A Generalized Theory on Supervisor Reduction

Rong Su and W. Murray Wonham

Abstract— **To make a supervisor comprehensible to a designer has been a long-standing goal in the supervisory control community. One strategy is to reduce the size of a supervisor to generate a control equivalent version, whose size is optimistically much smaller than the original one so that a user or control designer can easily check whether a designed controller fulfils its objectives and requirements. After the first journal paper on this topic appeared in 1986 by Vaz and Wonham, which relied on the concept of control covers, Su and Wonham proposed in 2004 to use control congruences to ensure computational viability. This work was later adopted in supervisor localization theory, which aims for a control equivalent distributed implementation of a given centralized supervisor. Despite these publications some fundamental questions, which might have been addressed in the first place, have not yet been answered, namely what information is critical to ensure control equivalence, what information is responsible for size reduction, and whether partial observation makes the problem essentially different. In this paper we address these questions by showing that there exists a unified supervisor reduction theory, which is applicable to all feasible supervisors regardless of whether they are under full observation or partial observation. Our theory proposes a preorder (called *leanness*) over all control equivalent feasible supervisors based on their enabling, disabling and marking information such that, if a supervisor S_1 is leaner than another supervisor S_2, then the size of the minimal control cover defined over the state set of S_1 is no bigger than that of S_2.**

I. Introduction

In supervisory control theory (SCT) [8] [14], the control problem associated with a discrete-event system (DES) is to enforce controllable and nonblocking behavior of the plant that is admissible by the specification. When applying SCT to a real application, there are two basic questions that require a user to answer, that is, are we doing the right thing, and are we doing things in the right way. The first question is about the correctness of the plant and requirement models. The second is about correctness of supervisor synthesis, which, if computational complexity is not a concern, has been adequately answered by SCT researchers. When computational complexity is indeed a concern, several efficient synthesis approaches have been proposed in the literature, e.g., [5] [9] [10] [15], which can ensure correct behaviours of the closed-loop system with low computational complexity. The first question, on the other hand, has been a long-standing hurdle to SCT being adopted by industry because so far there is no efficient way to identify potential errors in plant models or requirement models. The current practice is to synthesize a supervisor based on a given plant model and requirements. An empty supervisor usually indicates a fault either in the model or in the requirements; this should prompt the system designer to undertake model or requirement updates. The current SCT and its relevant tools can assist the designer to quickly locate the problems in the model that lead to emptiness of the supervisor. The real challenge is how to determine whether the plant model and the requirements are correct, when the supervisor synthesis returns a non-empty supervisor. In this case it usually requires not only syntactic correctness but also semantic correctness, i.e., the designer has to understand the true meaning and impact of every transition in the synthesized supervisor. Thus, to make a supervisor small enough for a designer to understand its function becomes important.

A supervisor carries two types of information: the key information at each state for event enabling/disabling and marking, and the information that tracks the evolution of the plant. The latter may contain some redundancy because the plant itself also carries such evolution information. In principle, it is possible to remove redundant transitional information from the supervisor, which will not interfere with the first kind of information, i.e., a reduced supervisor can still ensure the same control capability as that of the original supervisor. This is the key idea used in Vaz and Wonham's paper on supervisor reduction [12], which relies on the concept of *control cover*. A control cover is a collection of subsets of states in a supervisor, in which the states of each subset are "control consistent" with respect to event enabling/disabling and state marking; the exact meaning will be explained later. The authors proved two reduction theorems, and proposed a corresponding (exponential time) reduction algorithm. To overcome the computational complexity involved in supervisor reduction, Su and Wonham made a significant extension in [11] by first relaxing the concept of control cover, then providing a polynomial-time reduction algorithm based on a special type of cover called *control congruence*, and finally showing that the minimal supervisor problem (MSP) of computing a supervisor with minimal state size is NP-hard. A polynomial-time lower bound estimation algorithm provided in [11] has indicated that in many instances minimal supervisors can be achieved in polynomial time by using control congruence. Since then, this reduction algorithm has been used with gratifying results. One major application of supervisor reduction is in

Rong Su is affiliated with the School of Electrical and Electronic Engineering at Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798. Email: rsu@ntu.edu.sg. W. M. Wonham is affiliated with the Edward S. Rogers Sr. Department of Electrical and Computer Engineering at University of Toronto, 10 King's College Road, Toronto, Ontario, M5S 3G4, Canada. Email: wonham@control.utoronto.ca. The supports from Singapore Ministry of Education Tier 1 Academic Research Grant RG84/13 (2013-T1-002-177), National Research Foundation of Singapore DELTA-NTU CORP LAB-SMA, and from the Natural Sciences and Engineering Research Council (NSERC) of Canada, Grant DG7399, are gratefully acknowledged.

supervisor localization [1], which aims to create a control-equivalent distributed implementation of a given centralized supervisor.

The supervisor reduction theory proposed in [11] rests on two basic assumptions: (1) only full observation is considered; (2) the supervisor under consideration represents a sublanguage of the plant, which can be easily satisfied by applying supremal synthesis. Since then, many questions have been raised by users. For example, can we apply supervisor reduction to partially reduced supervisors (which may not necessarily represent sublanguages of a given plant) and can we apply supervisor reduction in cases with partial observation? Some results have been reported in the literature about the second question, see e.g., [3]. The main objective of supervisor reduction is to ensure control equivalence between the original supervisor and a reduced supervisor. The fundamental questions are (1) **Q1:** what information ensures control equivalence, even under partial observation, and (2) **Q2:** what information determines the state size of a reduced supervisor, which is the main performance index of supervisor reduction. Ever since [12] [11], these questions are still open. In this paper we aim to provide an answer. We first propose a generalized supervisor reduction theory which is applicable to all *feasible* supervisors, regardless of whether they are subject to full observation or partial observation; here a feasible supervisor does not disable uncontrollable events and always issues the same control command after strings that are not distinguishable based on observations. In the case of partial observation, a supervisor does not in general represent a sublanguage of the plant. We show that for each feasible supervisor S of a plant G, there exists a feasible supervisor **SUPER** derivable from the linguistic definition of *uncertainty subset construction* [13]). **SUPER** has the "universal" property that any feasible supervisor that is control equivalent to S with respect to G, and non-redundant with respect to S (i.e. without superfluous transitions), can be projected from **SUPER** via a suitable control cover on its state space, namely is a "quotient" of **SUPER** with respect to this cover. This result will answer our first question **Q1**. After that, we define a preorder $\preceq$ (referred to as "leanness") on feasible supervisors by using key information about event enabling/disabling and state marking such that for any two control equivalent supervisors S_1 and S_2 with respect to G, if S_1 is leaner than S_2, i.e., $S_1 \preceq S_2$, then the minimal reduced supervisor induced by a minimal control cover on S_1 is no bigger than the one induced by a minimal control cover on S_2. This result provides an answer to the second question **Q2**. As a direct consequence of this result, as long as control equivalence holds, a feasible supervisor under full observation always results in a reduced supervisor no bigger than the one induced from a supervisor under partial observation. Our theory is independent of the specific way of achieving observability, for instance via the property of normality [6] or of relative observability [2], or by direct search [6] - the effect of such a choice is lumped into the property of control feasibility, which states that a feasible supervisor must apply the same control law to all transitional sequences which cannot be distinguished based on observations.

The remainder of the paper is organized as follows. In Section II, we provide preliminaries on supervisor reduction. In Section III we discuss critical information for ensuring control equivalence. Then in Section IV we address information that determines reduction efficiency. We draw conclusions in Section V. Owing to limited space, all proofs are omitted in the paper, which can be retrieved at www.ntu.edu.sg/home/rsu/Publications.htm.

II. Preliminaries on supervisor reduction

Given an arbitrary finite alphabet Σ, let Σ^* be the free monoid on Σ whose elements are all the finite strings of zero or more elements from Σ, with the empty string ϵ being the identity element and string concatenation being the monoid operation. 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'. Any subset $L \subseteq \Sigma^*$ is called a *language*. The *prefix closure* of L is defined as $\overline{L} = \{s \in \Sigma^* | (\exists t \in L)\, s \leq t\} \subseteq \Sigma^*$. Given two languages $L, L' \subseteq \Sigma^*$, let $LL' := \{ss' \in \Sigma^* | s \in L \wedge s' \in L'\}$ denote their concatenation. 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

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

where $2^{\Sigma'^*}$ and 2^{Σ^*} denote the power sets of Σ'^* and Σ^*, respectively. When L is a singleton, say $L = \{s\}$, we will use $P^{-1}(s)$ to denote $P^{-1}(\{s\})$ for simplicity throughout the paper.

