Supervisory control of partially observed weighted discrete-event systems

Systems Engineering Group
Department of Mechanical Engineering
Eindhoven University of Technology
PO Box 513
5600 MB Eindhoven
The Netherlands
http://se.wtb.tue.nl/

SE Report: Nr. 2010-03

Supervisory Control of Partially Observed Weighted Discrete-Event Systems

Rong Su, Jan H. van Schuppen and Jacobus E. Rooda

ISSN: 1872-1567

SE Report: Nr. 2010-03
Eindhoven, February 2010
SE Reports are available via http://se.wtb.tue.nl/sereports

Abstract

When the Ramadge-Wonham supervisory control paradigm is applied to practical problems, it is desirable to require a closed-loop system be finitely coreachable in the sense that a marker state can be reached within a finite number of transitions regardless of the current state. Furthermore, considering that actions in a real system usually carry costs, it is desirable to synthesize a supervisor that incurs only a minimum cost. Pursuing finite coreachability with a minimum cost is the main motivation for developing a theory about optimal supervisory control of weighted discrete-event systems in the literature. In this paper we follow the same line of optimal supervisory control but with a new focus on partial observation, which is common in practical applications. We first define three finitely-weighted supervisory control problems, namely (1) to decide the existence of a finitely-weighted controllable and normal sublanguage; (2) to compute a finitely-weighted controllable and normal sublanguage, when the answer to Problem (1) is affirmative; (3) to compute the supremal minimum-weighted controllable and normal sublanguage, when the answer to Problem (1) is affirmative. Then we provide concrete algorithms to solve them.

1 Introduction

The Ramadge-Wonham supervisory control paradigm [2] [3] has been applied to many industrial applications. In this paradigm a supervisor is synthesized so that the closed-loop system satisfies some pre-specified requirements. There are three important notions: *controllability*, which ensures a supervisor to disable only controllable events; *observability*, which ensures a supervisor to respond only upon observations; and *nonblockingness*, which ensures a supervisor to be able to drive a system towards a marker state regardless of the current state. The last notion, nonblockingness, may potentially cause a problem for implementation because it does not guarantee *finite coreachability* which is defined as being able to reach a marker state within a finite number of transitions regardless of the current state. There are at least two approaches to solve the finite coreachability problem: (1) to add weights to transitions, and solve a finitely-weighted supervisory control problem; (2) to model a plant as an ω automaton, and synthesize a supervisor for infinite behaviors. The first approach follows the standard Ramadge-Wonham control paradigm with only a minor extension, namely to deal with weights. The idea is to compute a supervisor such that the sum weight of any sequence of transitions in the closed-loop system is finitely upper bounded. If there is no cycle in the system with zero sum weight, then the closed-loop system can always reach a marker state within a finite number of transitions. The weighted automata can not only help us achieve finite coreachability, but also allow us to model and solve some optimization problems such as control with minimum cost. This has been discussed in the work about optimal supervisory control, e.g., [8] [10]. The second approach is to synthesize a supervisor such that, by adopting an infinite-string acceptance condition, e.g. the Rabin or Muller acceptance condition, any run in the closed-loop system visits marker states infinitely often [14] [15], which implies finite coreachability in a finite-state automaton. The second approach has an arguable shortcoming: it uses Safra's determinization construction [18], which is rather difficult to implement [19] [20], even though its complexity is exponential time. For this reason, in this paper we adopt the setting of weighted finite-state automata with a goal of synthesizing a supervisor that enforces finite coreachability with a minimum cost under an assumption that partial observation may be present.

We model a plant as a weighted automaton, in which each transition is associated with a nonnegative real number. A requirement is modeled as an unweighted automaton. After defining the weight of a language as the largest of sum weights of strings in this language, we first present three control problems: (1) to decide whether there exists a finitely-weighted sublanguage of the plant's marked behavior (subject to the requirement), which is controllable and normal with respect to the plant; (2) to compute a finitely-weighted controllable and normal sublanguage, when the answer to Problem (1) is affirmative; (3) to compute the supremal minimum-weighted controllable and normal sublanguage, when the answer to Problem (1) is affirmative. Problems (1) and (2) are related to finite coreachability, where Problem (1) is about deciding the existence of a solution, and Problem (2) is to find one such solution. Problem (3) is about performance optimization. Then we provide concrete algorithms to solve these problems. The complexities of solving Problems (1) and (2) depend on the complexity of the relevant natural projection, which projects out unobservable events from the plant model. If the projection is a natural observer [12], whose complexity is shown to be polynomial time, then the complexities of solving Problems (1) and (2) are also polynomial time. The complexity of solving Problem (3) turns out to be NP-hard. Owing to the limited space, we have decided to put the NP-hardness result in a companion paper, which is still under preparation.

Several papers in the literature about optimal supervisory control of weighted discrete-

event systems have a similar setting as ours, e.g., [9] [8] [10] [11] [22]. In these papers weights are assigned to events or transitions, and the goal is to synthesize a supervisor, which can drive a target system from the initial state to a marker state with a minimum cost. Compared with them, the present paper is different in various aspects. The most significant difference is that partial observation is not considered in the these papers except [11]. Without partial observation, the aforementioned three problems can be solved by a single polynomial time algorithm described in [8], or in [10] when weights are also assigned to control actions. In fact, the algorithm of [8] is used in this paper to solve Problem (1) and (2), after we project out unobservable transitions and create a model with only observable transitions. But this strategy works under an assumption that the relevant natural projection is a natural observer [12]. Otherwise, we need to elaborate the projected plant model to make sure that, an optimal control strategy for the projected plant model will not cause blocking in the original plant model. This idea bears some similarity with the work described in [11]. For example, in [11] the authors use normality to handle partial observation by assuming that all unobservable events are uncontrollable (which makes observability and normality coincide). They use projection to remove unobservable transitions and create a so-called C-observer in order to obtain an optimal supervisor based it. To ensure that costs of unobservable transitions can be properly recorded in the projected model, they introduce the notion of *locally computed cost*, which has its counterpart in the present paper. To ensure that a supervisor based on the projected model will not cause blocking in the original model, they introduce the concept of *admissible control actions*, which serves the same purpose as the notion of *auxiliary weight function* in the present paper. Although two papers share many similar features, their difference is also significant. In [11] the authors assume that the marker state in the plant model has an indicator event, which is treated as controllable and observable. Without this assumption their approach based on projection may not work. In the present paper we do not make such an assumption. The authors in [11] pursue the DP-optimality with respect to the projected plant model. In the present paper we aim at the supremal controllable and normal sublanguage with respect to the original plant model. As a consequence, the algorithms used in the present paper are completely different from the one used in [11]. Besides partial observation, the models and/or cost functions in the aforementioned papers are not exactly the same as the one used in this paper. For example, in [9] all events are assumed to be controllable. In [10] and [11] weights are assigned to events instead of directly on transitions, and are interpreted as occurrence costs and control costs separately. Their notion of DP-optimality is different from our notion of optimality in the sense that it consists of many local cost functions that need to be minimized altogether, one for each local state whose value denotes the cost of reaching a marker state from the current state. In [22] weights are assigned to transitions, and the authors define their cost function by taking the maximum over some local cost functions - one for each prefix substring whose value denotes the cost of extending the current string to a marked one. Such differences on models and/or cost functions make their algorithms deviate from ours further.

The problems discussed in this paper are also related to supervisory control of timed automata, where, in a restricted form, each transition is associated with a time interval applicable to a specific clock. The control goal is to synthesize a supervisor such that the closed-loop system can reach a certain desirable state (e.g. a marker state) with a duration no more than some pre-specified value, even when uncontrollable events happen. A timed automaton can be treated as a special weighted automaton. If we use only one clock, and restrict every time interval to a singleton, then a timed automaton becomes an ordinary weighted automaton with weights on transitions. Settings close to ours can be found, e.g., in [5] [6] [7]. Compared with these papers, ours is different in the following aspects. First, we deal with partial observation and they do not. Second, we use standard Ramadge-Wonham control paradigm to deal with synthesis, and they use a game theo-

retic approach, where uncontrollable events play the role of the opponent, which tries to maximize the cost, and the controllable events correspond to the player who tries to minimize the cost. As a result, a solution to their control problems corresponds to a strategy of the player in response to the opponent's moves. Finally, all these papers except [7] aim to find one strategy, which need not be the least restrictive. This synthesis objective is close to ours in solving Problems (1) and (2), and part of (3) without addressing the supremality. In [7] the least restrictive control strategy is in terms of game strategies, which is close to state-based feedback control. As a contrast, our least restrictive control strategy is in terms of supremal minimum-weighted controllable and normal sublanguages.

This paper is organized as follows. In Section II we first provide all relevant concepts about languages and automata, then introduce three supervisory control problems. After presenting algorithms to solve those problems in Section III, conclusions are drawn in Section IV.

2 Finitely-weighted Supervisory Control Problems

In this section we first review basic concepts of languages and weighted finite-state automata. Then we present three finitely-weighted supervisory control problems under partial observation.

Let Σ be a finite alphabet, and Σ^* denote the Kleene closure of Σ, i.e. the collection of all finite sequences of events taken from Σ. Given two strings $s, t \in \Sigma^*$, s is called a *prefix substring* of t, written as $s \leq t$, if there exists $s' \in \Sigma^*$ such that $ss' = t$, where ss' denotes the concatenation of s and s'. We use ϵ to denote the empty string of Σ^* such that for any string $s \in \Sigma^*$, $\epsilon s = s\epsilon = s$. A subset $L \subseteq \Sigma^*$ is called a *language*. $\overline{L} = \{s \in \Sigma^* | (\exists t \in L)\, s \leq t\} \subseteq \Sigma^*$ is the *prefix closure* of L. We say L is *prefix closed* if $L = \overline{L}$. Given two languages $L, L' \subseteq \Sigma^*$, let $LL' := \{ss' \in \Sigma^* | s \in L \wedge s' \in L'\}$ denote the concatenation of two sets. When L is a singleton, say $L = \{s\}$, then we simply use sL' to denote $\{s\}L'$. Let $s/L := \{s' \in \Sigma^* | ss' \in L\}$. Let $\Sigma' \subseteq \Sigma$. A mapping $P : \Sigma^* \to \Sigma'^*$ is called the *natural projection* with respect to (Σ, Σ'), if

1. $P(\epsilon) = \epsilon$
2. $(\forall \sigma \in \Sigma)\, P(\sigma) := \begin{cases} \sigma & \text{if } \sigma \in \Sigma' \\ \epsilon & \text{otherwise} \end{cases}$
3. $(\forall s\sigma \in \Sigma^*)\, P(s\sigma) = P(s)P(\sigma)$

Given a language $L \subseteq \Sigma^*$, $P(L) := \{P(s) \in \Sigma'^* | s \in L\}$. The inverse image mapping of P is

$$P^{-1} : 2^{\Sigma'^*} \to 2^{\Sigma^*} : L \mapsto P^{-1}(L) := \{s \in \Sigma^* | P(s) \in L\}$$

Let $P_o : \Sigma^* \to \Sigma_o^*$ be the natural projection. Given $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$, the *synchronous product* of L_1 and L_2 is defined as:

$$L_1 || L_2 := P_1^{-1}(L_1) \cap P_2^{-1}(L_2)$$

where $P_1 : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_1^*$ and $P_2 : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_2^*$ are natural projections.

A *weighted finite-state automaton* is a pair $(G = (X, \Sigma, \xi, x_0, X_m), f)$, where G denotes a deterministic finite-state automaton with X for the state set, Σ for the alphabet, $\xi : X \times \Sigma \to X$ for the (partial) transition function, x_0 for the initial state, and X_m for the marker state set, and $f : X \times \Sigma \to \mathbb{R}^+ \cup \{+\infty\}$ is the (partial) weight function, where $\mathbb{R}^+$ denotes the set of nonnegative reals. Here we treat $+\infty$ as a single value such that: (1) $(+\infty) + (+\infty) = +\infty$; (2) for all $a \in \mathbb{R}^+$, $+\infty + a = +\infty$. We use $\xi(x, \sigma)!$ to denote that, the transition $\xi(x, \sigma)$ is defined, and $\neg\, \xi(x, \sigma)!$ for $\xi(x, \sigma)$ not being defined. As usual, we extend the domain of ξ from $X \times \Sigma$ to $X \times \Sigma^*$. Define $\mu_G(x) := \{\sigma \in \Sigma | \xi(x, \sigma)!\}$, which is the collection of all events that are defined at state x in G. Let $L(G) := \{s \in \Sigma^* | \xi(x_0, s)!\}$ be the *closed* behavior of G and $L_m(G) := \{s \in L(G) | \xi(x_0, s) \in X_m\}$ for the *marked* behavior of G. We call G *nonblocking* if $\overline{L_m(G)} = L(G)$. Let $\Phi(\Sigma)$ be the collection of all weighted finite-state automata, whose alphabet is Σ, and $\phi(\Sigma)$ for the collection of all unweighted finite-state automata, whose alphabet is Σ. We say an automaton $G \in \phi(\Sigma)$ *recognizes* a language $K \subseteq \Sigma^*$, if $L_m(G) = K$ and $\overline{L_m(G)} = L(G)$.

