Mean Payoff Supervisory Control under Partial Observation

Yiding Ji, Xiang Yin and Stéphane Lafortune

***Abstract*— The problem under investigation is mean payoff supervisory control on a partially observed quantitative discrete event system modeled by a finite state weighted automaton. We intend to design a partial-observation supervisor such that the limit-average weights of all infinite sequences in the supervised system remain nonnegative. This problem may be viewed as a two-player quantitative game between the supervisor and the environment, with asymmetric information and a mean payoff objective. To cope with partial observation of the supervisor, we introduce the energy information state which incorporates information about both state estimate and energy change for supervisor's decision making. Based on that, we transfer the supervisory control problem into a two-player reachability game under full observation and propose a finite bipartite structure called First Cycle Energy Inclusive Controller (FCEIC). Further analysis demonstrates that winning strategies in the FCEIC lead to solutions to the original control problem.**

I. Introduction

Supervisory control under the framework of discrete event systems (DES) has been a classic problem since initiated in [18]. The supervisor is designed to restrict the original behavior of the system so that a given specification is achieved. Ever since that, supervisory control has been thoroughly studied for various DES models, such as finite state automata, see, e.g., [18], Petri nets, see, e.g., [7], networked systems, see, e.g., [21] and other classes of systems.

In the context of DES, due to the limited sensing capabilities, the plant is usually partially observed which gives rise to supervisory control under partial observation [12]. Many works fall in this category, see, e.g., [2], [10], [23]. Recently, a novel approach was developed in [24] and extended in [25] to synthesize maximal permissive partial-observation supervisors for enforcing a series of qualitative properties in DES without assumptions on the relation between controllable events and observable events.

Besides logical properties, supervisory control of quantitative DES has also been investigated by introducing some performance measures. Optimal supervisory control is one problem of particular interest, starting with [15]. Since that, different frameworks of optimal supervisory control have been discussed. The work [20] defined event enablement and disablement costs, then finds a minimum cost controller by a dynamic programming approach. This framework was extended in [13] by considering partial observation of the system. Furthermore, the work [14] studied optimal supervisory control in probabilistic DES and the work [22] proposed a timed optimal supervisor. All the above works evaluate the supervisor's performance for finite behaviors of the system. In contrast, the work [17] considered optimizing the worst case limit average weight of the infinite sequences generated by the controlled system. The problem was formulated and solved as a mean payoff game between the supervisor and the environment, under full observation.

In many practical situations, the operation of the system may generate or consume some resources, e.g., energy. The specification may be to design a controller such that the long run average rate of resource gain/cost is above or below a certain threshold. Also it may not be feasible to sense every step of the system's execution thus the designer only has partial information of the system. Motivated by these considerations, we investigate supervisory control with a mean payoff objective under partial observation. To the best of our knowledge, there is no prior research on this topic.

In this work, the goal of the supervisor is to prevent all infinite behaviors of the system that violate the mean payoff objective. Besides, the supervised system should be non-terminating so that it may execute events perpetually. To achieve these goals, we define *energy information states*, which incorporate necessary information about the system's state and payoff for supervisor's decision making. Then we transfer the proposed problem into a two-player reachability game [1] between the supervisor and the "environment" (aka system) on a finite information structure. The structure is called First Cycle Energy Inclusive Controller (FCEIC) and by construction, we show that the winning strategies for the supervisor in the FCEIC correspond to solutions for our problem. The FCEIC is inspired by the information structure in [9] for the different problem of opacity enforcement.

Our work is also inspired by quantitative game theory in theoretical computer science, especially mean payoff games [6]. A mean payoff game is an infinite-duration turn-based two-player game on a weighted graph. The two players take turns to play by selecting an outgoing edge at their positions, resulting in an infinite path. The objective of the first player is to enforce the limit average weight of the traversed edges above a given threshold while the second player is to do the opposite, thus the game is *zero sum* in general. Well structured solutions were proposed with complexity analyzed for mean payoff games of *perfect information* [6], where both players have complete knowledge of their opponent's moves and positions up to the current state. What is more challenging is a mean payoff game with *imperfect information* where one player may be absent from the complete decision history of the other player [4]. Such games are in general undecidable [5] and some decidable classes are presented in [8], which motivates our restrictions

Research supported in part by the US National Science Foundation under grants CNS-1446298 and CNS-1738103.

Y. Ji and S. Lafortune are with the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, Michigan, USA. {jiyiding;stephane}@umich.edu

X. Yin is with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China. {yinxiang}@sjtu.edu.cn

on the system in this work. Finally, the work [16] also discussed supervisory control in a game framework, namely fixed-initial-credit energy game under partial observation.