A plant is modelled as a *deterministic finite-state automaton*, $G = (X, \Sigma, \xi, x_0, X_m)$, where X stands 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 $X_m \subseteq X$ for the marker state set. Here we adopt the notations in [13] and write $\xi(x, \sigma)!$ to denote that the transition $\xi(x, \sigma)$ is defined. The domain of ξ can be extended to $X \times \Sigma^*$, where $\xi(x, \epsilon) = x$ for all $x \in X$, and $\xi(x, s\sigma) := \xi(\xi(x, s), \sigma)$. The *closed* behavior of G is defined as $L(G) := \{s \in \Sigma^* | \xi(x_0, s)!\}$, and the *marked* behavior of G is $L_m(G) := \{s \in L(G) | \xi(x_0, s) \in X_m\}$. G is *nonblocking* if $\overline{L_m(G)} = L(G)$. We say G is *reachable* if for each $x \in X$ there exists $s \in L(G)$ such that $\xi(x_0, s) = x$. From now on we consider only reachable automata. We denote by $|X|$ the size of the state set X. In some circumstances, when the state set is not explicitly mentioned, we also write $|G|$ for the size of an automaton, namely the size of its state set. Given two finite-state automata $G_i = (X_i, \Sigma, \xi_i, x_{i,0}, X_{i,m})$ $(i = 1, 2)$, the *meet* of G_1 and G_2, denoted as $G_1 \wedge G_2$, is

a (reachable) finite-state automaton such that $L(G_1 \wedge G_2) = L(G_1) \cap L(G_2)$ and $L_m(G_1 \wedge G_2) = L_m(G_1) \cap L_m(G_2)$.

Let $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$, where Σ_c, Σ_{uc} denote respectively the controllable and uncontrollable subalphabets, always disjoint, and similarly Σ_o, Σ_{uo} denote the observable and unobservable subalphabets, of the full alphabet Σ. A *(feasible) supervisor of* $G = (X, \Sigma, \xi, x_0, X_m)$ *under* $P_o : \Sigma^* \to \Sigma_o^*$ is a finite state automaton $S = (Z, \Sigma, \delta, z_0, Z_m)$ such that

- [*control existence*] $(\forall s \in L(G \wedge S))(\forall \sigma \in \Sigma_{uc})\, s\sigma \in L(G) \Rightarrow s\sigma \in L(S)$,
- [*control feasibility*] $(\forall s, s' \in L(S))\, P_o(s) = P_o(s') \Rightarrow \delta(z_0, s) = \delta(z_0, s')$.
- [*marking feasibility*] $(\forall z \in Z_m)(\exists s \in L_m(G \wedge S))\, \delta(z_0, s) = z$.

The first property says that a supervisor can only disable controllable events, thus, all uncontrollable events allowed by the plant G must be allowed by the supervisor S. This property can be ensured by enforcing controllability [8] on the closed-loop system behaviors. The second property says that a supervisor will issue the same (enabling) control command to strings which are observationally identical under P_o, that is, the supervisor may change its control command only when an observable event is received, or equivalently, all unobservable events can only be selflooped at relevant states, and any transition between two different states must be observable. This property guarantees implementation feasibility of the supervisor, and can be ensured by enforcing *observability* [6] on the closed-loop system behaviors*. The last property states that any marker state in the supervisor S must be reachable by a string $s \in L_m(G \wedge S)$, namely there is no redundant marking information in S. The closed-loop behavior of the system is denoted by two languages: the closed behavior $L(G \wedge S) = L(G) \cap L(S)$ and the marked behavior $L_m(G \wedge S) = L_m(G) \cap L_m(S)$. A supervisor S satisfying the properties of control existence, control feasibility and marking feasibility can be computed by the following simple procedure.

Procedure of Synthesis of Feasible Supervisors

- Step 1: Compute, by any method, a controllable observable sublanguage $K \subseteq L_m(G)$. For instance, TCT [17] can compute controllable and normal (or relatively observable) sublanguages and SuSyNA [18] is able to compute controllable and normal sublanguages.
- Step 2: **Uncertainty Subset Construction** (Chapter 6, [13]). Let $\equiv_K$ be the *Nerode equivalence relation* for K, i.e., for all $s, s' \in \overline{K}$,

$$s \equiv_K s' \iff (\forall u \in \Sigma^*)[su \in K \iff s'u \in K].$$

Let $[s]_K := \{s' \in \overline{K} | s \equiv_K s'\}$ the equivalence class of $s \in \overline{K}$ in the quotient $\overline{K}/\equiv_K = \{[s]_K | s \in \overline{K}\}$. Let $\simeq$ be the equivalence relation on $\overline{K}$ defined by

$$(\forall s, s' \in \overline{K})\, s \simeq s' \iff \{[t]_K | t \in P_o^{-1}(P_o(s)) \cap \overline{K}\} = \{[t']_K | t' \in P_o^{-1}(P_o(s')) \cap \overline{K}\}.$$

Compute $S = (Z, \Sigma, \delta, z_0, Z_m)$, where $Z := \overline{K}/\simeq$, $Z_m := \{z \in Z | z \cap K \neq \varnothing\}$, $z_0 := [\epsilon]_\simeq$ and the transition function $\delta : Z \times \Sigma \to Z$ is induced in the natural way (see [13]), i.e.,

$$(\forall z, z' \in Z)(\forall \sigma \in \Sigma)\, [\delta(z, \sigma) = z' \iff (\exists s \in z, s' \in z')\, s\sigma = s'].$$

It is straightforward to check that S is feasible and qualified as a proper supervisor for G that synthesizes K; namely $K = L_m(G \wedge S)$ and $\overline{K} = L(G \wedge S)$. A similar procedure called P-supervisor Synthesis and Standard Realization in [4] can also achieve the same goal, which essentially relies on the automaton subset construction algorithm [16].

A deterministic supervisor in the Ramadge-Wonham paradigm, whose control command at each state is determined by the set of enabled events at that state, must satisfy the control existence property and the control feasibility property. The marking feasibility property on the other hand does not necessarily hold in a deterministic supervisor, which could create some restriction on the applicability of the reduction theory in this paper. Nevertheless, for any supervisor generated by existing synthesis tools such as TCT, SuSyNA and Supremica, the marking feasibility property always holds. Thus, the restriction imposed by the last property is rather mild from an application point of view.

As a simple running example consider the single-tank system of Figure 1, consisting of a water supply source

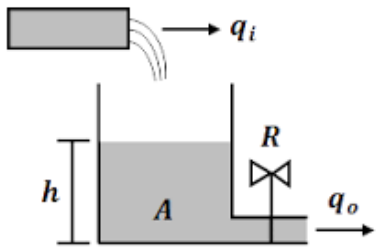

Fig. 1. Example 1: A single-tank system

whose supply rate q_i is a constant (for simplicity), a tank, and a control valve at the bottom of the tank controlling the outgoing flow rate q_o, whose value depends on the valve opening and the water level h. We assume that the valve can only be fully open or fully closed, and in case of a full opening, the water level h can only go down. The water level h can be measured; its value change triggers one of the following predefined events: h=L, h=M, h=H, and h=EH, which denote that the water level is changed to *low*, *medium*, *high*, *extremely high*, respectively. The plant model G of the system is depicted in Figure 2, where the alphabet Σ consists exactly of the events shown in the figure. The actions of opening the valve ($q_o = 1$) and closing the valve ($q_o = 0$) are controllable but unobservable, and all water level events are observable but uncontrollable. In the model a shaded double circle denotes a marker state, i.e., states 5 and 9 in Figure 2. Assume that we do not want the water level to become

*Various properties such as normality [6] or relative observability [2] can achieve observability

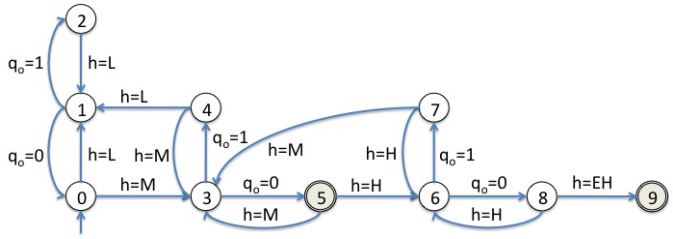

Fig. 2. Example 1: Automaton model of the plant G

extremely high, i.e., the event h=EH should not occur. To prevent state 9 from being reached, we bring in requirement E shown in Figure 3, whose alphabet is {h=L, h=M, h=H, h=EH}, but the event h=EH is prohibited from occurring. A controllable and observable sublanguage, i.e., a closed-loop behavior $K = L_m(G \wedge S)$, can be synthesized by using the standard Ramadge-Wonham supervisory control paradigm, which is also depicted in Figure 3. The corresponding feasi-

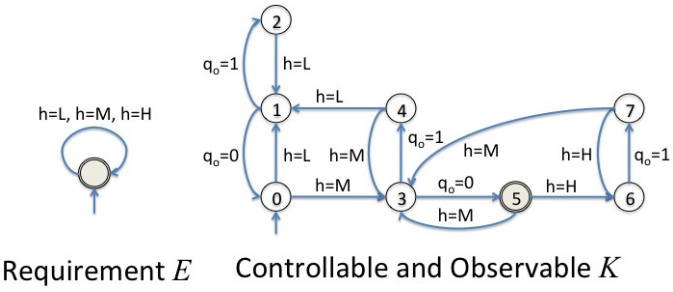

Fig. 3. Example 1: Automaton models of a requirement E (Left) and the controllable and observable sublanguage K (Right)

ble supervisor S via the uncertainty subset construction on K is depicted in Figure 4. We can see that in S all unobservable

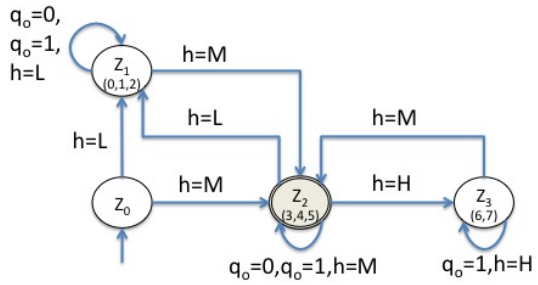

Fig. 4. Example 1: A feasible supervisor S

events are selflooped at some states, and transitions between different states are all labeled by observable events.