To associate each sublanguage with a weight, we first define a weight for each string. To this end, let $\theta_{G,f} : X \times \Sigma^* \to \mathbb{R}^+$ be a map, where

1. $\theta_{G,f}(x, \epsilon) = 0$
2. $(\forall s\sigma \in \Sigma^*)\, \theta_{G,f}(x, s\sigma) := \begin{cases} \theta_{G,f}(x, s) + f(\xi(x, s), \sigma) & \text{if } \xi(x, s\sigma)! \\ +\infty & \text{otherwise} \end{cases}$

In other words, the weight of a string is simply the sum of weights of transitions appearing in this string. For each $K \subseteq L(G)$, the *weight* of K with respect to G is defined as follows:

$$\omega_{G,f}(K) := \begin{cases} \sup_{s \in K} \theta_{G,f}(x_0, s) & \text{if } K \neq \varnothing \\ +\infty & \text{otherwise} \end{cases}$$

The reason why we define the weight of K as the supremum of string weights in K is that, we intend to use the "worst-case" execution cost in terms of the largest weight of all execution paths (i.e., strings) to measure the quality of a controlled behavior described by a sublanguage. The motivation of assigning an infinite weight to the empty set is that, there is no string in an empty set that can bring the system to a marker state, thus, no finite weights can be incurred.

In this paper we will frequently use an automaton $G' \in \phi(\Sigma)$ that recognizes a sublanguage of $L_m(G)$. To associate a proper weight function with G', we introduce the following concept.

Definition 2.1. Given $G_i = (X_i, \Sigma, \xi_i, x_{i,0}, X_{i,m}) \in \phi(\Sigma)$ for $i = 1, 2$, let $g : X_1 \to X_2$ be a map. We say G_1 is *homomorphic* to G_2 with respect to g, if

1. $(\forall x \in X_{1,m})\, g(x) \in X_{2,m}$
2. $(\forall x, x' \in X_1)(\forall \sigma \in \Sigma)\, x' \in \xi_1(x, \sigma) \Rightarrow g(x') \in \xi_2(g(x), \sigma)$ □

Suppose $G' = (X', \Sigma, \xi', x'_0, X'_m) \in \phi(\Sigma)$ is homomorphic to $G = (X, \Sigma, \xi, x_0, X_m)$ with respect to a map g. Then we say a weight function $f' : X' \times \Sigma \to \mathbb{R}^+ \cup \{+\infty\}$ is *induced*

from $(G, f) \in \Phi(\Sigma)$, if

$$(\forall x \in X')(\forall \sigma \in \Sigma)\, f'(x, \sigma) := f(g(x), \sigma)$$

Given a sublanguage $K \subseteq L_m(G)$, which is regular, we can always find a finite-state automaton $G' \in \Phi(\Sigma)$, which recognizes K and is homomorphic to G. Let f' be induced from (G, f). For each $K' \subseteq K = L_m(G') \subseteq L_m(G)$, it is straightforward to show that $\omega_{G',f'}(K') = \omega_{G,f}(K')$.

To present our supervisory control problem, we need the concepts of controllability and normality. Let $\Sigma = \Sigma_c \cup \Sigma_{uc} = \Sigma_o \cup \Sigma_{uo}$, where disjoint subsets Σ_c and Σ_{uc} denote respectively the set of *controllable* events and the set of *uncontrollable* events, and disjoint subsets Σ_o and Σ_{uo} denote respectively the set of *observable* events and the set of *unobservable* events. We have the following two definitions.

Definition 2.2. [13] Given $G \in \phi(\Sigma)$, a language $K \subseteq L(G)$ is *controllable* with respect to G, if $\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K}$. □

Definition 2.3. Given $G \in \phi(\Sigma)$, a language $K \subseteq L(G)$ is *normal* with respect to G and the natural projection $P_o : \Sigma^* \to \Sigma_o^*$, if $\overline{K} = L(G) \cap P_o^{-1}(P_o(\overline{K}))$. □

Def. 2.3 is different from the definition of normality defined in [4], where normality is a property on the language itself, not on its prefix closure as used in Def. 2.3. We hope this slight abuse of notation will not cause any confusion for readers. When P_o is known, we simply say that K is normal with respect to G. Given another automaton $E \in \phi(\Sigma)$, which is treated as a requirement, let

$$\mathcal{C}(G, E) := \{K \subseteq L_m(G) \cap L_m(E) | K \text{ is controllable with respect to } G\}$$

and

$$\mathcal{CN}(G, E) := \{K \in \mathcal{C}(G, E) | K \text{ is normal with respect to } G\}$$

Since controllability and normality is closed under set union, we use $\sup\mathcal{C}(G, E)$ and $\sup\mathcal{CN}(G, E)$ to denote respectively the supremal controllable sublanguage and the supremal controllable and normal sublanguage of G with respect to E. Let

$$\mathcal{WCN}(G, f, E) := \{K \in \mathcal{CN}(G, E) | \omega_{G,f}(K) < +\infty\}$$

be the collection of controllable and normal sublanguages of G with respect to E, whose weights are finite. It is possible that, $\mathcal{WCN}(G, f, E) = \varnothing$ while $\mathcal{CN}(G, E) \neq \varnothing$. For example, consider a weighted automaton $(G, f) \in \Phi(\Sigma)$ with only uncontrollable transitions and some of them form a loop. If E allows all transitions in (G, f), e.g., $L(E) = L_m(E) = \Sigma^*$, then clearly $\mathcal{CN}(G, E) = \{L_m(G)\} \neq \varnothing$. But $\mathcal{WCN}(G, f, E) = \varnothing$ because the loop cannot be eliminated by disabling controllable events. Because

$$\min_{(x,\sigma) \in X \times \Sigma : f(x,\sigma) > 0} f(x, \sigma) > 0,$$

for all $K \in \mathcal{WCN}(G, f, E)$ we can derive that

$$\{\omega_{G,f}(K') | K' \in \mathcal{WCN}(G, f, E) \wedge \omega_{G,f}(K') \leq \omega_{G,f}(K)\}$$

is finite. Thus, there exists $K^* \in \mathcal{WCN}(G, f, E)$ such that

$$(\forall K \in \mathcal{WCN}(G, f, E))\, \omega_{G,f}(K^*) \leq \omega_{G,f}(K)$$

We call K^* a *minimum weighted controllable and normal sublanguage of* (G, f) *with respect to* E. Let $\Xi(G, f, E) \subseteq \mathcal{WCN}(G, f, E)$ be the collection of all minimum weighted controllable and normal sublanguages of (G, f) with respect to E. Since an arbitrary

union of controllable and normal sublanguages is still controllable and normal, we have that

$$\cup_{K\in\Xi(G,f,E)}K \in \Xi(G,f,E),$$

which is called the *supremal minimum weighted controllable and normal sublanguage of* (G,f) *with respect to* E, and denote it as $\sup\Xi(G,f,E)$. We make the following assumption:

Assumption 1: Throughout this paper each plant model $(G=(X,\Sigma,\xi,x_0,X_m),f)\in\Phi(\Sigma)$ is *marking deadlock* in the sense that, for all $x\in X_m$, $\mu_G(x)=\varnothing$; and *zero-weighted-loop-free* in the sense that, for all $s_1s_2^*\subseteq L(G)$ with $s_1,s_2\in\Sigma^*$, we have $\theta_{G,f}(\xi(x_0,s_1),s_2)>0$. □

In other words, every marker state of G is a deadlock state and there is no loop in G, whose weight is zero. The reason why we are interested in zero-weighted-loop-free automata is that we want to design a supervisor such that finite coreachability holds in the closed-loop system. If there is a loop in (G,f) with zero weight, then it is possible that the weight of a controllable and normal sublanguage is finite, but the closed-loop system is not finitely coreachable. The reason why we impose marking deadlock on (G,f) is that we are interested in a supervisor that can drive the closed-loop system to a marker state from the initial state with a finite (and ideally, the minimum) cost. As long as a marker state is reached, whether the plant continues or stops is not our concern. In reality, after a marker state is reached, the system can be reset to repeat the same sequence. From now on we assume that $\Phi(\Sigma)$ is the collection of all weighted finite-state automata, which are marking deadlock and zero-weighted-loop-free. We now state our problems:

Problem 2.4. Given $(G,f)\in\Phi(\Sigma)$ and $E\in\phi(\Sigma)$, decide whether $\mathcal{WCN}(G,f,E)\neq\varnothing$. □

Problem 2.5. When $\mathcal{WCN}(G,f,E)\neq\varnothing$, compute one $K\in\mathcal{WCN}(G,f,E)$. □

Problem 2.6. When $\mathcal{WCN}(G,f,E)\neq\varnothing$, compute $\sup\Xi(G,f,E)$. □

For notation simplicity we use WSCP to denote the above computational problems, which stands for the *Weighted Supervisory Control Problem*. To distinguish individual problems, we use WSCP1, WSCP2 and WSCP3 respective. WSCP1 is about deciding whether there exists a finitely weighted controllable and normal sublanguage of (G,f) with respect to E. WSCP2 aims to find one such a sublanguage. WSCP3 aims to find the supremal minimum-weighted controllable and normal sublanguage of (G,f) with respect to E. Next, we will provide concrete algorithms to show that WSCP is solvable.

3 Algorithms for Solving WSCP

In this section we first present an algorithm to solve WSCP in its general setting, namely with a plant modeled as a deterministic weighted finite-state automaton and a requirement modeled as a deterministic unweighted finite-state automaton. Then we show that, WSCP can be solved more efficiently if the natural projection P_o possesses a certain property.

3.1 An algorithm for solving WSCP in the general setting

Given $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$, we construct another finite-state automaton $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m) \in \phi(\Sigma_o)$, where

1. $Z \subseteq 2^X$, $Z_m := \{z \in Z | z \cap X_m \neq \varnothing\}$, $z_0 := \{x \in X | (\exists u \in \Sigma_{uo}^*)\, \xi(x_0, u) = x\}$
2. $(\forall z \in Z)(\forall \sigma \in \Sigma_o)\, \delta(z, \sigma) := \{x \in X | (\exists x' \in z)(\exists s \in P_o^{-1}(\sigma))\, \xi(x', s) = x\}$

The definition of $\hat{G}$ is simply a power-set construction [13], and $\hat{G}$ recognizes $P_o(L_m(G))$, i.e., $L_m(\hat{G}) = P_o(L_m(G))$ and $L(\hat{G}) = P_o(L(G))$. We call $\hat{G}$ the *observable behavior* of (G, f). In general, G being nonblocking does not necessarily implies $\hat{G}$ being nonblocking, because the natural projection may mask out some blocking behaviors of G. We define a map $\hat{h} : Z \times 2^{\Sigma_o} \to \{+\infty, 0\}$, where for each $(z, \gamma) \in Z \times 2^{\Sigma_o}$,

$$\hat{h}(z, \gamma) := \begin{cases} 0 & \text{if } (\forall x \in z)\, [(\exists u \in \Sigma_{uo}^*)\, \xi(x, u) \in X_m \ \vee\ (\exists \sigma \in \gamma)(\exists s \in P_o^{-1}(\sigma))\, \xi(x, s)!] \\ +\infty & \text{otherwise} \end{cases}$$

We call $\hat{h}$ the *auxiliary weight function* of $(\hat{G}, \hat{f})$ with respect to (G, f). If we interpret γ as the collection of events that are allowed to be fired at z, then $\hat{h}(z, \gamma) = 0$ if and only if under the transitions of γ every state $x \in z$ can either reach a marker state of G via a 'silent' path or reach another state via some path whose projected image is in γ. For each $z, z' \in Z$, let $\rho[z, z'] : z \times z' \times \Sigma_o \to \mathbb{R}^+$ be a (partial) map, where, for all $x \in z$, $x' \in z'$ and $\sigma \in \Sigma_o$,

$$\rho[z, z'](x, x', \sigma) := \begin{cases} \sup_{s \in P_o^{-1}(\sigma) \wedge \xi(x,s) = x'} \theta_{G,f}(x, s) & \text{if } (\exists s \in P_o^{-1}(\sigma))\, \xi(x, s) = x' \\ \textit{not defined} & \text{otherwise} \end{cases}$$

The value of $\rho[z, z'](x, x', \sigma)$ is the largest weight of all pathes that connect x and x', whose project images are σ. It can be computed by using an ϵ-removal algorithm [21], which creates between x and x' multiple transitions labeled by σ but with different weights. The maximum weight of these transitions is equal to $\rho[z, z'](x, x', \sigma)$. We now use a simple example to illustrate these two functions.

Suppose we have a weighted automaton $(G, f) \in \Phi(\Sigma)$ depicted in Figure 1, where $\Sigma = \{a, b, c, u_1, u_2, u_3, u_4\}$ and $\Sigma_o = \{a, b, c\}$. The label $a/1$ on the transition from state 0 to state 1 means that the event is a and the corresponding weight is $f(0, a) = 1$. Other transition labels have similar meanings. The observable behavior $\hat{G}$ is depicted in Figure 1, where $z_0 = \{0, 2\}$, $z_1 = \{1, 3, 5\}$, $z_2 = \{3, 5\}$ and $z_4 = \{3, 4, 5, 6\}$. For state z_0, if we choose $\gamma = \{a\}$, then $\hat{h}(z_0, \{a\}) = 0$ because both state 0 and state 2 can fire the transition a in G. But if we choose $\gamma = \{b, c\}$ then $\hat{h}(z_0, \{b, c\}) = +\infty$. Similarly, for state z_1, if we choose $\gamma = \{b\}$, then $\hat{h}(z_1, \{b\}) = +\infty$ because state 1 is a blocking state and there is no string from state 1 to another state, whose projected image is b. We can easily check that, $\hat{h}(z_2, \{b\}) = \hat{h}(z_3, \{b\}) = 0$. Since there are infinite number of finite strings from state 0 to state 1 whose projected images are a, we can derive that $\rho[z_0, z_1](0, 1, a) = +\infty$. Since there is only one string $s = u_3 b$ from state 3 to state 6, whose project is b, we have $\rho[z_1, z_3](3, 6, b) = \theta_{G,f}(3, u_3 b) = 3$. Other values of the function ρ can be derived accordingly.