The following sections are organized as follows. Section II describes the system model. In Section III, we make some assumptions on the system and formulate the mean payoff supervisory control problem under partial observation. Section IV introduces energy information states and the First Cycle Energy Inclusive Controller (FCEIC). Section V analyzes some properties of the FCEIC and shows that the winning control strategies in the FCEIC lead to solutions of the proposed problem in Section III. Finally, Section VI concludes the paper.

II. System model

We consider supervisory control in a quantitative discrete event system modeled as a weighted finite-state automaton:

$$G = (X, E, f, x_0, \omega) \tag{1}$$

where X is the finite state space, E is the finite set of events, $f : X \times E \to X$ is the partial transition function, $x_0 \in X$ is the initial state, $\omega : E \to \mathbb{Z}$ is the weight function that assigns an integer to each event. We view the weight of an event as the *energy payoff* in this work. A positive number stands for energy gain while a negative number stands for energy cost. For the sake of calculation in the following discussion, we assume the system's initial energy is 0. The transition function is extended to $X \times E^*$ in the standard manner and we still denote the extended function by f. The language generated by G is defined as $\mathscr{L}(G) = \{s \in E^* : f(x_0, s)!\}$ where ! means "is defined". We denote by $s \leq u$ if string s is a prefix string u, and $s < u$ if $s \leq u, s \neq u$. The function ω is additive and its domain can be extended to E^* by letting $\omega(\varepsilon) = 0$, $\omega(se_o) = \omega(s) + \omega(e_o)$ for $s \in E^*$ and $e \in E$. Given a string $s = e_1 e_2 \cdots e_n \in \mathscr{L}(G)$, the *payoff* of s is just the sum of the weight of each event, i.e. $\omega(s)$, which is also called the *energy level* of the system by s.

In this work, we assume that the safety property is always satisfied and we do not consider the non-blockingness property, thus no marked states are included in the system model. Instead, we discuss the (weak) *liveness* property: a system G is live if its generated language $\mathscr{L}(G)$ is live, i.e., $\forall s \in \mathscr{L}(G)$, $\exists u \in E$, s.t. $su \in \mathscr{L}(G)$. That is, there is a transition defined at each state in the system, which thus never terminates. The liveness requirement on G is without loss of generality since it can be relaxed by adding observable self-loops at terminal states where no active events are defined, as is done in [19].

Given automaton G, for $x_1, x_2 \in X$ and $e \in E$, we denote by $x_1 \xrightarrow{e} x_2$ if $f(x_1, e) = x_2$. A *run* in G is a sequence of states and events: $r = x_1 \xrightarrow{e_1} x_2 \xrightarrow{e_2} \cdots \xrightarrow{e_{n-1}} x_n$ and it may be infinitely long. Denote by $Run(G)$ the set of runs in G. A run is *initial* if its initial state is the initial state of the system. We say a run forms a *cycle* if $x_1 = x_n$ and a cycle is *simple* if $\forall i, j \in \{1, 2, \cdots n-1\}$, $i \neq j \Rightarrow x_i \neq x_j$. If r is a cycle, there is a corresponding loop $e_1 e_2 \cdots e_{n-1}$ starting from and ending in x_1. We further call the loop *simple* if the cycle is simple.

For a finite run $x_1 \xrightarrow{e_1} x_2 \xrightarrow{e_2} \cdots \xrightarrow{e_n} x_{n+1}$, its payoff is $\sum_{i=1}^{n} \omega(e_i)$ and its mean payoff is $\frac{1}{n}\sum_{i=1}^{n} \omega(e_i)$. Furthermore, we let $Run_{inf}(G)$ be the set of infinite runs in G and define $V_{lim} : Run_{inf}(G) \to \mathbb{R}$ as *limit mean payoff* where for an infinite run $r = x_1 \xrightarrow{e_1} x_2 \xrightarrow{e_2} \cdots$, $V_{lim}(r) = \liminf_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \omega(e_i)$.