For a plant G, there may exist more than one supervisor that achieves a control objective, e.g., ensures that the closed-loop system behavior is contained in a predefined requirement language $E \subseteq \Sigma^*$. Two supervisors S_1 and S_2 of G are *control equivalent* [11] if $L(G \wedge S_1) = L(G \wedge S_2)$ and $L_m(G \wedge S_1) = L_m(G \wedge S_2)$. Let $\mathcal{F}(G, S)$ be the collection of all feasible supervisors of G under partial observation P_o, which are control equivalent to a given supervisor S. It is desirable to find one supervisor $S_* \in \mathcal{F}(G, S)$ such that for all $S' \in \mathcal{F}(G, S)$ we have $|S_*| \leq |S'|$, i.e., the supervisor S_* has the minimum number of states. It has been shown in [11] that, unfortunately, finding S_* based on the concept of *control covers* is NP-hard, even for a supervisor under full observation. Each control cover is a collection of subsets of states in S, in which the states of each subset are "control consistent", to be defined shortly. Thus, by grouping those compatible states of S together, we may get a new supervisor S' such that (1) S' is control equivalent to S; (2) $|S'| < |S|$. In the next two sections we will investigate which information is responsible for control consistency, and which for size reduction.

III. Information that ensures control equivalence

Given a plant $G = (X, \Sigma, \xi, x_0, X_m)$ and a feasible supervisor $S = (Z, \Sigma, \delta, z_0, Z_m)$, at each state $z \in Z$ there are four pieces of information shown below:

- Let $En_S : Z \to 2^\Sigma$ with
$$z \mapsto En_S(z) := \{\sigma \in \Sigma | \delta(z, \sigma)!\}$$
be the (S-)enabled event set at state $z \in Z$.
- Let $D_S : Z \to 2^\Sigma$ with
$$z \mapsto D_S(z) := \{\sigma \in \Sigma | \neg\delta(z, \sigma)! \wedge (\exists s \in L(G))\, s\sigma \in L(G) \wedge \delta(z_0, s) = z\}$$
be the (S-)disabled event set at state $z \in Z$.
- Let $M_S : Z \to \{true, false\}$ with
$$z \mapsto M_S(z) := true \text{ if } (\exists s \in L_m(G \wedge S))\, \delta(z_0, s) = z$$
be the S-marking indicator at state $z \in Z$.
- Let $T_S : Z \to \{true, false\}$ with
$$z \mapsto T_S(z) := true \text{ if } (\exists s \in L_m(G))\, \delta(z_0, s) = z$$
be the G-marking indicator at state $z \in Z$.

The $(S-)$enabled event sets can be easily obtained by simply checking the transition structure of S. To determine the other sets for each state $z \in Z$, we can first construct the meet $G \wedge S$, and then check each state pair (x, z) in the meet associated with the state $z \in Z$. Compared with relevant definitions in [11], the only change is made to the concept of S-marking indicator, which requires that $M_S(z) = true$ iff z is reachable by a string $s \in L_m(G \wedge S)$, instead of simply requiring $z \in Z_m$ as in [11], because here it is not necessary that $L_m(S) \subseteq L_m(G)$.

As an illustration, we revisit the supervisor S for the single-tank system depicted in Figure 4. By computing the meet $G \wedge S$ we obtain the transition structure recognizing K shown in the right-hand picture of Figure 3. From that structure we get the following:

- $En_S(z_0) = \{$h=L, h=M$\}$, $D_S(z_0) = \varnothing$, $M_S(z_0) = false$, $T_S(z_0) = false$,
- $En_S(z_1) = \{$q$_0$=0, q$_0$=1, h=L, h=M$\}$, $D_S(z_1) = \varnothing$, $M_S(z_1) = false$, $T_S(z_1) = false$,

- $En_S(z_2) = \{q_0=0,\ q_0=1,\ h=L,\ h=M,\ h=H\}$, $D_S(z_2) = \varnothing$, $M_S(z_2) = true$, $T_S(z_2) = true$,
- $En_S(z_3) = \{q_0=1,\ h=M,\ h=H\}$, $D_S(z_3) = \{q_0=0\}$, $M_S(z_3) = false$, $T_S(z_3) = false$.

Let $\mathcal{R} \subseteq Z \times Z$ be a binary relation, where $(z, z') \in \mathcal{R}$ iff the following two properties hold:

1) $En_S(z) \cap D_S(z') = En_S(z') \cap D_S(z) = \varnothing$,
2) $T_S(z) = T_S(z') \Rightarrow M_S(z) = M_S(z')$.

We call $\mathcal{R}$ the *binary compatibility relation* over Z. The first condition requires that no event enabled at one state can be disabled at the other state. The second condition requires that both states must have the same marking status, if they are reachable by strings from the marked behavior of G. Notice that $\mathcal{R}$ is reflexive and symmetric, but needn't be transitive, namely is a *tolerance* but not an equivalence relation. Any two states satisfying $\mathcal{R}$ may potentially be merged together, if their suffix behaviors are "compatible", which is precisely captured in the following concept. Let I be a finite index set. We assume that $m \notin I$.

Definition 1: A collection $\mathcal{C} = \{(Z_i, i) | Z_i \subseteq Z \wedge i \in I\}$ is a *control cover* on S if

1) $\cup_{i \in I} Z_i = Z$, $(\forall (Z_i, i), (Z_j, j) \in 2^Z \times I)\, i = j \Rightarrow Z_i = Z_j$,
2) $(\forall i \in I)\, Z_i \neq \varnothing \wedge (\forall z, z' \in Z_i)\, (z, z') \in \mathcal{R}$,
3) $(\forall i \in I)(\forall \sigma \in \Sigma)(\exists j \in I)[(\forall z \in Z_i)\delta(z, \sigma)! \Rightarrow \delta(z, \sigma) \in Z_j]$.

$\mathcal{C}$ is *minimal* if for all other control covers $\mathcal{C}'$ of Z we have $|\mathcal{C}| \leq |\mathcal{C}'|$. □

The definition of a control cover ensures that any two different elements in the cover must have distinct index values from I. But it is possible that $Z_i = Z_j$ when $i \neq j$. Because $\mathcal{C} := \{(\{z\}, z) | z \in Z\}$ is trivially a control cover, we know that S is non-empty if and only if there exists a non-empty control cover $\mathcal{C}$ of S. Given a control cover $\mathcal{C} = \{(Z_i, i) | Z_i \subseteq Z \wedge i \in I\}$ on S, we construct an induced supervisor $S_{\mathcal{C}} = (I, \Sigma, \kappa, i_0, I_m)$, where $i_0 \in I$ such that $z_0 \in Z_{i_0}$, $I_m := \{i \in I | Z_i \cap Z_m \neq \varnothing\}$, and $\kappa : I \times \Sigma \to I$ is the partial transition map such that for each $i \in I$ and $\sigma \in \Sigma$, $\kappa(i, \sigma) := j$ if j is chosen to satisfy the following property:

$$(\exists z \in Z_i)\delta(z, \sigma) \in Z_j \wedge [(\forall z' \in Z_i)\, \delta(z', \sigma)! \Rightarrow \delta(z', \sigma) \in Z_j];$$

otherwise, $\kappa(i, \sigma)$ is not defined. In general, there may exist more than one choice of j with the above property. An arbitrary selection among multiple choices will be adopted to ensure a deterministic transitional structure for $S_{\mathcal{C}}$.

Theorem 1: Given a feasible supervisor $S = (Z, \Sigma, \delta, z_0, Z_m)$ for a plant $G = (X, \Sigma, \xi, x_0, X_m)$, let $\mathcal{C} = \{(Z_i, i) | Z_i \subseteq Z \wedge i \in I\}$ be a control cover on S, and $S_{\mathcal{C}}$ be an induced supervisor from $\mathcal{C}$. Then $S_{\mathcal{C}}$ is a feasible supervisor, which is control equivalent to S.

Theorem 1 indicates that we can start with any given plant G and feasible supervisor S to generate another feasible supervisor S', which is control equivalent to S with respect to G, by applying the aforementioned construction induced by a properly chosen control cover on S. Of special interest is the fact that we do not need to know how S was obtained in the first place. Thus, we have a unified way of undertaking supervisor reduction regardless of whether S is under full observation or partial observation. As an illustration we revisit the single-tank system, whose feasible supervisor S is depicted in Figure 4. Based on the aforementioned analysis about those four sets, i.e., $En_S(z)$, $D_S(z)$, $M_S(z)$ and $T_S(z)$, we can check that the set $\mathcal{C} := \{(\{z_0, z_1, z_2\}, 1), (\{z_3\}, 2)\}$ is a control cover, where $I = \{1, 2\}$. The resulting induced supervisor $S_{\mathcal{C}}$ is depicted in Figure 5. We can easily check

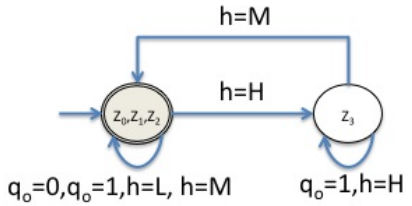

Fig. 5. Example 1: An induced supervisor $S_{\mathcal{C}}$

that $S_{\mathcal{C}}$ is control equivalent to S with respect to G. From $S_{\mathcal{C}}$ we can see that what S really does is preventing the valve from being closed when the water level is high, which matches our expectation perfectly.