We now present an algorithm, which returns a finite value v when $\mathcal{WCN}(G, f, G) \neq \varnothing$ (or equivalently, $\Xi(G, f, G) \neq \varnothing$); or returns $v = +\infty$ when $\mathcal{WCN}(G, f, G) = \varnothing$. When

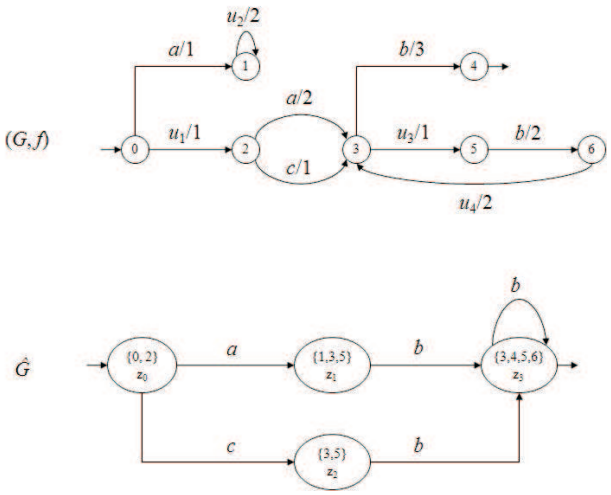

Figure 1: Example 1: Weighted automaton $(G, f) \in \Phi(\Sigma)$ and observable behavior $\hat{G} \in \phi(\Sigma_o)$

$v < +\infty$, it returns a $K \in \mathcal{WCN}(G, f, G)$. This algorithm will be used to solve WSCP.

Pivotal Procedure 1 (PP1):

1. Input: $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$
2. Initialization:
   (a) Let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m) \in \phi(\Sigma_o)$ be the observation behavior of (G, f).
   (b) For all $z \in Z$ let $\Gamma_z := \{\gamma \subseteq \mu_{\hat{G}}(z) | \Sigma_{uc} \cap \Sigma_o \cap \mu_{\hat{G}}(z) \subseteq \gamma\}$.
   (c) For each $z \in Z$ and $x \in z$, if $x \in X_m$ then $\kappa_0(z, x) := 0$; if $x \notin X_m$, then $\kappa_0(z, x) := +\infty$; $\kappa_0(z) := \max_{x \in z} \kappa_0(z, x)$.
3. Iterate on $k = 1, 2, \cdots$, as follows:
   (a) For each $z \in Z$ and $x \in z$ we have

$$\kappa_k(z) := \begin{cases} \min_{\gamma \subseteq \Gamma_z} W_k(z, \gamma) + \hat{h}(z, \gamma) & \text{if } \kappa_{k-1}(z) = +\infty \\ \kappa_{k-1}(z) & \text{otherwise} \end{cases}$$

   where

$$W_k(z, \gamma) := \max_{x \in z} \max(v(z, \gamma, x) \cup \varphi(z, x))$$

   with

$$\begin{aligned} v(z, \gamma, x) &:= \{\rho[z, z'](x, x', \sigma) + \kappa_{k-1}(z', x') | (\exists \sigma \in \gamma, x' \in z' = \delta(z, \sigma))\, \rho[z, z'](x, x', \sigma)!\} \\ \varphi(z, x) &:= \{\theta_{G,f}(x, u) | u \in \Sigma_{uo}^* \wedge \xi(x, u) \in X_m\} \end{aligned}$$

   When $\kappa_{k-1}(z) = +\infty$ and $\kappa_k(z) < +\infty$, let $\gamma_z \in \Gamma_z$ be the one such that $W_k(z, \gamma_z) + \hat{h}(z, \gamma_z) = \kappa_k(z)$. For each $x \in z$, let

$$\kappa_k(z, x) := \begin{cases} \max(v(z, \gamma_z, x) \cup \varphi(z, x)) & \text{if } \kappa_{k-1}(z) = +\infty \text{ and } \kappa_k(z) < +\infty \\ \kappa_{k-1}(z, x) & \text{otherwise} \end{cases}$$

   (b) Termination when: $(\exists r \in \mathbb{N})(\forall z \in Z)(\forall x \in z)\, \kappa_{r-1}(z, x) = \kappa_r(z, x)$

4. Outputs:
   - $v := \kappa_r(z_0, x_0)$
   - When $v < \infty$, output $K_* := L_m(\hat{G}')||L_m(G)$ with $\hat{G}' := (Z', \Sigma_o, \delta', z_0', Z_m')$, where
     - $Z' := \{z \in Z | \kappa_r(z) < +\infty\}$, $Z_m' := Z_m \cap Z'$ and $z_o' := z_0$.
     - $\delta' : Z' \times \Sigma_o \to Z'$, where for all $z, z' \in Z'$ and $\sigma \in \Sigma_o$, $\delta'(z, \sigma) = z'$ if $\sigma \in \gamma_z$. □

The complexity of constructing $\hat{G}$ is exponential time [16]. The main feature of Step (3) is that, once $\kappa_k(z, x)$ becomes finite, it will not be changed in the subsequent iterations. This finite value denotes one finite "worst-case" weight of all strings from state x in z to a marker state in G with no more than k transitions. Here, "worst-case" means that the evolution of a string may be diverged by occurrences of uncontrollable events. By an elaborated argument (which is not shown in this paper owing to limited space), we can check that Step (3) terminates with k no more than the length of the longest string in G, i.e., no more than the total number of states in G. Thus, the complexity of iterations in Step (3) is polynomial with respect to the size of $\hat{G}$. Finally, constructing $\hat{G}'$ can be done in linear time with respect to the size of $\hat{G}$, and K_* can be computed in polynomial time. Thus, we can conclude that the complexity of PP1 is exponential time with respect to the size of G. We have the following results.

Proposition 3.1. PP1 always terminates. □

Proof: We can check that, for each $z \in Z$, $x \in z$ and $k \in \mathbb{N}$, $\kappa_k(z, x)$ is either $+\infty$ or finite. When $\kappa_k(z, x)$ is finite, namely $\kappa_k(z) < +\infty$, we can derive that, for all $l \geq k$ we have $\kappa_l(z, x) = \kappa_k(z, x)$. Before the termination condition holds, at each k there exist at least one $z \in Z$ and $x \in z$ such that $\kappa_{k-1}(z, x) \neq \kappa_k(z, x)$. The total number of changes are no more than $|Z| \times |X|$. Therefore, there must exist $r \leq |Z| \times |X|$ such that

$$(\forall z \in Z)(\forall x \in z)\, \kappa_{r-1}(z, x) = \kappa_r(z, x)$$

This means PP1 terminates within no more than $|Z| \times |X|$ times iteration. ■

The termination of PP1 comes from the fact that, we have only a finite number of states to consider, which are no more than $|Z| \times |X|$, and the value of each one of these states can be changed no more than once during the iteration.

Proposition 3.2. Given $(G, f) \in \Phi(\Sigma)$, suppose the output v of PP1 is finite. Then

1. $K_* \in \mathcal{CN}(G, G)$;
2. $\omega_{G,f}(K_*) \leq v$. □

Proof: (1) For each $s \in \overline{K_*}$ and $\sigma \in \Sigma_{uc}$, suppose $s\sigma \in L(G)$. Let $z = \delta(z_0, P_o(s))$ and $x = \xi(x_0, s)$ and $x' = \xi(x_0, s\sigma)$. There are two cases to consider.
Case 1: $\sigma \in \Sigma_{uo}$. Then by the definition of $\hat{G}$ we know that $x, x' \in z$. Since $s \in \overline{K_*}$, we know that $z \in Z'$, which means $\kappa_r(z) < +\infty$. Thus, $\kappa_r(z, x') < +\infty$. By PP1 and the definition of $\hat{G}$ we can derive that, there must exist $s' \in \Sigma^*$ such that $s\sigma s' \in K_*$, which

means $s\sigma \in \overline{K_*}$.
Case 2: $\sigma \in \Sigma_o$. Let $z' = \delta(z, \sigma)$. Since $\sigma \in \Sigma_{uc}$ we can derive that $\sigma \in \gamma_z$. Since $\kappa_r(z) < +\infty$ we get that $\kappa_r(z') < +\infty$. Thus, again we can derive that there must exist $s' \in \Sigma^*$ such that $s\sigma s' \in K_*$, which means $s\sigma \in \overline{K_*}$.
In either case $s\sigma \in \overline{K_*}$. Thus, K_* is controllable with respect to G.
To show that K_* is normal with respect to G and P_o, let $s \in \overline{L_m(K_*}$ and $s' \in L(G)$ with $P_o(s) = P_o(s')$. Let $s = u_0\sigma_0u_1\sigma_1 \cdots u_n\sigma_nu_{n+1}$ with $u_0, \cdots, u_{n+1} \in \Sigma_{uo}^*$ and $\sigma_0, \cdots, \sigma_n \in \Sigma_o$. Since $P_o(s) = P_o(s')$ we get that $s' = u_0'\sigma_0u_1'\sigma_1 \cdots u_n'\sigma_nu_{n+1}'$ with $u_0', \cdots, u_{n+1}' \in \Sigma_{uo}^*$. Clearly, for any $\check{s} \leq s$ and $\check{s}' \leq s'$, if $P_o(\check{s}) = P_o(\check{s}')$, then $\xi(x_0, \check{s}), \xi(x_0, \check{s}') \in \delta(z_0, P_o(\check{s}))$. Let $x' = \xi(x_0, s')$ and $z = \delta(z_0, P_o(s))$. Since $s \in \overline{K_*}$, we have $\kappa_r(z) < +\infty$, which means $\kappa_r(z, x') < +\infty$. Thus, there exists $s'' \in \Sigma^*$ such that $s's'' \in K_*$, which means $s' \in \overline{K_*}$.
(2) To show that $\omega_{G,f}(K_*) \leq v$, let $s \in K_*$. Then either $s \in \Sigma_{uo}^*$ or there exist $u_0, \cdots, u_n \in \Sigma_{uo}^*$ with $n \geq 1$ and $\sigma_1, \cdots, \sigma_n \in \Sigma_o$ such that $s = u_0\sigma_1u_1\sigma_2 \cdots u_{n-1}\sigma_nu_n$. Suppose PP1 terminates at r. For the former case, i.e., $s \in \Sigma_{uo}^*$, we get that

$$\kappa_r(z_0, x_0) = v \geq \phi(z_0, x_0) \geq \theta_{G,f}(x_0, s)$$

For the latter case, let $x_1 = \xi(x_0, u_0\sigma_1u_1)$ and $x_i := \xi(x_{i-1}, \sigma_iu_i)$ for $i = 2, \cdots, n$. Let $z_i := \delta(z_{i-1}, \sigma_i)$ for $i = 1, 2 \cdots, n$. Clearly, $x_i \in z_i$ for $i = 0, 1, 2, \cdots, n$. Since $s \in K_* \subseteq L_m(G)$, we know that $x_n \in X_m$. We now use induction to show that

$$(\forall i \in \mathbb{N} : 1 \leq i \leq n-1)\, \kappa_r(z_i, x_i) \geq \theta_{G,f}(x_i, \sigma_{i+1}u_{i+1} \cdots u_{n-1}\sigma_nu_n) \tag{1}$$

For $i = n-1$, we have

$$\begin{aligned}
\kappa_r(z_{n-1}, x_{n-1}) &= \max(\upsilon(z_{n-1}, \gamma_{z_{n-1}}, x_{n-1}) \cup \varphi(z_{n-1}, x_{n-1})) \\
&\geq \max \upsilon(z_{n-1}, \gamma_{z_{n-1}}, x_{n-1}) \\
&\geq \rho[z_{n-1}, z_n](x_{n-1}, x_n, \sigma_n) \text{ because } \kappa_{r-1}(z_n, x_n) = 0 \\
&\geq \theta_{G,f}(x_{n-1}, \sigma_nu_n)
\end{aligned}$$

Suppose when $2 \leq i = l \leq n-1$ Expression (1) holds. Then we have

$$\begin{aligned}
\kappa_r(z_{l-1}, x_{l-1}) &= \max(\upsilon(z_{l-1}, \gamma_{z_{l-1}}, x_{l-1}) \cup \varphi(z_{l-1}, x_{l-1})) \\
&\geq \max \upsilon(z_{l-1}, \gamma_{z_{l-1}}, x_{l-1}) \\
&\geq \rho[z_{l-1}, z_l](x_{l-1}, x_l, \sigma_l) + \kappa_{r-1}(z_l, x_l) \\
&\geq \theta_{G,f}(x_{l-1}, \sigma_lu_l) + \theta_{G,f}(x_l, \sigma_{l+1}u_{l+1} \cdots u_{n-1}\sigma_nu_n) \\
&= \theta_{G,f}(x_{l-1}, \sigma_lu_l\sigma_{l+1}u_{l+1} \cdots u_{n-1}\sigma_nu_n)
\end{aligned}$$

Thus, Expression (1) holds. Clearly, we have

$$\begin{aligned}
\kappa_r(z_0, x_0) &= \max(\upsilon(z_0, \gamma_{z_0}, x_0) \cup \varphi(z_0, x_0)) \\
&\geq \max \upsilon(z_0, \gamma_{z_0}, x_0) \\
&\geq \rho[z_0, z_1](x_0, x_1, \sigma_1) + \kappa_{r-1}(z_1, x_1) \\
&\geq \theta_{G,f}(x_0, u_0\sigma_1u_1) + \theta_{G,f}(x_1, \sigma_2u_2 \cdots u_{n-1}\sigma_nu_n) \\
&= \theta_{G,f}(x_0, u_0\sigma_1u_1\sigma_2u_2 \cdots u_{n-1}\sigma_nu_n) = \theta_{G,f}(x_0, s)
\end{aligned}$$

In either case, we have $v \geq \theta_{G,f}(x_0, s)$. Thus, $v \geq \max_{s \in K_*} \theta_{G,f}(x_0, s) = \omega_{G,f}(K_*)$. ■

By Prop. 3.2 we know that the finite output of PP1 is no smaller than the weight of a controllable and normal sublanguage of $L_m(G)$. By an extended argument we actually can show that $v = \omega_{G,f}(K_*)$, which means the finite output of PP1 is actually the weight of a controllable and normal sublanguage of $L_m(G)$. But owing to the limited space, we are content with $v \geq \omega_{G,f}(K_*)$, which is sufficient for our purpose. The next result indicates that, the existence of a controllable and normal sublanguage of $L_m(G)$ implies the finite output of PP1.

