRef]
21. Ma, C.; Wonham, W.M. STSLib and its application to two benchmarks. In Proceedings of the 9th International Workshop on Discrete Event System (WODES’08), Gothenburg, Sweden, 28–30 May 2008; pp. 119–124.
22. Mirenadi, S.; Lennartson, B. Symbolic on-the-fly synthesis in supervisory control theory. *IEEE Trans. Control Syst. Technol.* 2016, 24, 1705–1716. [CrossRef]
23. Gu, C.; Wang, X.; Li, Z.W. Synthesis of supervisory control with partial observation on normal state-tree structures. *IEEE Trans. Autom. Sci. Eng.* 2018, 16, 984–997. [CrossRef]
24. Gu, C.; Wang, X.; Li, Z.W.; Wu, N.Q. Supervisory control of state-tree structures with partial observation. *Inf. Sci.* 2018, 465, 523–544. [CrossRef]
25. Wang, D.G.; Wang, X.; Li, Z.W. Nonblocking supervisory control of state-tree structures with conditional-preemption matrices. *IEEE Trans. Ind. Inf.* 2019, 16, 3744–3756. [CrossRef]
26. Jiao, T.; Gan, T.; Xiao, G.; Wonham, W.M. Exploiting symmetry of state tree structures for discrete-event systems with parallel components. *Int. J. Control* 2017, 90, 1639–1651. [CrossRef]
27. Wang, X.; Li, Z.W.; Wonham, W.M. Real-time scheduling based on nonblocking supervisory control of state-tree structures. *IEEE Trans. Autom. Control* 2020, 66, 4230–4237. [CrossRef]
28. Cai, K.; Wonham, W.M. Supervisor localization of discrete-event systems based on state tree structure. *IEEE Trans. Autom. Control* 2013, 59, 1329–1335. [CrossRef]
29. Dong, K.; Quan, Q.; Wonham, W.M. Failsafe mechanism design for autonomous aerial refueling using state tree structures. *Unmanned Syst.* 2019, 7, 261–279. [CrossRef]
30. Chao, W.J.; Gan, Y.M.; Wang, Z.A.; Wonham, W.M. Decentralized supervisory control of discrete-event systems based on state tree structures. In Proceedings of the 31st Chinese Control Conference, Hefei, China, 25–27 July 2012; pp. 2184–2190. [CrossRef]
31. Chao, W.J.; Gan, Y.M.; Wang, Z.A.; Wonham, W.M. On computation of coordinator in blocking decentralized supervisory control of state tree structures. In Proceedings of the 31st Chinese Control Conference, Hefei, China, 25–27 July 2012; pp. 2157–2162. [CrossRef]
32. Chao, W.J.; Gan, Y.M.; Wang, Z.A.; Wonham, W.M. Modular supervisory control and coordination of state tree structures. *Int. J. Control* 2013, 86, 9–21. [CrossRef]
33. Wang, X.; Moor, T.; Li, Z.W. Top-down nested supervisory control of state-tree structures based on state aggregations. *IFAC-Papers OnLine* 2020, 53, 11175–11180. [CrossRef]