Next, we will present a result similar to the Generalized Quotient Theorem in [11].

Definition 2: Given a plant G and a feasible supervisor S, let $S' = (Z', \delta', \Sigma, z_0', Z_m')$ be another feasible supervisor of G. Then S' is *non-redundant* with respect to S if the following hold:

1) $(\forall z \in Z')(\forall \sigma \in \Sigma)\delta'(z, \sigma)! \Rightarrow (\exists s \in L(G \wedge S))\, s\sigma \in L(G \wedge S) \wedge \delta'(z_0', s) = z$,
2) $(\forall z \in Z_m')(\exists s \in L_m(G \wedge S))\, \delta'(z_0', s) = z$. □

Definition 2 indicates that each transition in a non-redundant supervisor S' must belong to a string in the closed behavior $L(G \wedge S)$, and every marker state in S' must be reachable by a string in the marked behavior $L_m(G \wedge S)$. Both synthesis tools TCT and SuSyNA can be used to compute a non-redundant feasible supervisor.

Definition 3: Given automata $G_A = (X_A, \Sigma, \xi_A, x_{A,0}, X_{A,m})$ and $G_B = (X_B, \Sigma, \xi_B, x_{B,0}, X_{B,m})$, we say that G_B is the image of G_A under the DES-isomorphism $\theta : X_A \to X_B$ if

1) θ is bijective,
2) $\theta(x_{A,0}) = x_{B,0}$ and $\theta(X_{A,m}) = X_{B,m}$,
3) $(\forall x, x' \in X_A)(\forall \sigma \in \Sigma)\xi_A(x, \sigma) = x' \Rightarrow \xi_B(\theta(x), \sigma) = \theta(x')$,
4) $(\forall x \in X_B)(\forall \sigma \in \Sigma)\xi_B(x, \sigma)! \Rightarrow \xi_A(\theta^{-1}(\{x\}), \sigma)!$. □

Definition 3 states that an automaton G_A is DES-isomorphic to another automaton G_B if G_A and G_B are identical up to state relabeling, i.e., every state in G_A maps to a unique state in G_B, in particular, the initial state of G_A maps to the initial state of G_B, each marker state of G_A maps to a marker state of G_B, each marker state in G_B must have

one marker state in G_A as a pre-image, each transition in G_A corresponds to one transition in G_B, and each transition in G_B has one transition in G_A as the pre-image.

Given a plant G and a feasible supervisor S, by computing the meet of G and S, i.e., $G \wedge S$, we can obtain the closed-loop (closed and marked) behaviours. By applying the uncertainty subset construction shown in Procedure 1 to $G \wedge S$ with respect to $P_o : \Sigma^* \to \Sigma_o^*$, we can derive a feasible supervisor, say **SUPER**, which is control equivalent to S. The following main result shows that any non-redundant feasible supervisor, which is control equivalent to S with respect to G, can be constructed from **SUPER** by using a suitable control cover on S. It is an extension of the Generalized Quotient Theorem stated in [11] to more generally defined feasible supervisors, and clearly shows the "universality" of control covers.

Theorem 2: **[Extended Quotient Theorem]** Given a feasible supervisor S of a plant G, let **SUPER** be constructed as above. Then for any non-redundant feasible supervisor **SIMSUP** with respect to S, which is control equivalent to S with respect to G, there exists a control cover $\mathcal{C}$ on **SUPER** and an induced feasible supervisor $S_{\mathcal{C}}$ such that $S_{\mathcal{C}}$ is DES-isomorphic to **SIMSUP**.

Up to now we have developed a general theory on supervisor reduction, which unifies both the full observation case and the partial observation case. It is clear that the concrete way of ensuring observability in a feasible supervisor is not important in achieving control equivalence during supervisor reduction. Knowledge of the plant G and a feasible supervisor S will be sufficient for us to construct a feasible supervisor, which is control equivalent to S, and optimistically has a (significantly) smaller size.

IV. Information that determines reduction efficiency

Our case studies indicate that a supervisor with full observation usually allows a much higher reduction ratio than that allowed by a supervisor with partial observation. An interesting question is what causes such a discrepancy. In this section we propose an answer to this question, and explain the actual effects of partial observation on supervisor reduction.

Given a plant G and a feasible supervisor S, each feasible supervisor $S' \in \mathcal{F}(G, S)$ carries four pieces of critical information captured by $(En_{S'}, D_{S'}, M_{S'}, T_{S'})$. We define a preorder "$\preceq$" among elements of $\mathcal{F}(G, S)$[†], where for all $S_i = (Z_i, \Sigma, \delta_i, z_{i,0}, Z_{i,m}) \in \mathcal{F}(G, S)$ $(i = 1, 2)$, we say S_1 is *leaner than* S_2, denoted as $S_1 \preceq S_2$, if for all $s \in L(G \wedge S)$ we have

- $En_{S_1}(\delta_1(z_{1,0}, s)) \subseteq En_{S_2}(\delta_2(z_{2,0}, s))$ and $D_{S_1}(\delta_1(z_{1,0}, s)) \subseteq D_{S_2}(\delta_2(z_{2,0}, s))$,
- $M_{S_1}(\delta_1(z_{1,0}, s)) = true \Rightarrow M_{S_2}(\delta_2(z_{2,0}, s)) = true$,
- $T_{S_1}(\delta_1(z_{1,0}, s)) = true \Rightarrow T_{S_2}(\delta_2(z_{2,0}, s)) = true$.

In words, S_1 is leaner than S_2 if for each pair of states z_1 in S_1 and z_2 in S_2 reachable by the same string in $L(G \wedge S)$, the enabled and disabled event sets at z_1 are subsets of those at z_2; and, if the values of the S-marking indicator and the G-marking indicator at z_1 are both true, then those values at z_2 are also true. Informally speaking, each string $s \in L(G \wedge S)$ carries two types of information. Type 1: information associated with s only, i.e., the set of enabled events $\psi(s) := \{\sigma \in \Sigma | s\sigma \in L(G \wedge S)\}$ and the set of disabled events $\lambda(s) := \{\sigma \in \Sigma | s\sigma \in L(G) \wedge s\sigma \notin L(S)\}$. Type 2: information associated with all strings that reach the same state as that reached by s, e.g., z_1 in S_1 above, i.e., the set of enabled events at the state reached by s, e.g., $En_{S_1}(z_1)$, and the set of disabled events at the state reached by s, e.g., $D_{S_1}(z_1)$. It is clear that $\psi(s) \subseteq En_{S_1}(z_1)$ and $\lambda(s) \subseteq D_{S_1}(z_1)$. In other words, when the plant G executes the string s, the supervisor S_1 allows more events specified by $En_{S_1}(z_1) - \psi(s)$ and disables more events specified by $D_{S_1}(z_1) - \lambda(s)$, owing to the need for ensuring control feasibility. Such extra enabled/disabled events are clearly redundant for s. If $S_1 \preceq S_2$, we know that $En_{S_1}(z_1) - \psi(s) \subseteq En_{S_2}(z_2) - \psi(s)$, and $D_{S_1}(z_1) - \lambda(s) \subseteq D_{S_2}(z_2) - \lambda(s)$, that is, S_1 carries less redundant information (or equivalently, is leaner) than S_2 does. Although such redundant information does not affect control equivalence of relevant supervisors, it does affect the state-size reduction ratio, when we construct a reduced supervisor based on a control cover.

The example of Figure 6 illustrates preorder over control equivalent feasible supervisors. The alphabet of the plant G is $\Sigma = \{a, b, c, d_1, d_2, e\}$, $\Sigma_c = \{d_1, d_2\}$, and all events

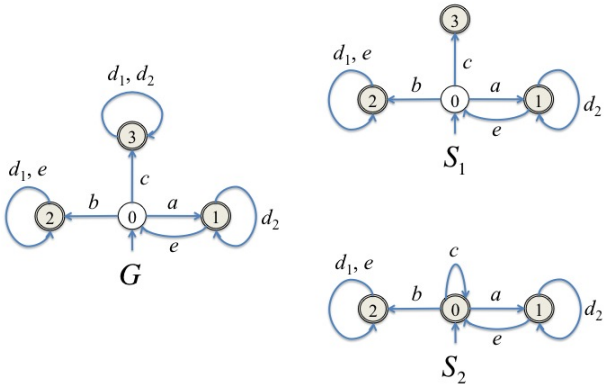

Fig. 6. Example 2: A plant G (left), supervisors S_1 (right top) and S_2 (right bottom)

are observable for the sake of simplicity. It is not difficult to check that S_1 and S_2 are control equivalent - they both disable events d_1 and d_2 after executing the event c. To check that S_1 is leaner than S_2, we only need to check those conditions for two strings $s = \epsilon$ and $s = c$ because for other strings in $L(G \wedge S)$, S_1 and S_2 are the same. For $s = \epsilon$, we have $z_1 = 0$ in S_1 and $z_2 = 0$ in S_2. Clearly, $En_{S_1}(z_1) = \{a, b, c\} = En_{S_1}(z_2)$, and $D_{S_1}(z_1) = \varnothing \subseteq \{d_1, d_2\} = D_{S_2}(z_2)$. In addition, we can check that $M_{S_1}(z_1) = false$ and $M_{S_2}(z_2) = true$, and $T_{S_1}(z_1) = false$ and $T_{S_2}(z_2) = true$. Thus, those conditions hold for $s = \epsilon$. For $s = c$ we have $z_1 = 3$ in S_1 and $z_2 = 0$ in S_2. Clearly, $En_{S_1}(z_1) = \varnothing \subseteq En_{S_1}(z_2) = \{a, b, c\}$, and

[†]A *preorder* is reflexive and transitive but not necessarily antisymmetric.