Proposition 3.3. Given a weighted automaton $(G, f) \in \Phi(\Sigma)$, if $\mathcal{WCN}(G, f, G) \neq \varnothing$, then the output v of PP1 is finite. □

Proof: Suppose $G = (X, \Sigma, \xi, x_0, X_m)$. Since $\mathcal{WCN}(G, f, G) \neq \varnothing$ and G is zero-weighted-loop-free, we know that there exists a finite $K \in \mathcal{WCN}(G, f, G)$. Let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m)$ be the observation behavior of (G, f). Let $\Omega := \{z \in Z | (\exists s \in \overline{K})\, \delta(z_0, P_o(s)) = z\}$. Since K is normal with respect to G and P_o, we know that, if $z \in \Omega$, then for all $x \in z$, there exists $s \in \overline{K}$ such that $x = \xi(x_0, s)$. Let $\gamma(z) := \{\sigma \in \Sigma_o | \neg\delta(z, \sigma)! \vee (\exists s \in \overline{K})\, \delta(z_0, P_o(s)) = z \wedge s\sigma \in \overline{K}\}$. Since K is controllable with respect to G and normal with respect to G and P_o, we get that

$$\Sigma_{uc} \cap \Sigma_o \cap \mu_{\hat{G}}(z) \subseteq \gamma(z)$$

which means $\gamma(z) \in \Gamma_z$ for each $z \in \Omega$. Sice $K \in \mathcal{WCN}(G, f, G)$, namely $\omega_{G,f}(K) < +\infty$, by the definition of $\hat{G}$ and PP1, we can get that the output v is finite. ■

From Prop. 3.2 and Prop. 3.3 we can see that, the finiteness of the output v of PP1 can be used to decide the emptiness of $\mathcal{WCN}(G, f, G)$. In addition, when $v < +\infty$, by Prop. 3.2 we can compute one $K \in \mathcal{WCN}(G, f, G)$. This allows us to present the following procedure to solve WSCP1 and WSCP2.

Procedure for Solving WSCP1 and WSCP2 (PWSCP12):

1. Input: a plant $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$
2. Compute $K = \sup\mathcal{CN}(G, E)$.
3. If $K = \varnothing$ then set $K_{CN} = \varnothing$ and go to step (6).
4. Let G' recognizes K and is homomorphic to G, and f' is induced from (G, f).
5. Apply PP1 to (G', f'). If the output $v = +\infty$, then set $K_{CN} := \varnothing$ and go to step (6). Otherwise, let $K_{CN} := K_*$, where K_* is the output of PP1.
6. Output: K_{CN} □

In PWSCP12 we use Step (2) to handle the requirement E. Since the complexity of computing $\sup\mathcal{CN}(G, E)$ is exponential time [13] with respect to the size of the product of G and E, and the complexity of PP1 is exponential time with respect to the size of $\sup\mathcal{CN}(G, E)$, we may think that the complexity of PWSCP12 is double exponential. Fortunately, our luck is not that bad. If $\sup\mathcal{CN}(G, E)$ is computed based on the power-set construction to take partial observation into consideration, then the construction of $\hat{G}$ from $\sup\mathcal{CN}(G, E)$ will be only polynomial time with respect to the size of the resulting automaton obtained from the power-set construction. Thus, in the end the complexity of PWSCP12 is exponential time with respect to the size of G.

Lemma 3.4. PWSCP12 terminates. □

Proof: By Prop. 3.1 we know that PP1 terminates. Thus, PWSCP12 terminates. ■

Theorem 3.5. Given a plant $(G, f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$, let K_{CN} be computed by PWSCP12. Then (1) $K_{CN} = \varnothing$ if and only if $\mathcal{WCN}(G, f, E) = \varnothing$; (2) When $K_{CN} \neq \varnothing$, we have $K_{CN} \in \mathcal{WCN}(G, f, E)$. □

Proof: (1) Let $K = \sup\mathcal{CN}(G, E)$. Suppose $K_{CN} = \varnothing$. Then we have two cases to consider: Case 1.1: $K = \varnothing$, namely $L_m(G) \cap L_m(E)$ has no controllable and normal sublanguage with respect to G; Case 1.2: the output v of PP1 is infinite. For Cases 1.1, clearly, $\mathcal{WCN}(G, f, E) = \varnothing$. For Case 1.2, by Prop. 3.3 we know that $\mathcal{WCN}(G', f', G') = \varnothing$, which means $\mathcal{WCN}(G, f, E) = \varnothing$ because $L_m(G) = \sup\mathcal{CN}(G, E)$.
On the other hand, if $\mathcal{WCN}(G, f, E) = \varnothing$, then either $L_m(G) \cap L_m(E)$ has no controllable and normal sublanguage or there is no controllable and normal sublanguage with a finite weight. In the former case, clearly, $K_{CN} = \varnothing$ because $K = \varnothing$. In the latter case, we have $\mathcal{WCN}(G', f', G') = \varnothing$. By prop. 3.2, the output v of PP1 is infinite. Thus, we have $K_{CN} = \varnothing$.
(2) Suppose $K_{CN} \neq \varnothing$. Then $K_{CN} = K_*$, where K_* is the output of PP1. By Prop. 3.2 we have $K_* \in \mathcal{WCN}(G', f', G')$. Since $L_m(G') = \sup\mathcal{CN}(G, E)$, we have $K_* \in \mathcal{CN}(G, E)$. Since $\omega_{G',f'}(K_*) < +\infty$ and f' is induced from (G, f), we have $K_{CN} = K_* \in \mathcal{WCN}(G, f, E)$. ■

By Theorem 3.5 we know that, when PWSCP12 terminates, WSCP1 can be solved by simply checking the nonemptiness of K_{CN}. When $K_{CN} \neq \varnothing$, then it is a solution to WSCP2. Next, we present an algorithm to solve WSCP3.

Procedure for Solving WSCP3 (PWSCP3):

1. Input: a plant $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$, a requirement $E \in \phi(\Sigma)$
2. Compute $K = \sup\mathcal{CN}(G, E)$.
3. If $K = \varnothing$ then set $K_{CN} = \varnothing$ and go to step (7).
4. Let G' recognizes K and is homomorphic to G, and f' is induced from (G, f).
5. Apply PP1 to (G', f'). When the output $v < +\infty$, let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m)$ be the observable behavior of G' and suppose the termination condition of PP1 holds at r. Let S' be a recognizer of the sublanguage
$$\{s \in K | \theta_{G,f}(x_0, s) \leq v \wedge (\forall s' \leq s)\, \kappa_r(\delta(z_0, P_o(s'))) < +\infty\}$$
Compute
$$\hat{K} = \sup\mathcal{CN}(G, S')$$
6. Set $\hat{K}_0 := \hat{K}$ and iterates on $r = 0, 1, \cdots$, as follows:
   (a) Search a set $\psi(\hat{K}_r) := \{s \in \hat{K}_r | \theta_{G,f}(x_0, s) = \omega_{G,f}(\hat{K}_r)\}$.
   (b) Compute the largest $\hat{K}_{r+1} \subseteq \hat{K}_r - \psi(\hat{K}_r)$ that is controllable and normal w.r.t. G.
   (c) If $\hat{K}_{r+1} = \varnothing$ then set $K_{CN} := \hat{K}_r$ and go to step (7). Otherwise, continue on $r + 1$.
7. Output: K_{CN} □

What PWSCP3 does is to fist run PWSCP12 but output v and $\hat{G}$, then based on v and $\hat{G}$ to construct a sublanguage $\hat{K}$. It will be shown that the supremal controllable and normal sublanguage of (G, f) with respect to E is contained in $\hat{K}$. Step (6) is used to find such a supremal solution by continuously taking out strings with the maximum weight until no such strings can be taken out without nullifying the chance of finding a controllable and normal sublanguage. As we have known, the complexity of PWSCP12 is exponential time. Step (6) in PWSCP3 makes the situation even worse because it requires to enumerate all strings in $\hat{K}_0$, and compute a finite number of controllable and normal sublanguages $\hat{K}_r$. So the complexity of PWSCP3 is at least exponential time.

Lemma 3.6. PWSCP3 terminates. □

Proof: Since v is finite, and (G, f) is zero-weighted-loop-free, we know that $\hat{K}_r$ is finite. Thus, the set $\psi(\hat{K}_r)$ can be computed, e.g. by simply enumerating each string in $\hat{K}_r$ and determining its weight. Thus, PWSCP3 terminates. ■

Theorem 3.7. Given a plant $(G, f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$, let K_{CN} be computed by PWSCP3. When $K_{CN} \neq \varnothing$, we have $K_{CN} = \sup\Xi(G, f, E)$. □

Proof: Since $K_{CN} \neq \varnothing$, we know that $\mathcal{WCN}(G, f, E) = \mathcal{WCN}(G', f', G') \neq \varnothing$. By Prop. 3.3, the output v of PP1 is finite. Furthermore, by Prop. 3.2 we know that, there exists $K' \in \mathcal{WCN}(G', f', G')$ such that

$$\omega_{G,f}(\sup\Xi(G, f, E)) \leq \omega_{G,f}(K') = \omega_{G',f'}(K') \leq v$$

Since $\sup\Xi(G, f, E) \in \mathcal{WCN}(G', f', G')$, we can derive that, for every $s \in \sup\Xi(G, f, E)$ and for all $s' \leq s$, we have $\kappa_r(\delta(z_0, P_o(s'))) < +\infty$. Thus, $\sup\Xi(G, f, E) \subseteq L_m(S')$, which means $\sup\Xi(G, f, E) \subseteq \hat{K}_0 \in \mathcal{WCN}(G, f, E)$. Suppose step (6) terminates at $r + 1$ with $r \geq 0$. We use induction to show that, $\sup\Xi(G, f, E) \subseteq \hat{K}_r$. It is true for $l = 0$. We assume that it is also true for l and we need to show that, it is true for $l + 1 \leq r$. Since $\sup\Xi(G, f, E) \subseteq \hat{K}_l$, we have

$$\omega_{G,f}(\hat{K}_l) \geq \omega_{G,f}(\sup\Xi(G, f, E))$$

For any $s \in \hat{K}_l$ with $\theta_{G,f}(x_0, s) = \omega_{G,f}(\hat{K}_l)$, we have $s \in \psi(\hat{K}_l)$. Since $\hat{K}_r \neq \varnothing$ and $l < r$, we have $\hat{K}_l - \psi(\hat{K}_l) \neq \varnothing$. Then

$$\omega_{G,f}(\hat{K}_l - \psi(\hat{K}_l)) < \omega_{G,f}(\hat{K}_l)$$

Let $\hat{K}_{l+1}$ be the largest controllable and normal sublanguage of $\hat{K}_l - \psi(\hat{K}_l)$ with respect to G. Since $l + 1 \leq r$, we have $\hat{K}_{l+1} \neq \varnothing$. Thus, $\hat{K}_{l+1} \in \Xi(G, f, E)$, which means

$$\omega_{G,f}(\sup\Xi(G, f, E)) \leq \omega_{G,f}(\hat{K}_{l+1}) < \omega_{G,f}(\hat{K}_l)$$

Since $\sup\Xi(G, f, E) \subseteq \hat{K}_l$ and for any sublanguage $W \subseteq \hat{K}_l$, which is controllable and normal with respect to G, we have

$$\omega_{G,f}(W) < \omega_{G,f}(\hat{K}_l) \Rightarrow W \subseteq \hat{K}_{l+1}$$

we get that $\sup\Xi(G, f, E) \subseteq \hat{K}_{l+1}$. Thus, the induction is true, namely $\sup\Xi(G, f, E) \subseteq \hat{K}_r$. Since $\hat{K}_{r+1} = \varnothing$, for any controllable and normal sublanguage $W \subseteq \hat{K}_r$ with respect to G, we get that, $W \cap \psi(\hat{K}_r) \neq \varnothing$, which means $\omega_{G,f}(W) = \omega_{G,f}(\hat{K}_r)$. Since $\sup\Xi(G, f, E) \subseteq \hat{K}_r$, we get $\omega_{G,f}(\sup\Xi(G, f, E)) = \omega_{G,f}(\hat{K}_r)$. Thus, $\sup\Xi(G, f, E) = \hat{K}_r = K_{CN}$. ■

Theorem 3.7 confirms that, we can solve WSCP3 by using PWSCP3. Since Step 6 in PWSCP3 is computationally intensive, naturally, we want to know whether there exists a polynomial-time algorithm to solve WSCP3. Unfortunately, it can be shown that solving WSCP3 is NP-hard.

We now apply PWSCP12 and PWSCP3 to a small example to illustrate their main steps. Suppose we have a weighted automaton $(G, f) \in \Phi(\Sigma)$ depicted in Figure 2, where $\Sigma = \{a, b, c, u_1, u_2, u_3, u_4\}$, $\Sigma_c = \Sigma_o = \{a, b, c\}$. Since $L_m(E) = \Sigma^*$, we get that

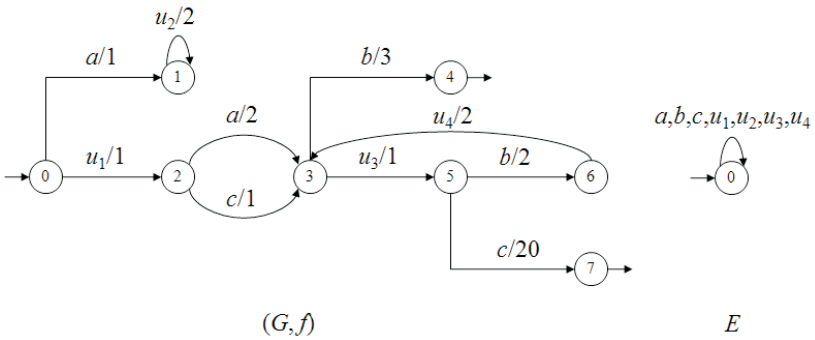

Figure 2: Example 2: Weighted automaton $(G, f) \in \Phi(\Sigma)$ and requirement $E \in \phi(\Sigma)$

$K = \sup\mathcal{CN}(G, E) = L_m(G)$. Let $(G', f') = (G, f)$. We now apply PP1 to (G', f'). We first compute the observable behavior $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m) \in \phi(\Sigma_o)$, which is depicted in Figure 3. When $k = 0$ we have $\kappa_0(z_0) = \kappa_0(z_1) = \kappa_0(z_2) = \kappa_0(z_3) = +\infty$ and

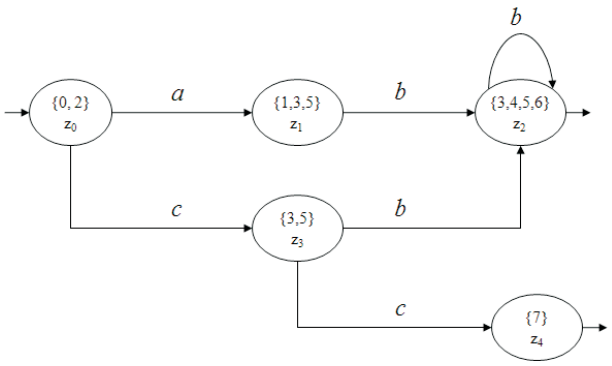

Figure 3: Example 2: Observable behavior $\hat{G} \in \phi(\Sigma_0)$

$\kappa_0(z_4) = 0$. When $k = 1$, we have $\kappa_1(z_0) = \kappa_1(z_1) = \kappa_1(z_2) = +\infty$, $\kappa_1(z_3) = 21$ with $\gamma_{z_3} = \{c\}$, $\kappa_1(z_4) = 0$. Since $\kappa_0(z_3) = +\infty$ and $\kappa_1(z_3) < +\infty$, we get that $\kappa_1(z_3, 3) = 21$ and $\kappa_1(z_3, 5) = 20$. When $k = 2$, we have $\kappa_2(z_1) = \kappa_2(z_2) = +\infty$, $\kappa_2(z_0) = 23$ with $\gamma_{z_0} = \{c\}$, $\kappa_2(z_3) = 21$ and $\kappa_2(z_4) = 0$. Since $\kappa_1(z_0) = +\infty$ and $\kappa_2(z_0) < +\infty$, we have $\kappa_2(z_0, 0) = 23$ and $\kappa_2(z_0, 2) = 22$. When $k = 3$, we have $\kappa_3(z_1) = \kappa_3(z_2) = +\infty$, $\kappa_3(z_0) = 23$, $\kappa_3(z_3) = 21$ and $\kappa_3(z_4) = 0$. Thus, the termination condition holds at $k = 3$. The outputs of PP1 are $v = \kappa_3(z_0, 0) = 23$, and $K_* = \{u_1cu_3c\}$. Thus, the output of PWSCP12 is $K_{CN} = K_* = \{u_1cu_3c\}$. If we apply PWSCP3, then it is trivial to check that

the language recognized by S' is $\{u_1cu_3c\}$. Thus, $\hat{K}_0 = \sup\mathcal{CN}(G, S') = \{u_1cu_3c\}$. Since $\hat{K}_0$ is a singleton, we have $\hat{K}_1 = \varnothing$ because $\psi(\hat{K}_0) = \hat{K}_0$. Thus, we have $K_{CN} = \{u_1cu_3c\}$, which coincides with the output of PWSCP12.

3.2 An algorithm for solving WSCP in a special setting

The aforementioned algorithms PWSCP12 and PWSCP3 depend on PP1, which has two costly computational steps: to construct the observable behavior $\hat{G}$, and to utilize functions $\hat{h}$ and $\rho[z, z']$ with $z, z' \in Z$, where Z is the state set of $\hat{G}$. It is natural to ask under what condition(s) we can reduce or even eliminate such costly computations. One answer is to require the natural projection $P_o : \Sigma^* \to \Sigma_o^*$ be an appropriate *natural observer*, whose definition is provided below.

Definition 3.8. [12] Given a language $L \subseteq \Sigma^*$ and an alphabet $\Sigma' \subseteq \Sigma$, we say the natural projection $P : \Sigma^* \to \Sigma'^*$ is a *natural observer* with respect to L, if

$$(\forall t \in P(L))(\forall s \in \overline{L})\, P(s) \leq t \Rightarrow (\exists s' \in \Sigma^*)\, ss' \in L \wedge P(ss') = t$$

For simplicity, we also call P an L-*observer*. □

If P_o is an L-observer, then any string $s \in \overline{L}$, whose projected image $P_o(s)$ is a prefix substring of $t \in P_o(L)$ must be extendable to a string in L by concatenating a string s' such that the projected image of ss' is t. We have the following result.

Theorem 3.9. Given a nonblocking $G \in \phi(\Sigma)$, let $P_o : \Sigma^* \to \Sigma_o^*$ be the natural projection. Let $G' \in \phi(\Sigma_o)$ such that $L_m(G') = P_o(L_m(G))$ and $L(G') = P_o(L(G))$. If P_o is an $L_m(G)$-observer, then we have the following results:

1. If a sublanguage $L' \subseteq P_o(L_m(G))$ is controllable with respect to G', then $L'||L_m(G)$ is controllable and normal with respect to G and P_o.
2. If a sublanguage $\hat{L} \subseteq L_m(G)$ is controllable and normal with respect to G, then $P_o(\hat{L})$ is controllable with respect to G'. □

Proof: (1) Suppose L' is controllable with respect to G'. We first show that $L'||L_m(G)$ is controllable with respect to G. Let $s \in \overline{L'||L_m(G)}$ and $\sigma \in \Sigma_{uc}$. Suppose $s\sigma \in L(G)$. We have two cases to consider.
Case 1: $\sigma \in \Sigma_o$. Then $P_o(s\sigma) = P_o(s)\sigma \in P_o(L(G)) = L(G')$. Since $s \in L'||L_m(G)$, we have $P_o(s) \in P_o(\overline{L'||L_m(G)}) \subseteq \overline{L'}$. Since L' is controllable with respect to G', we get that $P_o(s)\sigma \in \overline{L'}$, which means $P_o(s\sigma) \in \overline{L'}$. Thus, $s\sigma \in \overline{L'}||L(G) = \overline{L'}||\overline{L_m(G)}$. Since P_o is an $L_m(G)$-observer, we have $\overline{L'}||\overline{L_m(G)} = \overline{L'||L_m(G)}$. Thus, $s\sigma \in \overline{L'||L_m(G)}$.
Case 2: $\sigma \notin \Sigma_o$. Since $s \in \overline{L'||L_m(G)} = \overline{L'}||L(G)$ and $s\sigma \in L(G)$, we have $s\sigma \in \overline{L'}||L(G) = \overline{L'||L_m(G)}$.
In either case we have $s\sigma \in \overline{L'||L_m(G)}$. Thus, $L'||L_m(G)$ is controllable with respect to G,
To show $L'||L_m(G)$ is normal with respect to G and P_o, we have

$$L(G) \cap P_o^{-1}P_o(\overline{L'||L_m(G)}) = L(G)||P_o(\overline{L'}||L(G)) = L(G)||\overline{L'}||P_o(L(G)) = L(G)||\overline{L'} = \overline{L'||L_m}$$

(2) Suppose $\hat{L}$ is controllable and normal with respect to G. We show that $P_o(\hat{L})$ is controllable with respect to G'. Let $s \in \overline{P_o(\hat{L})}$ and $\sigma \in \Sigma_{uc} \cap \Sigma_o$. Suppose $s\sigma \in$

$L(G') = P_o(L(G))$. Since $s \in \overline{P_o(\hat{L})}$, there must exist $t \in \overline{\hat{L}}$ such that $P_o(t) = s$. Since $s\sigma \in L(G')$ and P_o is $L_m(G)$-observer and $L(G') = \overline{L_m(G')} = P_o(\overline{L_m(G)})$, there must exist $tt'\sigma \in L(G)$ such that $P_o(tt') = s$. Since $t \in \overline{\hat{L}}$ and $P_o(t) = P_o(tt') = s$, we know that $tt' \in \overline{\hat{L}}$ because $\hat{L}$ is normal with respect to G. Since $\sigma \in \Sigma_{uc}$ and $\overline{\hat{L}}$ is controllable with respect to G, we have $tt'\sigma \in \overline{\hat{L}}$. Thus, $s\sigma = P_o(tt'\sigma) \in P_o(\overline{\hat{L}}) = \overline{P_o(\hat{L})}$. ■

By Theorem 3.9 and the fact that two different normal sublanguages of $L_m(G)$ must have different projected images, we can derive that, there is a one-to-one mapping between $\mathcal{CN}(G, G)$ and $\mathcal{C}(G', G')$. This one-to-one mapping also holds between $\mathcal{CN}(G, E)$ and $\mathcal{C}(G', E)$ for all $E \in \phi(\Sigma_o)$. In particular, there is a direct connection between $\sup\mathcal{CN}(G, E)$ and $\sup\mathcal{C}(G', E)$, which is described in the following corollary.

Corollary 3.10. Given a nonblocking $G \in \phi(\Sigma)$ and a requirement $E \in \phi(\Sigma_o)$, let $G' \in \phi(\Sigma_o)$ such that $L_m(G') = P_o(L_m(G))$ and $L(G') = \overline{L_m(G')}$. Suppose P_o is an $L_m(G)$-observer. Then $\sup\mathcal{CN}(G, E) = \sup\mathcal{C}(G', E)||L_m(G)$. □

Proof: For all $K' \in \sup\mathcal{C}(G', E)$, by Theorem 3.9 we have $K'||L_m(G) \in \mathcal{CN}(G, E)$. Thus, $K'||L_m(G) \subseteq \sup\mathcal{CN}(G, E)$. In particular, $\sup\mathcal{C}(G', E)||L_m(G) \subseteq \sup\mathcal{CN}(G, E)$.
On the other hand, for all $K \in \mathcal{CN}(G, E)$, by Theorem 3.9 we have $P_o(K) \in \mathcal{C}(G', E)$, which means $P_o(K) \subseteq \sup\mathcal{C}(G', E)$. In particular, $P_o(\sup\mathcal{CN}(G, E)) \subseteq \sup\mathcal{C}(G', E)$. Thus, $\sup\mathcal{CN}(G, E) \subseteq P_o^{-1}(\sup\mathcal{C}(G', E)) \cap L_m(G) = \sup\mathcal{C}(G', E)||L_m(G)$. The corollary follows. ■

Given $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$, let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m)$ be the observable behavior of G. We define a (partial) weight function $\hat{f} : Z \times \Sigma_o \to \mathbb{R}^+ \cup \{+\infty\}$ as follows:

$$(\forall z \in Z)(\forall \sigma \in \Sigma_o)\, \hat{f}(z, \sigma) := \begin{cases} \sup_{s \in P_o^{-1}(\sigma) \wedge (\exists x \in z)\, \xi(x,s)!} \theta_{G,f}(x, s) & \text{if } \delta(z, \sigma)! \\ \textit{not defined} & \text{otherwise} \end{cases}$$

We call $(\hat{G}, \hat{f})$ the *over-weighted observable behavior* of (G, f). As an illustration, let (G, f) be depicted in the left picture of Figure 4, where $\Sigma = \{b, c, u_1, u_2\}$ and $\Sigma_o = \{b, c\}$.