$D_{S_1}(z_1) = \{d_1, d_2\} = D_{S_2}(z_2)$. In addition, we can check that $M_{S_1}(z_1) = true = M_{S_2}(z_2)$, and $T_{S_1}(z_1) = true = T_{S_2}(z_2)$. Thus, we conclude that S_1 is leaner than S_2.

Theorem 3: Given a plant G and a feasible supervisor S, let **SUPER** be the same as that stated in Theorem 2. Then for all $S' \in \mathcal{F}(G, S)$, we have **SUPER** $\preceq S'$.

Theorem 3 indicates that for all feasible supervisors in $\mathcal{F}(G, S)$, **SUPER** has the leanest information which still ensures control equivalence. The interesting point is that for any feasible supervisor $S' \in \mathcal{F}(G, S)$, we can construct **SUPER** by applying the uncertainty subset construction on $G \wedge S'$, namely we can always obtain the leanest feasible supervisor, which is control equivalent to S with respect to G. Nevertheless, the size of **SUPER** could be large for a practical application. Thus, supervisor reduction may be directly applied to any feasible supervisor $S' \in \mathcal{F}(G, S)$. The following result indicates that the state size of a reduced supervisor solely depends on the leanness of the key information specified by those four functions - the leaner the information, the smaller the reduced state size.

Theorem 4: Given a plant G and a feasible supervisor S, let $S_1, S_2 \in \mathcal{F}(G, S)$ be non-redundant with respect to S, and assume that $S_1 \preceq S_2$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be minimal control covers of S_1 and S_2, respectively. Then $|\mathcal{C}_1| \leq |\mathcal{C}_2|$.

As an illustration, in Example 2 depicted in Figure 6 we know that $S_1 \preceq S_2$. We can easily compute $\hat{S}_1$ and $\hat{S}_2$, which are minimal feasible supervisors control equivalent to S_1 and S_2, respectively. The results are shown in Figure 7 below. To show that $\hat{S}_1$ is minimal, we notice that any

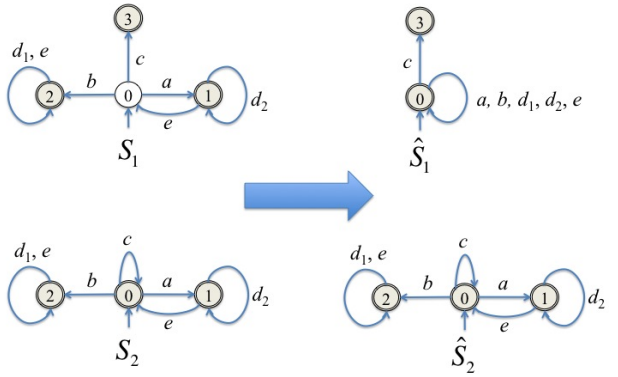

Fig. 7. Example 2: Reduced supervisors $\hat{S}_1$ (right top) and $\hat{S}_2$ (right bottom)

feasible supervisor, which is control equivalent to S_1, needs to have at least two different control patterns, i.e., to disable d_1 and d_2 together (at state 3 in G), and to disable neither of them (at the remaining states in G). Thus, it must have at least two states. Since $\hat{S}_1$ has precisely two states, it is a minimal supervisor. But it is not the unique one because we can check that the cover $\mathcal{C} = \{(\{0, 3\}, 1), (\{1, 2\}, 2)\}$ with $I = \{1, 2\}$ is also a control cover of S_1, whose induced supervisor is different from $\hat{S}_1$ and has two states. To see that $\hat{S}_2$ is minimal, with the same argument as before, we know that any feasible supervisor, which is control equivalent to S_2, must have at least two states. But we can check manually that none of the covers of S_2, whose size is 2, is a control cover of S_2. Thus, the size of any control cover of S_2 must be at least 3, which means S_2 is the smallest one, which cannot be reduced further. It is clear that $|\hat{S}_1| = 2 < |\hat{S}_2| = 3$, which matches the conclusion made in Theorem 4.

With Theorem 3 and Theorem 4 we are able to answer the question: among all feasible supervisors that are control equivalent, which one will lead to a minimal reduced supervisor, and for any two control equivalent feasible supervisors, which one will result in a smaller reduced supervisor.

V. Conclusions

We have developed a generalized supervisor reduction theory, which is applicable to all feasible supervisors, regardless of whether they are designed under the assumption of full or of partial observation. We have shown that the generalized quotient theorem in [11] for supervisors with full observation has a counterpart in the generalized reduction theory, which states that for each feasible supervisor S of a plant G, there exists a feasible supervisor **SUPER** derivable from the uncertainty subset construction on $G \wedge S$ such that all feasible supervisors that are control equivalent to S with respect to G and non-redundant with respect to S can be derived via the quotient construction based on a properly chosen control cover on **SUPER**. In addition, we have provided a specific way of ordering those feasible supervisors by using the key information described in those four functions such that for any two control equivalent supervisors S_1 and S_2 with respect to (G, S), if S_1 is leaner than S_2, i.e., $S_1 \preceq S_2$, then the minimal reduced supervisor induced from S_1 is no bigger than the one induced from S_2.

References

[1] K. Cai and W.M. Wonham. Supervisor localization: a top-down approach to distributed control of discrete-event systems. *IEEE Trans. Automatic Control*, 55(3):605-618, 2010.

[2] K.Cai, R. Zhang and W. M. Wonham. Relative observability of discrete-event systems and its supremal sublanguages. *IEEE Transactions on Automatic Control*, 60(3):659-670, 2013.

[3] R. Zhang and K.Cai. On supervisor localization based distributed control of discrete-event systems under partial observation. In *Proc. 2016 American Control Conference*, pp. 764-767, Boston, 2016.

[4] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems* (2nd Ed.), Springer, 2008.

[5] L. Feng and W.M. Wonham. Supervisory control architecture for discrete-event systems. *IEEE Trans. Automatic Control*, 53(6):1449-1461, 2008.

[6] F. Lin and W. M. Wonham.On observability of discrete-event systems. *Information Sciences*, 44(3):173-198, 1988.

[7] C. H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.

[8] 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.

[9] R. Su, J.H. van Schuppen and J.E. Rooda. Aggregative synthesis of distributed supervisors based on automaton abstraction. *IEEE Trans. Automatic Control*, 55(7):1627-1640, 2010.

[10] R. Su, J.H. van Schuppen and J.E. Rooda. Maximally permissive coordinated distributed supervisory control of nondeterministic discrete-event systems. *Automatica*, 48(7):1237-1247, 2012.

[11] R. Su and W. M. Wonham. Supervisor reduction for discrete-event systems. *Journal of Discrete Event Dynamic Systems*, 14(1):31-53, 2004.

[12] A. F. Vaz and W. M. Wonham. On supervisor reduction in discrete-event systems. *International Journal of Control*, 44(2):475-491, 1986.
[13] W. M. Wonham. *Supervisory Control of Discrete-Event Systems*. Systems Control Group, Dept. of ECE, University of Toronto. URL: www.control.utoronto.ca/DES, 2016.
[14] 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.
[15] S. Mohajerani, R. Malik and M. Fabian. A framework for compositional synthesis of modular nonblocking supervisors. *IEEE Transactions on Automatic Control*, 59 (1):150-162, 2014.
[16] G. Van Noord. Treatment of epsilon moves in subset construction. *Computational Linguistics*, 26(1):61-76, 2000.
[17] TCT: A Computation Tool for Supervisory Control Synthesis. http://www.control.utoronto.ca/DES/Research.html.
[18] SuSyNA: Supervisor Synthesis for Nondeterministic Automata. http://www.ntu.edu.sg/home/rsu/Downloads.htm.