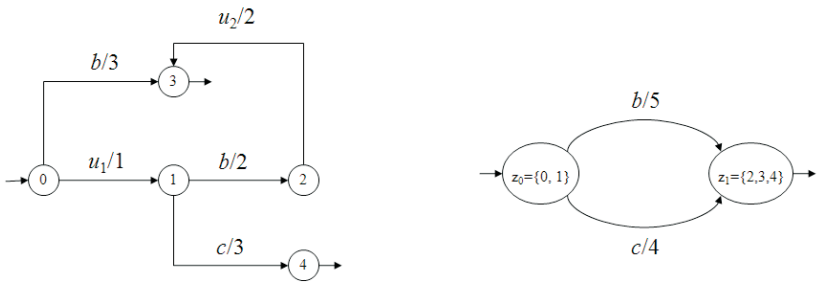

Figure 4: Example 3: (G, f) (left) and the over-weighted observable behavior $(\hat{G}, \hat{f})$ (right)

The corresponding observable behavior $\hat{G}$ is depicted in the right picture of Figure 4. To compute $\hat{f}(z_0, b)$, we first determine the set $\{\{s \in \Sigma^* | P_o(s) = b \wedge \xi(0, s)!\}, \{s \in$

$\Sigma^*|P_o(s) = b \wedge \xi(1,s)!]\}$, which can be easily found as $\{\{b, u_1bu_2\}, \{bu_2\}\}$. Then we compute

$$\max\{\theta_{G,f}(0,b), \theta_{G,f}(0,u_1bu_2), \theta_{G,f}(1,bu_2)\} = \max\{3,5,4\} = 5$$

Thus, we have $\hat{f}(z_0,b) = 5$. Similarly, we can get $\hat{f}(z_0,c) = 4$. Using a procedure defined in [12] we can check whether the projection P_o is an $L_m(G)$-observer.

Theorem 3.11. Given $(G,f) \in \Phi(\Sigma)$, let $(\hat{G},\hat{f}) \in \Phi(\Sigma_o)$ be constructed as above. Suppose $P_o : \Sigma^* \to \Sigma_o^*$ is an $L_m(G)$-observer. For each $K \in \mathcal{CN}(G,G)$, if $P_o(K) \neq \{\epsilon\}$, then we have $\omega_{G,f}(K) \leq \omega_{\hat{G},\hat{f}}(P_o(K))$. In addition, if $\omega_{G,f}(K) < +\infty$ then $\omega_{\hat{G},\hat{f}}(P_o(K)) < +\infty$. □

Proof: (1) Since $P_o(K) \neq \{\epsilon\}$ and P_o is an L_m-observer, we can derive that, for each $s \in K$ with $P_o(s) = \epsilon$, there must exist $s' \in \Sigma^*$ with $P_o(s') \neq \epsilon$ such that $ss' \in K$. Let $K' := \{s \in K | P_o(s) \neq \epsilon\}$. Clearly, $\omega_{G,f}(K') = \omega_{G,f}(K)$. Thus, to show that $\omega_{G,f}(K) \leq \omega_{\hat{G},\hat{f}}(P_o(K))$, we only need to show that, for each $s \in K'$, $\theta_{G,f}(x_0,s) \leq \theta_{\hat{G},\hat{f}}(z_0,P_o(s))$. For each $s \in K'$, there must exist $u_1,\cdots,u_{n+1} \in \Sigma_{uo}^*$ and $\sigma_1,\cdots,\sigma_n \in \Sigma_o$ such that $s = u_1\sigma_1u_2\sigma_2\cdots u_n\sigma_nu_{n+1}$. Let $x_1,\cdots,x_n \in X$ such that $x_1 = \xi(x_0,u_1\sigma_1u_2)$ and $x_{i+1} = \xi(x_i,\sigma_iu_{i+1})$ for $i = 2,\cdots,n$. Similarly, let $z_1,\cdots,z_n \in Z$ such that $z_1 = \delta(z_0,\sigma_1)$ and $z_{i+1} = \delta(z_i,\sigma_i)$ for $i = 2,\cdots,n$. By the construction of $(\hat{G},\hat{f})$, we get that, $x_1 \in z_1$ and $\theta_{G,f}(x_0,u_1\sigma_1u_2) \leq \theta_{\hat{G},\hat{f}}(z_0,\sigma_1)$. Furthermore, $x_i \in z_i$ and $\theta_{G,f}(x_i,\sigma_iu_{i+1}) \leq \theta_{\hat{G},\hat{f}}(z_i,\sigma_i)$ for $i = 1,\cdots,n$. Thus, $\theta_{G,f}(x_0,s) \leq \theta_{\hat{G},\hat{f}}(z_0,P_o(s))$.
(2) To show that $\omega_{G,f}(K) < +\infty$ implies $\omega_{\hat{G},\hat{F}}(P_o(K)) < +\infty$, suppose it is not true. Then there exists a proper subset $K \subseteq L_m(G)$, which is controllable and normal with respect to G, and $\omega_{G,f}(K) < +\infty$ but $\omega_{\hat{G},\hat{f}}(P_o(K)) = +\infty$. Since $\omega_{G,f}(K) < +\infty$, we know that there is no 'cycle' in K, whose weight is nonzero, namely, for each sublanguage $s_1s_2^*s_3 \in K$, we have $\theta_{G,f}(\xi(x_0,s_1),s_2) = 0$. Thus, if $\omega_{\hat{G},\hat{f}}(P_o(K)) = +\infty$, then there is only one possibility, namely there must exist $z \in Z$ and $\sigma \in \Sigma_o$ such that $\hat{f}(z,\sigma) = +\infty$. This means, there exist $x \in z$ such that

$$\max_{s\in\Sigma^*:P_o(s)=\sigma\,\wedge\,(\exists x\in z)\,\xi(x,s)!} \theta_{G,f}(x,s) = +\infty$$

Clearly, either $x \notin \{x' \in X | (\exists s \in K)\, \xi(x_0,s) = x'\}$ or there exists $s' \in \Sigma^*$ with $P_o(s') = \sigma$ such that $\overline{K}s' \notin \overline{K}$. In either case, K is not normal with respect to G, which contradicts the assumption that K is controllable and normal with respect to G. Thus, we have $\omega_{\hat{G},\hat{f}}(P_o(K)) < +\infty$. ■

Since in $\hat{G}$ all events are observable, every sublanguage of $L_m(\hat{G})$ is normal with respect to $\hat{G}$ and $\hat{P}_o : \Sigma_o^* \to \Sigma_o^*$. Thus, instead of writing $\mathcal{WCN}(\hat{G},\hat{f},\hat{G})$, we simply write $\mathcal{WC}(\hat{G},\hat{f},\hat{G})$ to denote all controllable sublanguages of $L_m(\hat{G})$, whose weight is finite. By Theorem 3.9 and Theorem 3.11 we can derive the following important result.

Corollary 3.12. Given $(G,f) \in \Phi(\Sigma)$, let $(\hat{G},\hat{f}) \in \Phi(\Sigma_o)$ be the over-weighted observable behavior of (G,f). Suppose $P_o : \Sigma^* \to \Sigma_o^*$ is an $L_m(G)$-observer, and for all $K \in \mathcal{CN}(G,G)$, $P_o(K) \neq \{\epsilon\}$. Then (1) $\mathcal{WCN}(G,f,G) \neq \varnothing$ if and only if $\mathcal{WC}(\hat{G},\hat{f},\hat{G}) \neq \varnothing$; (2) for all $\hat{K} \in \mathcal{WC}(\hat{G},\hat{f},\hat{G})$, there exists $K \in \mathcal{WCN}(G,f,G)$ such that $\omega_{G,f}(K) \leq \omega_{\hat{G},\hat{f}}(\hat{K})$; (3) $\omega_{G,f}(\sup\Xi(G,f,G)) \leq \omega_{\hat{G},\hat{f}}(\sup\Xi(\hat{G},\hat{f},\hat{G}))$. □

Proof: (1) Suppose $\mathcal{CN}(G,f,G) \neq \varnothing$. Then there exists $K \in \mathcal{CN}(G,f,G)$, which means $\omega_{G,f}(K) < +\infty$. Since $P_o(K) \neq \{\epsilon\}$. By Theorem 3.11 we know that $\omega_{\hat{G},\hat{f}}(P_o(K)) <$

$+\infty$. By Theorem 3.9 we get that $P_o(K) \in \mathcal{C}(\hat{G}, \hat{G})$. Thus, $\mathcal{WC}(\hat{G}, \hat{f}, \hat{G}) \neq \varnothing$.
Suppose $\mathcal{WC}(\hat{G}, \hat{f}, \hat{G}) \neq \varnothing$. Then there exists $\hat{K} \in \mathcal{WC}(\hat{G}, \hat{f}, \hat{G})$, namely $\omega_{\hat{G},\hat{f}}(\hat{K}) < +\infty$. Let $K = P_o^{-1}(\hat{K})||L_m(G)$. Clearly, $P_o(K) = \hat{K}$. By Theorem 3.11 we have $\omega_{G,f}(K) \leq \omega_{\hat{G},\hat{f}}(\hat{K}) < +\infty$. By Theorem 3.9 we have $K \in \mathcal{CN}(G, G)$. Thus, $\mathcal{CN}(G, f, G) \neq \varnothing$. This also proves (2).
(3) Set $\hat{K} = \sup\Xi(\hat{G}, \hat{f}, \hat{G})$. Then

$$\omega_{G,f}(P_o^{-1}(\sup\Xi(\hat{G}, \hat{f}, \hat{G}))||L_m(G)) \leq \omega_{\hat{G},\hat{f}}(\sup\Xi(\hat{G}, \hat{f}, \hat{G})) < +\infty$$

By Theorem 3.9 we know that $P_o^{-1}(\sup\Xi(\hat{G}, \hat{f}, \hat{G}))||L_m(G) \in \mathcal{CN}(G, G)$. Thus,

$$\omega_{G,f}(\sup\Xi(G, f, G)) \leq \omega_{G,f}(P_o^{-1}(\sup\Xi(\hat{G}, \hat{f}, \hat{G}))||L_m(G))$$

which means $\omega_{G,f}(\sup\Xi(G, f, G)) \leq \omega_{\hat{G},\hat{f}}(\sup\Xi(\hat{G}, \hat{f}, \hat{G}))$. ■

Corollary 3.12 indicates that, as long as $P_o(K) \neq \{\epsilon\}$ for each $K \in \mathcal{CN}(G, G)$, we can determine the emptiness of $\mathcal{WCN}(G, f, G)$ by determining the emptiness of $\mathcal{WC}(\hat{G}, \hat{f}, \hat{G})$. Furthermore, when $\mathcal{WC}(\hat{G}, \hat{f}, \hat{G}) \neq \varnothing$, the value $\omega_{\hat{G},\hat{f}}(\sup\Xi(\hat{G}, \hat{f}, \hat{G}))$ is an upper bound of $\omega_{G,f}(\sup\Xi(G, f, G))$, and can be computed by the following polynomial algorithm.

Procedure for Supremal Minimum-Weighted Controllable Sublanguages (SMC):

1. Input: a weighted plant $(\hat{G} = (Z, \hat{\Sigma}, \delta, z_0, Z_m), \hat{f})$, where the controllable alphabet is $\hat{\Sigma}_c$ and the observable alphabet is $\hat{\Sigma}_o = \hat{\Sigma}$
2. Initialization: for each $z \in Z$, if $z \in Z_m$ then set $\kappa_0(z) := 0$; otherwise, set $\kappa_0(z) := +\infty$.
3. Iterate on $k = 1, 2, \cdots$, as follows:
   (a) For each $z \in Z$ we have

$$\kappa_k(z) := \begin{cases} \max_{\sigma \in \hat{\Sigma}_{uc} \cap \mu_{\hat{G}}(z)}(\hat{f}(z, \sigma) + \kappa_{k-1}(\delta(z, \sigma))) & \text{if } \mu_{\hat{G}}(z) \cap \hat{\Sigma}_{uc} \neq \varnothing \\ \min_{\sigma' \in \mu_{\hat{G}}(z)}(\hat{f}(z, \sigma') + \kappa_{k-1}(\delta(z, \sigma'))) & \text{if } \varnothing \neq \mu_{\hat{G}}(z) \subseteq \hat{\Sigma}_c \\ \kappa_{k-1}(z) & \text{otherwise} \end{cases}$$

   (b) Termination when: $(\exists r \in \mathbb{N})(\forall z \in Z)\, \kappa_{r-1}(z) = \kappa_r(z)$
4. Set $v := \kappa_r(z_0)$ and go to step (5).
5. Output v. □

SMC is almost the same as the one used in [8], which is aimed to compute the least restrictive supervisor that can make a system move from its initial state to a marker state with the smallest "worst-case" cost. The only difference is that, in [8] marker states may not be necessarily deadlock states but there is no transition from a marker state to a nonmarker state. Thus, once a marker state is reached, only marker states can be visited in the future. In our case this condition is automatically satisfied because of the marking deadlock assumption. By an argument similar to the one in [8], we can show that the value $\kappa_k(z)$ is the smallest "worst-case" cost of all strings from state z to a marker state in $(\hat{G}, \hat{f})$ with no more than k transitions, thus, the maximum value for k in SMC is no more than the length of the longest string in $L_m(\hat{G})$. In addition, the overall complexity

of SMC is polynomial time. We now present the following procedure to solve WSCP1 and WSCP2.

Procedure for Solving WSCP1 and WSCP2 (in the special setting) (SPWSCP12):

1. Input: a plant $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$
2. Compute $K = \sup\mathcal{CN}(G, E)$.
3. If $K = \varnothing$ then set $K_{CN} = \varnothing$ and go to step (8).
4. If $K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$, then set $K_{CN} = K \cap \Sigma_{uo}^*$ and go to step (8), if $\omega_{G,f}(K \cap \Sigma_{uo}^*) < +\infty$; or set $K_{CN} = \varnothing$ and go to step (8), if $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$.
5. Let G' recognizes K and is homomorphic to G, and f' is induced from (G, f).
6. Construct $(\hat{G}, \hat{f}) \in \Phi(\Sigma_o)$, which is the over-weighted observable behavior of (G', f').
7. Apply SMC on $(\hat{G}, \hat{f})$. If the output $v = +\infty$, then set $K_{CN} := \varnothing$ and go to step (8). Otherwise, let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m)$ and suppose SMC terminates at r. Let $S = (Z', \Sigma_o, \delta', z_0, Z'_m)$ where
   (a) $Z' := \{z \in Z | \kappa_r(z) < +\infty\}$
   (b) $Z'_m := Z' \cap Z_m$
   (c) $\delta' : Z' \times \Sigma_o \to Z'$, where for any $(z, \sigma) \in Z' \times \Sigma_o$,
   $$\delta'(z, \sigma) := \begin{cases} \delta(z, \sigma) & \text{if } \delta(z, \sigma) \in Z' \wedge \hat{f}(z, \sigma) + \kappa_r(z') \leq \kappa_r(z) \\ \textit{not defined} & \text{otherwise} \end{cases}$$
   Let $K_{CN} := P_o^{-1}(L_m(S)) || L_m(G)$.
8. Output: K_{CN} □

The main idea of SPWSCP12 is to project out all unobservable transitions and synthesize a finitely-weighted supervisor based on the projected model. But to do that, we need to make sure that (G, f) has no controllable and normal sublanguage only consisting of unobservable transitions. This is why we need Step (4) in SPWSCP12. If there does exist such a controllable and normal sublanguage $K \cap \Sigma_{uo}^*$, then, by normality we know that only observable and controllable events can be disabled. Therefore, any controllable and normal sublanguage must contain $K \cap \Sigma_{uo}^*$. This means, if $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$, then there certainly does not exist any finitely-weighted controllable and normal sublanguage. If $\omega_{G,f}(K \cap \Sigma_{uo}^*) < +\infty$, then clearly it is the minimum weight, and $K \cap \Sigma_{uo}^*$ is the supremal minimum-weighted controllable and normal sublanguage. In Step (7) by [8] we can derive that $L_m(S)$ is controllable with respect to $\hat{G}$, whose usage will be shown shortly. We have the following results.

Lemma 3.13. If (G, f) is zero-weighted-loop-free, then SPWSCP12 terminates. □

Proof: Since G and E are finite-state automata, K can be computed, so is $K \cap \Sigma_{uo}^*$. We now show that, determining whether $\omega_{G,f}(K \cap \Sigma_{uo}^*) < +\infty$ can be done in a finite number of steps. We only need to determine whether there exists a nonzero-weight loop in $\tilde{G}$ which recognizes $K \cap \Sigma_{uo}^*$. This can be done in polynomial time by expanding the

transition structure of $\tilde{G}$ as a tree until some node is revisited during the expansion. If $\tilde{G}$ does contain a loop with nonzero weight, then $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$. Otherwise, $\omega_{G,f}(K \cap \Sigma_{uo}^*) < +\infty$. Since SMC terminates, we know that SPWSCP12 terminates. ■

Theorem 3.14. Given a plant $(G, f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$, let K_{CN} be computed by SPWSCP12. Suppose $P_o : \Sigma^* \to \Sigma_o^*$ be a sup$\mathcal{CN}(G, E)$-observer. Then (1) $K_{CN} = \varnothing$ if and only if $\mathcal{WCN}(G, f, E) = \varnothing$; (2) When $K_{CN} \neq \varnothing$, we have $K_{CN} \in \mathcal{WCN}(G, f, E)$. □

Proof: (1) Let $K = \text{sup}\mathcal{CN}(G, E)$. Suppose $K_{CN} = \varnothing$. Then we have three cases to consider: Case 1.1: $K = \varnothing$, namely $L_m(G) \cap L_m(E)$ has no controllable and normal sublanguage with respect to G; Case 1.2: $K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$ and $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$; Case 1.3: the output of SMC is $v = +\infty$. For Case 1.1 and Case 1.2, clearly, $\mathcal{WCN}(G, f, E) = \varnothing$. For Case 1.3, by [8] we get that there is no controllable sublanguage of $P_o(K)$ with respect to $\hat{G}$ with a finite weight. Since P_o is K-observer, by Theorem 3.11 we know that, there is no sublanguage $K' \subseteq K$ with $P_o(K') \neq \{\epsilon\}$, which is controllable and normal with respect to G', and has a finite weight. Since $K = \text{sup}\mathcal{CN}(G, E)$, we get that, there is no sublanguage $K'' \subseteq L_m(G) \cap L_m(E)$ with $P_o(K'') \neq \{\epsilon\}$, which is controllable and normal with respect to G, and has a finite weight. Since SMC is called to compute v, we know that, there is no sublanguage $K''' \subseteq \Sigma_{uo}^* \cap K$ such that $\omega_{G,f}(K''') < +\infty$. Thus, $\mathcal{WCN}(G, f, E) = \varnothing$.
On the other hand, if $\mathcal{WCN}(G, f, E) = \varnothing$, then either $L_m(G) \cap L_m(E)$ has no controllable and normal sublanguage or there is no controllable and normal sublanguage with a finite weight. In the former case, clearly, $K_{CN} = \varnothing$ because $K = \varnothing$. In the latter case, there are two possibilities. One is that $K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$ but $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$. In this case, we have $K_{CN} = \varnothing$. The other possibility is that, $K \cap \Sigma_{uo}^* \notin \mathcal{CN}(G, E)$. Thus, for each sublanguage $K' \subseteq K$ with $K' \in \mathcal{CN}(G, E)$, we have $P_o(K') \neq \{\epsilon\}$. Since P_o is an K-observer, by Theorem 3.11 we know that, each controllable sublanguage of $P_o(K)$ has an infinite weight. Thus, the output of SMC is $v = +\infty$. In either case we have $K_{CN} = \varnothing$.
(2) Suppose $K_{CN} \neq \varnothing$. Then there are two cases. Case 1: $K_{CN} = K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$ and $\omega_{G,f}(K_{CN}) < +\infty$. Clearly, $K_{CN} \in \mathcal{WCN}(G, f, E)$. Case 2: $K \cap \Sigma_{uo}^* \notin \mathcal{CN}(G, E)$ and the output of SMC is finite. Then by a proof in [8] we can derive that $L_m(S) \in \mathcal{WC}(\hat{G}, \hat{f}, \hat{G})$. Since $K \cap \Sigma_{uo}^* \notin \mathcal{CN}(G, E)$, we know that, for all $K' \in \mathcal{CN}(G, E)$, we have $P_o(K') \neq \{\epsilon\}$. Thus, by Corollary 3.12 we can derive that $K_{CN} = P_o^{-1}(L_m(S))||L_m(G) \in \mathcal{WCN}(G, f, E)$. ■

Theorem 3.14 confirms that, when P_o is a natural observer, we can use SPWSCP12 to solve WSCP1 and WSCP2. Compared with PWSCP12, it is clear that SPWSCP12 is more computationally efficient. In fact, when P_o is a natural observer, the complexity of SPWSCP12 is polynomial-time because computing sup$\mathcal{CN}(G, E)$ [13], and constructing $(\hat{G}, \hat{f})$ [12] and S (and subsequently K_{CN}) can be done in polynomial time, and SMC terminates in polynomial time [8]. Next, we provide an algorithm to solve WSCP3.

Procedure for Solving WSCP3 (in the special setting) (SPWSCP3):

1. Input: a plant $(G = (X, \Sigma, \xi, x_0, X_m), f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$
2. Compute $K = \text{sup}\mathcal{CN}(G, E)$.
3. If $K = \varnothing$ then set $K_{CN} = \varnothing$ and go to step (9).

4. If $K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$, then set $K_{CN} = K \cap \Sigma_{uo}^*$ and go to step (9), if $\omega_{G,f}(K \cap \Sigma_{uo}^*) < +\infty$; or set $K_{CN} = \varnothing$ and go to step (9), if $\omega_{G,f}(K \cap \Sigma_{uo}^*) = +\infty$.

5. Let G' recognizes K and is homomorphic to G, and f' is induced from (G, f).

6. Construct $(\hat{G}, \hat{f}) \in \Phi(\Sigma_o)$, which is the over-weighted observable behavior of (G', f').

7. Apply SMC on $(\hat{G}, \hat{f})$. If the output $v = +\infty$, then set $K_{CN} := \varnothing$ and go to step (9). Otherwise, let $\hat{G} = (Z, \Sigma_o, \delta, z_0, Z_m)$ and the termination condition of SMC holds at r. Let S' be a recognizer of the sublanguage
$$\{s \in K | \theta_{G,f}(x_0, s) \leq v \wedge (\forall s' \leq s)\, \kappa_r(\delta(z_0, P_o(s'))) < +\infty\}$$
Compute
$$\hat{K} = \sup\mathcal{CN}(G, S')$$

8. Set $\hat{K}_0 := \hat{K}$ and iterates on $r = 0, 1, \cdots$, as follows:
   (a) Search a set $\psi(\hat{K}_r) := \{s \in \hat{K}_r | \theta_{G,f}(x_0, s) = \omega_{G,f}(\hat{K}_r)\}$.
   (b) Compute the largest $\hat{K}_{r+1} \subseteq \hat{K}_r - \psi(\hat{K}_r)$ that is controllable and normal w.r.t. G.
   (c) If $\hat{K}_{r+1} = \varnothing$ then set $K_{CN} := \hat{K}_r$ and go to step (9). Otherwise, continue on $r + 1$.

9. Output: K_{CN} □

What SPWSCP3 does is to fist run SPWSCP12 but output v and $\hat{G}$, then based on v and $\hat{G}$ to construct a sublanguage $\hat{K}$. It can be shown that the supremal controllable and normal sublanguage of (G, f) with respect to E is contained in $\hat{K}$. Step (8) is used to find such a supremal solution by continuously taking out strings with the maximum weight until no such strings can be taken out without nullifying the chance of finding a controllable and normal sublanguage. We have the following results.

Lemma 3.15. If (G, f) is zero-weighted-loop-free, then SPWSCP3 terminates. □

Proof: By Lemma 3.13 we only need to show that Step (8) terminates. Since v is finite, and (G, f) is zero-weighted-loop-free, we know that $\hat{K}_r$ is finite. Thus, the set $\psi(\hat{K}_r)$ can be computed, e.g. by simply enumerating each string in $\hat{K}_r$ and determining its weight. Thus, Step (8) terminates, which means SPWSCP3 terminates. ■

Theorem 3.16. Given a plant $(G, f) \in \Phi(\Sigma)$ and a requirement $E \in \phi(\Sigma)$, let K_{CN} be computed by SMCN. Suppose $P_o : \Sigma^* \to \Sigma_o^*$ is a $\sup\mathcal{CN}(G, E)$-observer. When $K_{CN} \neq \varnothing$, we have $K_{CN} = \sup\Xi(G, f, E)$. □

Proof: Suppose $K_{CN} \neq \varnothing$. There are two cases to consider. Case 1: $K_{CN} = K \cap \Sigma_{uo}^* \in \mathcal{CN}(G, E)$ and $\omega_{G,f}(K_{CN}) < +\infty$. Clearly, $K_{CN} = \sup\Xi WCN(G, f, E)$. Case 2: $K \cap \Sigma_{uo}^* \notin \mathcal{CN}(G, E)$. Then for all $K' \in \mathcal{CN}(G, E)$, we have $P_o(K') \neq \{\epsilon\}$. By Theorem 3.11 we can derive that, $\hat{K}_0 = \hat{K} \in \mathcal{WCN}(G, f, E)$, and $\sup\Xi(G, f, E) \subseteq \hat{K}_0$. By an argument similar to the one in the proof of Theorem 3.7, we can derive that $\sup\Xi(G, f, E) = \hat{K}_r = K_{CN}$. ■

Similar to PWSCP3, in Step (8) of SPWSCP3 the construction of S' is computationally intensive, making us wonder whether there exists a more computationally efficient solution to that. Unfortunately, it can be shown that, even when P_o is a natural observer, the complexity of solving WSCP3 is still NP-hard. We now use an example to illustrate SPWSCP12 and SPWSCP3.