APPENDIX

Proof of Theorem 1: 1. We first claim that $L_m(G \wedge S) \subseteq L_m(G \wedge S_{\mathcal{C}})$. Let $s \in L_m(G \wedge S)$. If $s = \epsilon$, then $z_0 \in Z_m$. Since $z_0 \in Z_{i_0}$, we have $Z_{i_0} \cap Z_m \neq \varnothing$. Therefore $i_0 \in I_m$, namely $\epsilon \in L_m(S_{\mathcal{C}})$. Let $s = \sigma_0 \cdots \sigma_k$ $(k > 0)$. Because

$$\delta(z_0, \sigma_0)!,\ \delta(z_0, \sigma_0\sigma_1)!, \cdots, \delta(z_0, \sigma_0\sigma_1 \cdots \sigma_k)!,$$

we have

$$\delta(z_0, \sigma_0)! \text{ and } \delta(z_j, \sigma_j)! \text{ with } z_{j+1} = \delta(z_0, \sigma_0 \cdots \sigma_j),\ j = 1, \cdots, k$$

Since $\{Z_i | i \in I\}$ is a control cover on Z, by Definition 1 and the definition of κ we have

$$(\forall j : 0 \leq j \leq k)(\exists i_j, i_{j+1} \in I) z_j \in Z_{i_j} \wedge z_{j+1} \in Z_{i_{j+1}} \wedge \kappa(i_j, \sigma_j) = i_{j+1}.$$

Therefore, $\kappa(i_0, s)!$. Since $s \in L_m(G \wedge S)$, we have $\kappa(i_0, s) \in I_m$. Therefore $s \in L_m(G \wedge S_{\mathcal{C}})$, namely

$$L_m(G \wedge S) \subseteq L_m(G \wedge S_{\mathcal{C}}).$$

By taking the prefix closure on both sides, and recalling that $\overline{L_m(G \wedge S)} = L(G \wedge S)$, we have

$$L(G \wedge S) \subseteq L(G \wedge S_{\mathcal{C}}).$$

2. For the reverse inclusion, let $s \in L(G \wedge S_{\mathcal{C}})$. If $s = \epsilon$ then, as $L(G \wedge S) \neq \varnothing$, $s \in L(G \wedge S)$. Suppose $s = \sigma$. Then $\kappa(i_0, s)!$, so there are $z \in Z_{i_0}$ and $z' \in Z$ such that $\delta(z, \sigma) = z'$, namely $\sigma \in En_S(z)$. By the definition of the control cover $\mathcal{C}$, $\sigma \notin D_S(z_0)$, so either $\delta(z_0, \sigma)!$ or

$$(\forall t \in \Sigma^*)\delta(z_0, t) = z_0 \Rightarrow t\sigma \notin L(G).$$

But since $s = \sigma \in L(G \wedge S_{\mathcal{C}})$, we have $\epsilon s \in L(G)$ and $\delta(z_0, \epsilon) = z_0$. Thus, we conclude that $\delta(z_0, \sigma)!$, namely $s \in L(G \wedge S)$. Of course, by the definition of the control cover $\mathcal{C}$, there follows $\delta(z_0, \sigma) = z' \in Z_{i'}$ for some $i' \in I$. In general, let $s = \sigma_0\sigma_1 \cdots \sigma_k$. Repeating the foregoing argument k-fold, we see that $s \in L(G \wedge S_{\mathcal{C}})$ implies $s \in L(G \wedge S)$. This shows that $L(G \wedge S_{\mathcal{C}}) \subseteq L(G \wedge S)$.

3. Let $s \in L_m(G \wedge S_{\mathcal{C}})$. As shown above, $\delta(z_0, s)!$ with $\delta(z_0, s) = z \in \kappa(i_0, s)$. Since $\kappa(i_0, s) \in I_m$, there exists $z' \in Z_{\kappa(i_0,s)} \cap Z_m$, namely $M_S(z') = true$. By the definition of feasible supervisor, we know that there is $s' \in L_m(G \wedge S)$ such that $\delta(z_0, s') = z'$, namely $T_S(z') = true$. At the same time, $s \in L_m(G \wedge S_{\mathcal{C}})$ implies $T_S(z) = true$. By definition of control cover $\mathcal{C}$, we get $M_S(z) = M_S(z') = true$, namely $\delta(z_0, s) = z \in Z_m$, and $s \in L_m(G \wedge S)$, as required.

So far we have shown that $L(G \wedge S) = L(G \wedge S_{\mathcal{C}})$ and $L_m(G \wedge S) = L_m(G \wedge S_{\mathcal{C}})$. Finally, we need to show that $S_{\mathcal{C}}$ is a feasible supervisor, namely the conditions of control existence and control feasibility must hold. The control existence condition obviously holds because the construction of $S_{\mathcal{C}}$ from S does not disable more events than S does. Since S is feasible, namely the control existence condition holds, we know that this condition must hold for $S_{\mathcal{C}}$. For the second condition of control feasibility, notice that, by the definition of control cover $\mathcal{C}$, unobservable events selflooped at certain states in S are also selflooped at appropriate states in $S_{\mathcal{C}}$. Thus, the control feasibility condition holds for $S_{\mathcal{C}}$. Since S is marking feasible, it is not difficult to check that $S_{\mathcal{C}}$ must also be marking feasible, which completes the proof. ■

Proof of Theorem 2: With **SUPER** $= (Z, \Sigma, \delta, z_0, Z_m)$ and **SIMSUP** $= (Y, \Sigma, \eta, y_0, Y_m)$, for each $y \in Y$, let

$$Z(y) := \{z \in Z | (\exists s \in L(G \wedge S)\delta(z_0, s) = z \wedge \eta(y_0, s) = y\}$$

and define $\mathcal{C} := \{(Z(y), y) | y \in Y\}$. We now check that $\mathcal{C}$ is a control cover on **SUPER**.

By non-redundancy of **SIMSUP**, we have $Z(y) \neq \varnothing$ for all $y \in Y$. Since **SUPER** is obtained by the uncertainty subset construction, for each $z \in Z$, there is $s \in L(G \wedge S) = L(G \wedge$ **SIMSUP**$)$ with $\delta(z_0, s) = z$ and $\eta(y_0, s)!$. Hence, $z \in Z(\eta(y_0, s))$. Thus, $\mathcal{C} = \{(Z(y), y) | y \in Y\}$ covers Z.

Next, fix $y \in Y$ and let $a, b \in Z(y)$ with $\sigma \in En_{\textbf{SUPER}}(a)$. We need to show that $\sigma \notin D_{\textbf{SUPER}}(b)$. Since **SUPER** is constructed via the uncertainty subset construction, we know that for all $s \in L(G \wedge S)$ such that $\delta(z_0, s) = a$, there exists $s' \in P_o^{-1}(P_o(s)) \cap L(G \wedge S)$ such that $s'\sigma \in L(G \wedge S)$. In addition, $\delta(z_0, s') = a$. Since $a \in Z(y)$, there exists $\hat{s} \in L(G \wedge S)$ such that $\delta(z_0, \hat{s}) = a$ and $\eta(y_0, \hat{s}) = y$. Thus, we know that there exists $\hat{s}' \in P_o^{-1}(P_o(\hat{s})) \cap L(G \wedge S)$ such that $\hat{s}'\sigma \in L(G \wedge S)$ and $\delta(z_0, \hat{s}') = a$. Since **SIMSUP** is a feasible supervisor, we know that $\eta(y_0, \hat{s}') = y$. Thus, $\eta(y, \sigma)!$. Since $b \in Z(y)$, there exists $t \in L(G \wedge S)$ such that $\delta(z_0, t) = b$ and $\eta(y_0, t) = y$. If there exists $t'\sigma \in L(G)$ such that $\delta(z_0, t') = b$, we know that there must exist $\hat{t} \in P_o^{-1}(P_o(t)) \cap L(G \wedge S)$ such that $\hat{t}\sigma \in L(G)$, $\delta(z_0, \hat{t}) = b$ and, because **SIMSUP** is a feasible supervisor, we have $\eta(y_0, \hat{t}) = y$. Since $\hat{t}\sigma \in L(G \wedge$ **SIMSUP**$) = L(G \wedge$ **SUPER**$)$, we know that $\delta(b, \sigma)!$. Thus, $\sigma \notin D_{\textbf{SUPER}}(b)$, namely $En_{\textbf{SUPER}}(a) \cap D_{\textbf{SUPER}}(b) = \varnothing$, as required.

Next, we show that

$$T_{\textbf{SUPER}}(a) = T_{\textbf{SUPER}}(b) \Rightarrow M_{\textbf{SUPER}}(a) = M_{\textbf{SUPER}}(b).$$

To this end, let $y \in Y$ and $a, b \in Z(y)$ with $M_{\textbf{SUPER}}(a) \neq M_{\textbf{SUPER}}(b)$. Without loss of generality, assume that $M_{\textbf{SUPER}}(a) = true$ and $M_{\textbf{SUPER}}(b) = false$. Since $M_{\textbf{SUPER}}(a) = true$, there exists $s \in L_m(G \wedge S)$

such that $\delta(z_0, s) = a$. Thus, $T_{\textbf{SUPER}}(a) = true$. Since $a \in Z(y)$, we know that there exists $s' \in L(G \wedge S)$ such that $\delta(z_0, s') = a$ and $\eta(y_0, s') = y$. Thanks to the uncertainty subset construction, we know that there exists $\hat{s} \in P_o^{-1}(P_o(s')) \cap L_m(G \wedge S)$ such that $\delta(z_0, \hat{s}) = a$ and, because **SIMSUP** is a feasible supervisor, we have $\eta(y_0, \hat{s}) = y$. This means $y \in Y_m$. Since $b \in Z(y)$, for all $t \in L(G \wedge S)$ with $\delta(z_0, t) = b$, thanks to the uncertainty subset construction and the fact that **SIMSUP** is a feasible supervisor, we can deduce that there exists $\hat{t} \in L(G \wedge S)$ such that $\delta(z_0, \hat{t}) = b$, $\eta(y_0, \hat{t}) = y$ and $t \in L_m(G) \iff \hat{t} \in L_m(G)$. Since $M_{\textbf{SUPER}}(b) = false$, we know that $\hat{t} \notin L_m(G \wedge S) = L_m(G \wedge \textbf{SIMSUP})$. Since $y \in Y_m$, we deduce that $\hat{t} \notin L_m(G)$. Thus, $t \notin L_m(G)$. Since t is arbitrarily chosen, we know that $T_{\textbf{SUPER}}(b) = false$. Thus, we have

$$M_{\textbf{SUPER}}(a) \neq M_{\textbf{SUPER}}(b) \Rightarrow T_{\textbf{SUPER}}(a) \neq T_{\textbf{SUPER}}(b),$$

or equivalently,

$$T_{\textbf{SUPER}}(a) = T_{\textbf{SUPER}}(b) \Rightarrow M_{\textbf{SUPER}}(a) = M_{\textbf{SUPER}}(b).$$

Finally, we need to show that for each $y \in Y$ and $\sigma \in \Sigma$, there exists $y' \in Y$ such that

$$(\forall z \in Z(y))\delta(z, \sigma)! \Rightarrow \delta(z, \sigma) \in Z(y').$$

Let $z \in Z(y)$ and $\delta(z, \sigma)!$. Clearly, there exists $s \in L(G \wedge S)$ such that $s\sigma \in L(G \wedge S)$ and $\delta(z_0, s) = z$. By using an argument similar to the above, we know that there exists $s' \in P_o^{-1}(P_o(s)) \cap L(G \wedge S)$ such that $\delta(z_0, s') = z$, $\eta(y_0, s') = y$, and $s'\sigma \in L(G \wedge S)$. Clearly, $\eta(y, \sigma)!$. Thus, $\delta(z, \sigma) \in Z(\eta(y, \sigma))$, as required.

So far we have shown that $\mathcal{C} = \{(Z(y), y) | y \in Y\}$ is a control cover on **SUPER**. Let $S_{\mathcal{C}} := (Y, \Sigma, \kappa, y_0, Y_m)$ be induced from $\mathcal{C}$, where

$$(\forall y \in Y)(\forall \sigma \in \Sigma)\, \kappa(y, \sigma) := \begin{cases} \eta(y, \sigma) & \text{if } \eta(y, \sigma)!, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

For any $y \in Y$ and $\sigma \in \Sigma$, assume that there exists $z \in Z(y)$ such that $\delta(z, \sigma)!$. For all $z' \in Z(y)$, we know that there exists $s \in L(G \wedge S)$ such that $\delta(z_0, s) = z'$ and $\eta(y_0, s) = y$. If $\delta(z', \sigma)!$, by the construction of **SUPER** we know that there must exist $s' \in \Sigma^*$ and $u \in \Sigma_{uo}^*$ such that $P_o(s) = P_o(s')$, $\delta(z_0, s'u) = z'$ and $s'u\sigma \in L(G \wedge \textbf{SUPER}) = L(G \wedge S) \subseteq L(\textbf{SIMSUP})$ (owing to the control equivalence of **SIMSUP** to S). Since **SIMSUP** is feasible, by the control feasibility property and the fact that $P_o(s) = P_o(s')$, we know that $\eta(y_0, s) = \eta(y_0, s') = y$. Since $s'u\sigma \in L(\textbf{SIMSUP})$, by the control feasibility property and the fact that $u \in \Sigma_{uo}^*$, we know that $\eta(y_0, s'u\sigma) = \eta(y, u\sigma) = \eta(y, \sigma)$. Thus, $\delta(z', \sigma) = \delta(z_0, s'u\sigma) \in Z(\eta(y, \sigma))$. This means

$$(\exists z \in Z(y))\delta(z, \sigma)! \Rightarrow (\forall z' \in Z(y))[\delta(z', \sigma)! \Rightarrow \delta(z', \sigma) \in Z(\eta(y, \sigma))].$$

By Theorem 1 we know that $S_{\mathcal{C}}$ is a feasible supervisor, which is control equivalent to S with respect to G. In addition, there exists a natural bijective mapping between the state set Y of **SIMSUP** and the state set Y of $S_{\mathcal{C}}$ with respect to the control cover $\mathcal{C}$,

$$\theta : Y \to Y : y \mapsto \theta(y) := y.$$

Thus, $S_{\mathcal{C}}$ is DES-isomorphic to **SIMSUP**, which completes the proof. ■

Proof of Theorem 3: Let **SUPER** $= (\hat{Z}, \Sigma, \hat{\delta}, \hat{z}_0, \hat{Z}_m)$ and $S' = (Z', \Sigma, \delta', z_0', Z_m')$. Let $s \in L(G \wedge S)$. Recall that **SUPER** is obtained by applying the uncertainty subset construction on $G \wedge S$. Thus, we know that the following properties hold:

(a) $(\forall \sigma \in En_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)))(\exists s' \in L(G \wedge S))\, s'\sigma \in L(G \wedge S) \wedge \hat{\delta}(\hat{z}_0, s') = \hat{\delta}(\hat{z}_0, s)$,
i.e., no transition in **SUPER** is redundant.

(b) $(\forall s', s'' \in L(G \wedge S))P_o(s') = P_o(s'') \Rightarrow \hat{\delta}(\hat{z}_0, s') = \hat{\delta}(\hat{z}_0, s'')$,
i.e., strings with the same projected image with respect to P_o reach the same state.

(c) for any two strings $s', s'' \in L(G \wedge S)$, if $\hat{\delta}(\hat{z}_0, s') = \hat{\delta}(\hat{z}_0, s'')$, then
$\{\sigma \in \Sigma | (\exists t \in L(G \wedge S) \cap P_o^{-1}(s'))t\sigma \in L(G \wedge S)\} = \{\sigma' \in \Sigma | (\exists t' \in L(G \wedge S) \cap P_o^{-1}(s''))t'\sigma' \in L(G \wedge S)\}$,
i.e., for any event $\sigma \in \Sigma$, there exists a string $t \in L(G \wedge S) \cap P_o^{-1}(s')$ such that σ is enabled after t iff there exists a string $t' \in L(G \wedge S) \cap P_o^{-1}(s'')$ such that σ is enabled after t'.

Let $z := \hat{\delta}(\hat{z}_0, s)$. Thus, we know that

$$\begin{aligned}
& En_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)) \\
= \;& \cup_{s' \in L(G \wedge S): \hat{\delta}(\hat{z}_0, s') = z} \{\sigma \in \Sigma | s'\sigma \in L(G \wedge S)\} \text{ by Property (a)} \\
= \;& \cup_{s' \in L(G \wedge S): \hat{\delta}(\hat{z}_0, s') = z} \{\sigma \in \Sigma | (\exists t \in L(G \wedge S))t\sigma \in L(G \wedge S) \wedge P_o \\
= \;& \{\sigma \in \Sigma | (\exists s' \in L(G \wedge S))s'\sigma \in L(G \wedge S) \wedge P_o(s) = P_o(s')\} \text{ by P} \\
\subseteq \;& \{\sigma \in \Sigma | (\exists s' \in L(S')P_o(s) = P_o(s') \wedge s'\sigma \in L(S')\} \text{ as } L(G \wedge S) \\
= \;& En_{S'}(\delta'(z_0', s))
\end{aligned}$$

Thus, $En_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)) \subseteq En_{S'}(\delta'(z_0', s))$.

To show that $D_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)) \subseteq D_{S'}(\delta'(z_0', s))$, let $\sigma' \in D_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s))$ and $z := \hat{\delta}(\hat{z}_0, s)$. Then $\neg\hat{\delta}(z, \sigma')!$ but there exists $s'\sigma' \in L(G)$ such that $\hat{\delta}(\hat{z}_0, s') = z$. Clearly, $s' \in L(G \wedge S)$ but $s'\sigma' \notin L(S)$. In addition, by definition of the uncertainty subset construction, we can choose s' in such a way that $P_o(s) = P_o(s')$. Thus, we know that

$$\sigma' \in \{\sigma \in \Sigma | (\exists s' \in L(G \wedge S))\, s'\sigma \in L(G) \wedge P_o(s) = P_o(s') \wedge s'\sigma \notin L(S)\}$$

which means $D_{\textbf{SUPER}}(z) \subseteq \{\sigma \in \Sigma | (\exists s' \in L(G \wedge S))\, s'\sigma \in L(G) \wedge P_o(s) = P_o(s') \wedge s'\sigma \notin L(S)\}$. Thus, we have

$$\begin{aligned}
D_{\textbf{SUPER}}(z) \;&\subseteq\; \{\sigma \in \Sigma | (\exists s' \in L(G \wedge S))\, s'\sigma \in L(G) \wedge P_o(s) = P_o(s' \\
&\subseteq\; \{\sigma \in \Sigma | (\exists s' \in L(G \wedge S'))\, s'\sigma \in L(G) \wedge s'\sigma \notin L(S') \wedge \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{becaus} \\
&=\; D_{S'}(\delta'(z_0', s))
\end{aligned}$$