Figure 5 depicts a simple manufacturing system, which consists of one input unit IU,

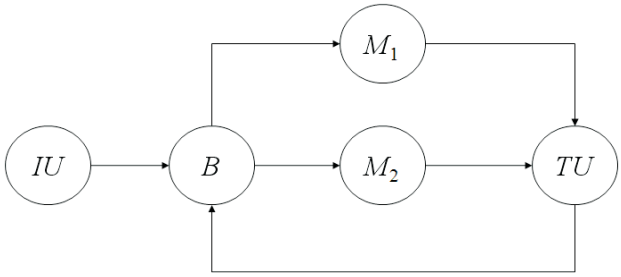

Figure 5: Example 4: A simple manufacturing system

two machines M_1 and M_2, one test unit TU, and one buffer B. Work pieces are add to the system through IU, which are stored in B. One of machines picks a work piece and processes it. After that, the machine sends the processed work piece to TU. If the quality of the work piece is good, then TU sends it out. Otherwise, TU sends it back to B for rework. Work pieces are grouped in batches. Each batch consists of a fixed number of work pieces. The system processes a batch, then is reset to its initial state before it processes the next batch. To simplify the example for the sake of illustration, we make the following assumptions: (1) the capacity of B is 1; (2) each batch consists of 2 work pieces; (3) each work piece takes at most one time rework before it is sent out of the system by TU. The component models are shown in Figure 6. The plant model is

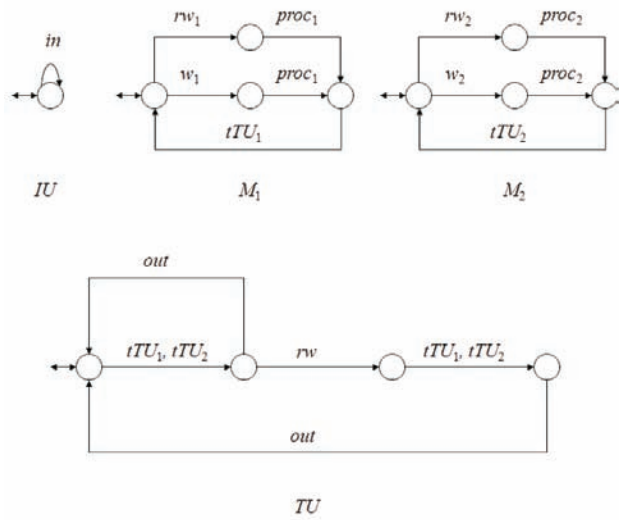

Figure 6: Example 4: component models

$$G := IU||M_1||M_2||TU$$

where $\Sigma_c = \{in, rw, rw_1, rw_2, w_1, w_2\}$ and $\Sigma_o = \Sigma_c$, i.e., all and only controllable events are observable. To simplify our analysis, we set all weights to be 1, except for weights for rw_1 and w_2, which are 2. In other words, M_1 is more expensive for rework, and M_2 is more expensive for normal processing. The requirements are shown in Figure 7. E_1 specifies that each batch consists of 2 work pieces, and E_2 specifies that, the work

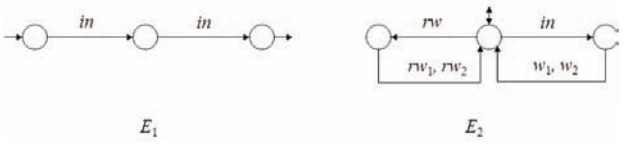

Figure 7: Example 4: requirement models

piece from IU is for normal processing and the piece from TU is for rework. By using some software tool we can check that P_o is a sup$\mathcal{CN}(G, E)$-observer. Thus, we can apply SPWSCP12 and SPWSCP3.

After running SPWSCP12 we get that the output of SMC is 26, and the final solution K_{CN} is depicted in Figure 8. The weight is 18. We can see that, for each work piece the normal processing is always done by M_1 and the rework is always done in M_2. This is because the normal processing in M_1 is "cheaper" than that in M_2, and the opposite holds for the rework. We now run SPWSCP3. The minimum weight is also 18 and the corresponding supremal minimum-weighted controllable and normal sublanguage is depicted in Figure 9. From the result we can see that, the result of SPWSCP12 is a subset of the result of SPWSCP3. This is because the latter is supremal, which contains all strings whose weights are no more than 18, e.g., if one work piece does not need rework, then the second work piece can use either machine for rework because the overall cost is certainly no more than the "worst-case" cost 18.

4 Conclusions

In this paper we have first presented three finitely-weighted supervisory control problems, i.e., WSCP1, WSCP2 and WSCP3, then provided concrete algorithms to solve them. It turns out that, it is always possible to decide whether there exists a finitely-weighted controllable and normal sublanguage of a weighted plant $(G, f) \in \Phi(\Sigma)$ with respect to an unweighted requirement $E \in \phi(\Sigma)$. If P_o is simply a natural projection, then computing a finitely-weighted controllable and normal sublanguage of (G, f) with respect to E can be done in exponential time by PWSCP12 because the complexity of projection is exponential time. If P_o is a natural observer, then computing a finitely-weighted controllable and normal sublanguage can be done in polynomial time by SPWCP12. Nevertheless, regardless of P_o being a natural observer, computing a minimum-weighted controllable and normal sublanguage is always NP-hard. Owing to the limited space, its proof is provided in a companion paper. In this paper we impose an assumption that the plant model is marking deadlock. We are still investigating whether those problems are solvable after dropping that assumption, and how to solve them efficiently.

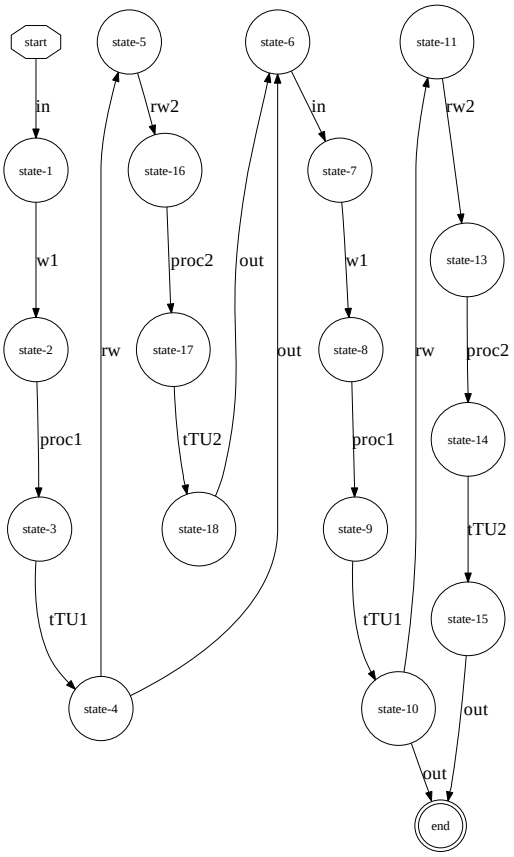

Figure 8: Example 4: a minimum-weighted controllable and normal sublanguage

The supervisory control problems presented in this paper are formulated in a centralized manner, namely there is one plant plus one specification. In reality, centralized synthesis may result in high computational complexity. Thus, it is of our primary interest to investigate whether similar approaches can be applied to a hierarchical and distributed system, which will be addressed in our future papers.

Acknowledgement: We would like to thank Dr. Albert T. Hofkamp of the Systems Engineering Group at Eindhoven University of Technology for coding all algorithms mentioned in this paper. We have used his code to generate the solution of Example 4 in Section III.

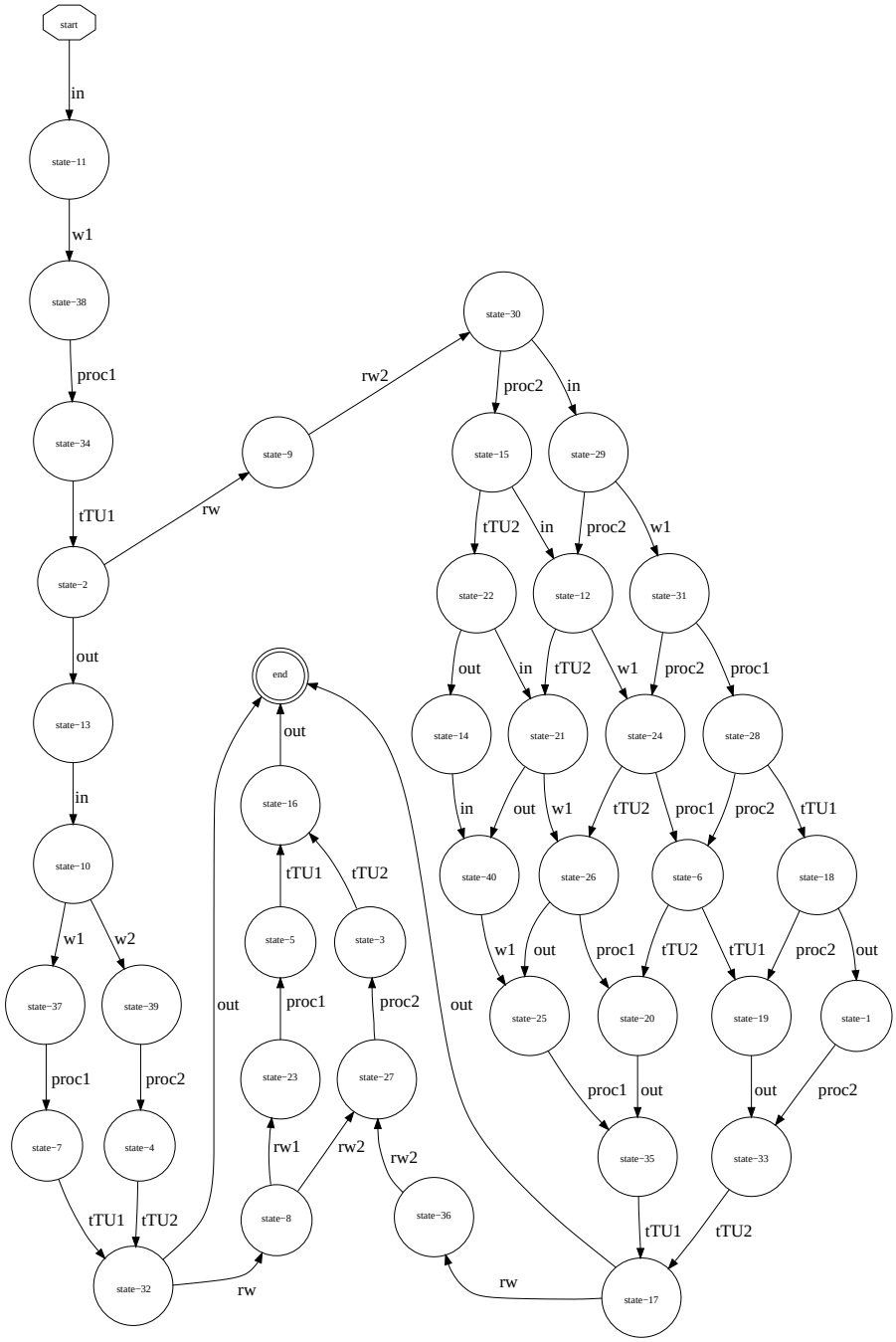

Figure 9: Example 4: supremal minimum-weighted controllable and normal sublanguage

Bibliography

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

[2] P.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event systems. *SIAM J. Control and Optimization*, 25(1):206–230, 1987.

[3] W.M. Wonham and P.J. Ramadge. On the supremal controllable sublanguage of a given language. *SIAM J. Control and Optimization*, 25(3):637–659, 1987.

[4] F. Lin and W. M. Wonham. On observability of discrete-event systems. *Information Sciences*, vol. 44, no. 3, pp. 173-198, 1988.

[5] O. Maler, A. Pnueli and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proc. 12th Symposium on Theoretical Aspects of Computer Science (STACS95), volume 900 of LNCS*, pages 229-242, 1995.

[6] E. Asarin, O. Maler, A. Pnueli and J. Sifakis. Controller synthesis for timed automata. In *Proc. IFAC Symposium on System Structure and Control*, pages 469-474, 1998.

[7] E. Asarin and O. Maler. As soon as possible: time optimal control for timed automata. In *Proc. 2nd International Workshop on Hybrid Systems: Computation and Control (HSCC99), volume 1569 of LNCS*, pages 19-30, 1999.

[8] Y. Brave and M. Heymann. On optimal attraction of discrete-event processes. *International Journal of Information Sciences*, 67(3):245-276, 1993.

[9] K. Passino and P. Antsaklis. On the optimal control of discrete event systems. In *Proc. 28th IEEE Decision and Control Conference (CDC89)*, pages 2713-2718, 1989.

[10] R. Sengupta and S. Lafortune. An optimal control theory for discrete event systems. *SIAM J. Control and Optimization*, 36(2):488-541, 1998.

[11] H. Marchand, O. Boivineau and S. Lafortune. On optimal control of a class of partially observed discrete event systems. *Automatica*, 38(11): 1935-1943, 2002.

[12] K.C. Wong and W.M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 6(3):241–273, 1996.

[13] W. M. Wonham (2007). *Supervisory Control of Discrete-Event Systems.* Systems Control Group, Dept. of ECE, University of Toronto. URL: www.control.utoronto.ca/DES.

[14] J.G. Thistle and W.M. Wonham. Supervision of infinite behavior of discrete-Event Systems. *SIAM Journal on Control and Optimization*, 32(4):1098-1113, 1994.

[15] J.G. Thistle and H.M. Lamouchi. Effective control synthesis for partially observed discrete-event systems. *SIAM J. Control Optim*, 48(3):1858-1887, 2009.

[16] K.C. Wong. On the complexity of projections of discrete-event systems. In *Proc. the 4th international workshop on discrete event systems (WOODES98)*, pages 201206, 1998.

[17] E.Grädel, W.Thomas and T.Wilke. *Automata, Logic and Infinite Games: A Guide to Current Research*. LNCS volumn 2500. Springer-Verlag, 2002.

[18] S. Safra. On the complexity of omega-automata. In *Proc. 29th Annual Symposium on Foundations of Computer Science*, pages 319-327, 1988.

[19] S. Tasiran, R. Hojati and R.K. Brayton. Language containment using non-deterministic omega-automata. In *Proc. 8th CHARME*, pages 261277, 1995.

[20] C.S. Althoff, W. Thomas and N. Wallmeier. Observations on determinization of büchi automata. *Theoretical Computer Science*, 363(2): 224-233, 2006.

[21] M. Mohri. Generic epsilon-removal algorithm for weighted automata. *Lecture Notes in Computer Science*, volumn 2088/2001, pages 230-242, 2001.

[22] J. Huang and R. Kumar. Optimal nonblocking directed control of discrete event systems. *IEEE Trans. Autom. Control*, 53(7):1592–1603, 2008.