Assume that $M_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)) = true$. Then $s \in L_m(G \wedge S) = L_m(G \wedge S')$, which means $M_{S'}(\delta'(z_0', s)) = true$. Assume that $T_{\textbf{SUPER}}(\hat{\delta}(\hat{z}_0, s)) =$

$true$. Then $s \in L_m(G)$, which means $T_{S'}(\delta'(z'_0, s)) = true$. Thus, we have **SUPER** $\preceq S'$. ■

Proof of Theorem 4: Let $S_j = (Z_j, \Sigma, \delta_j, z_{j,0}, Z_{j,m})$ $(j = 1, 2)$, and $\mathcal{R}_j \subseteq Z_j \times Z_j$ the binary compatibility relation over Z_j. Let $\mathcal{C}_2 = \{(Z_{2,i}, i) | Z_{2,i} \subseteq Z_2 \wedge i \in I_2\}$ be a minimal control cover on S_2. By Definition 1 we know that

1) $(\forall i \in I_2)\, Z_{2,i} \neq \varnothing \wedge (\forall z, z' \in Z_{2,i})\, (z, z') \in \mathcal{R}_2$,
2) $(\forall i \in I_2)(\forall \sigma \in \Sigma)(\exists j \in I_2)[(\forall z \in Z_{2,i})\delta_2(z, \sigma)! \Rightarrow \delta_2(z, \sigma) \in Z_{2,j}]$.

Since S_2 is non-redundant with respect to S, we can derive that for each $z \in Z_2$ there exists $s \in L(G \wedge S)$ such that $\delta_2(z_{2,0}, s) = z$. For each $(Z_{2,i}, i) \in \mathcal{C}_2$, let

$$\mathcal{L}(Z_{2,i}) := \{s \in L(G \wedge S) | \delta_2(z_{2,0}, s) \in Z_{2,i}\} \cup (\Sigma^* \setminus L(G)).$$

We can easily check that

$$En_{S_2}(Z_{2,i}) := \cup_{z \in Z_{2,i}} En_{S_2}(z) = \{\sigma \in \Sigma | s\sigma \in L(G \wedge S) \wedge s \in \mathcal{L}(Z_{2,i})\}.$$

Since $S_1, S_2 \in \mathcal{F}(G, S)$, we know that $\mathcal{L}(Z_{2,i}) \cap L(G) \subseteq L(G \wedge S_2) = L(G \wedge S_1)$. Let

$$\hat{\mathcal{C}}_1 := \{(Z_{1,i}, i) | Z_{1,i} \subseteq Z_1 \wedge [z \in Z_{1,i} \iff (\exists s \in \mathcal{L}(Z_{2,i}))\delta_1(z_{1,0}, s) = z] \wedge i \in I_2\}.$$

We now show that $\hat{\mathcal{C}}_1$ is a control cover of S_1. First, we show that $\{Z_{1,i} | i \in I_2\}$ is a cover of Z_1. To see this, notice that $\cup_{i \in I_2} \mathcal{L}(Z_{2,i}) \cap L(G) = L(G \wedge S) = L(G \wedge S_2) = L(G \wedge S_1)$. Since S_1 is also non-redundant with respect to S, we know that $\{Z_{1,i} | i \in I_2\}$ must be a cover of Z_1. It is obvious that, for all $(Z_{1,i}, i), (Z_{1,j}, j) \in \hat{\mathcal{C}}_1$, we have that $i = j$ implies $Z_{1,i} = Z_{1,j}$.

To show that $\hat{\mathcal{C}}_1$ is a control cover of S_1, we need to show that the remaining Conditions 2-3 stated in Definition 1 hold. To check Condition 2, for each $(Z_{1,i}, i) \in \hat{\mathcal{C}}_1$ and for all $z_1, z'_1 \in Z_{1,i}$, we know that there exist $s, s' \in \mathcal{L}(Z_{2,i})$ such that $\delta_1(z_{1,0}, s) = z_1$ and $\delta_1(z_{1,0}, s') = z'_1$. On the other hand, let $z_2 = \delta_2(z_{2,0}, s)$ and $z'_2 = \delta_2(z_{2,0}, s')$. Since $S_1 \preceq S_2$, we know that

- $En_{S_1}(z_1) \subseteq En_{S_2}(z_2)$ and $D_{S_1}(z_1) \subseteq D_{S_2}(z_2)$,
- $M_{S_1}(z_1) = true \Rightarrow M_{S_2}(z_2) = true$,
- $T_{S_1}(z_1) = true \Rightarrow T_{S_2}(z_2) = true$,

and

- $En_{S_1}(z'_1) \subseteq En_{S_2}(z'_2)$ and $D_{S_1}(z'_1) \subseteq D_{S_2}(z'_2)$,
- $M_{S_1}(z'_1) = true \Rightarrow M_{S_2}(z'_2) = true$,
- $T_{S_1}(z'_1) = true \Rightarrow T_{S_2}(z'_2) = true$.

Since $(z_2, z'_2) \in \mathcal{R}_2$, we have

- $En_{S_2}(z_2) \cap D_{S_2}(z'_2) = En_{S_2}(z'_2) \cap D_{S_2}(z_2) = \varnothing$,
- $T_{S_2}(z_2) = T_{S_2}(z'_2) \Rightarrow M_{S_2}(z_2) = M_{S_2}(z'_2)$.

Thus, we can easily conclude that

$$En_{S_1}(z_1) \cap D_{S_1}(z'_1) = En_{S_1}(z'_1) \cap D_{S_1}(z_1) = \varnothing.$$

To show that

$$T_{S_1}(z_1) = T_{S_1}(z'_1) \Rightarrow M_{S_1}(z_1) = M_{S_1}(z'_1),$$

it is clear that if $T_{S_1}(z_1) = T_{S_1}(z'_1) = false$, then by the definition of M_{S_1} we know that $M_{S_1}(z_1) = M_{S_1}(z'_1) = false$. So we only need to show that when $T_{S_1}(z_1) = T_{S_1}(z'_1) = true$, we have $M_{S_1}(z_1) = M_{S_1}(z'_1)$. Suppose otherwise. Then without loss of generality, let $M_{S_1}(z_1) = true$ and $M_{S_1}(z'_1) = false$. Since $M_{S_1}(z'_1) = false$ and $T_{S_1}(z'_1) = true$, we can conclude that $M_{S_2}(z'_2) = false$ due to the control equivalence of S_1 and S_2. But on the other hand, since $S_1 \preceq S_2$, we know that $M_{S_1}(z_1) = true$ implies that $M_{S_2}(z_2) = true$. Thus, we have $T_{S_2}(z_2) = T_{S_2}(z'_2) = true$, $M_{S_2}(z_2) = true$, and $M_{S_2}(z'_2) = false$, which contradicts our assumption that

$$T_{S_2}(z_2) = T_{S_2}(z'_2) \Rightarrow M_{S_2}(z_2) = M_{S_2}(z'_2).$$

Thus, we can only have $M_{S_1}(z_1) = M_{S_1}(z'_1)$, which means $(z_1, z'_1) \in \mathcal{R}_1$.

To see that Condition 3 is satisfied, for each $i \in I_2$, $\sigma \in \Sigma$, we know that there exists $j \in I_2$ such that

$$(\forall z \in Z_{2,i})\delta_2(z, \sigma)! \Rightarrow \delta_2(z, \sigma) \in Z_{2,j}.$$

For each $z' \in Z_{1,i}$, if $\delta_1(z', \sigma)!$, there there are two cases. Case 1: there exists $s \in \mathcal{L}(Z_{2,i})$ such that $\delta_1(z_{1,0}, s) = z'$ and $s\sigma \in L(G \wedge S)$. Since $\delta_2(z_{2,0}, s) = z'' \in Z_{2,i}$ and $\delta_2(z'', \sigma)!$, we know that $s\sigma \in \mathcal{L}(Z_{2,j})$. Thus, $\delta_1(z', \sigma) \in Z_{1,j}$. Case 2: for all $s' \in \mathcal{L}(Z_{2,i})$ with $\delta_1(z_{1,0}, s') = z'$, we have $s'\sigma \notin L(G \wedge S)$. Then clearly $s'\sigma \notin L(G)$ because otherwise the first condition of control cover will be violated. Thus, we still have that $s'\sigma \in \mathcal{L}(Z_{2,j})$. Thus, $\delta_1(z', \sigma) \in Z_{1,j}$. So in either case, we can conclude that

$$(\forall z \in Z_{1,i})\delta_1(z, \sigma)! \Rightarrow \delta_1(z, \sigma) \in Z_{1,j},$$

which completes our proof that $\hat{\mathcal{C}}_1$ is a control cover of S_1.

Clearly, $|\hat{\mathcal{C}}_1| = |\mathcal{C}_2|$. On the other hand, if $\mathcal{C}_1$ is a minimal control cover of S_1, we know that $|\mathcal{C}_1| \leq |\hat{\mathcal{C}}_1|$. Thus, we can conclude that $|\mathcal{C}_1| \leq |\mathcal{C}_2|$. ■