The system is controlled by a *supervisor* [3] that dynamically enables/disables events of the system so that some specification is achieved. The event set E is partitioned as $E = E_c \cup E_{uc}$, where E_c is the set of controllable events and E_{uc} is the set of uncontrollable events. A control decision $\gamma \in 2^E$ by the supervisor is *admissible* if $E_{uc} \subseteq \gamma$, i.e., the supervisor never disables uncontrollable events. We define $\Gamma = \{\gamma \in 2^E : E_{uc} \subseteq \gamma\}$ as the set of admissible control decisions. The system is also partially observable and E is partitioned as $E = E_o \cup E_{uo}$, where E_o is the set of observable events and E_{uo} is the set of unobservable events. Given a string $t \in E^*$, its natural projection $P : E^* \to E_o^*$ is recursively defined as $P(t) = P(t'e) = P(t')P(e)$ where $t' \in E^*$ and $e \in E$. The projection of an event is $P(e) = e$ if $e \in E_o$ and $P(e) = \varepsilon$ if $e \in E_{uo} \cup \{\varepsilon\}$, where ε is the empty string.

A supervisor is a function $S : P[\mathscr{L}(G)] \to \Gamma$ and we denote by $\mathbb{S}$ the set of supervisors. A supervisor makes decisions only based on the projected behavior of the system. We use S/G to represent the controlled system under S. Accordingly, we denote by $\mathscr{L}(S/G)$ the language generated in S/G and $Run(S/G)$ the set of runs in S/G, respectively.

We define some operators in G. Given a set of states $q \subseteq X$, the *unobservable reach*, denoted by $UR(q)$, is defined as: $UR(q) = \{x' \in X : \exists x \in q, \exists s \in E_{uo}^*,$ s.t. $f(x,s) = x'\}$. Specifically, the unobservable reach under a set of events $\gamma \subseteq E$, denoted as $UR_\gamma(q)$, is defined as: $UR_\gamma(q) = \{x' \in X : \exists x \in q, \exists s \in (E_{uo} \cap \gamma)^*,$ s.t. $f(x,s) = x'\}$. Besides, the *observable reach* under observable event e_o, denoted by $Next_{e_o}(q)$, is defined as: $Next_{e_o}(q) = \{x' \in X : \exists x \in q$ s.t. $f(x, e_o) = x'\}$. Then we define the *observer* of G: $Obs(G) = (X_{obs}, E_o, \delta, x_{obs,0})$ where $X_{obs} \subseteq 2^X$ is the state space, $x_{obs,0} = UR(x_0)$ is the initial state and $\forall x_{obs} \in X_{obs}$, $e_o \in E_o$, $\delta(x_{obs}, e_o) = UR(Next_{e_o}(x_{obs}))$. The weight function is omitted in the definition of observer.

III. Problem Formulation

In this section, we formulate the mean payoff supervisory control problem studied in this work. Before introducing the problem, we make some assumptions on the system.

In the observer of the system, given a state $x_{obs} \in X_{obs}$, let $Loop(x_{obs}) = \{l \in E_o^* \setminus \{\varepsilon\} : \delta(x_{obs}, l) = x_{obs}$ and $\forall l' < l$ s.t. $l' \neq \varepsilon, \delta(x_{obs}, l') \neq x_{obs}\}$ be the set of simple loops starting from x_{obs}. Also let $SimLp(x_{obs}, l) = \{t \in E^* \setminus \{\varepsilon\} : \exists x \in X_{obs}, \exists l \in Loop(x_{obs})$ s.t. $f(x,t) = x, P(t) = l$ and $\forall t' < t, f(x,t') \neq x\}$ be the set of non-ε simple loops with the same projection l, starting from some states in x_{obs}.

Assumption 1: Given automaton G and its observer, $\forall x_{obs} \in X_{obs}$, $\forall l \in Loop(x_{obs})$, and $\forall s, s' \in SimLp(x_{obs}, l)$, we have either $\omega(s) < 0 \Rightarrow \omega(s') < 0$ or $\omega(s) \geq 0 \Rightarrow \omega(s') \geq 0$.

In other words, for two simple loops with the same projection, their payoffs are both nonnegative or both negative.

This assumption is inspired by the decidable classes of mean payoff games with partial observation in [8]. Later on, we will see how this assumption helps us solve the mean payoff supervisory control problem. We say the system is with *unambiguous cycle payoffs* if Assumption 1 is satisfied. We also assume that there are no unobservable loops in G.

Assumption 2: Given an automaton G, $\forall x \in X$, $\forall s \in E^* \setminus \{\varepsilon\}$, $[f(x,s) = x] \Rightarrow [P(s) \neq \varepsilon]$.

We consider both qualitative and quantitative objectives and want to design a supervisor satisfying two conditions: (1) the supervised system is live; (2) the limit mean payoff of any infinite run in the supervised system is above a threshold v. Given v, we can construct a system by letting all the weights of the original system minus v and equivalently require the limit mean payoff be above 0. So we assume $v = 0$ and formulate the *mean payoff supervisory control problem*.

Problem 1 (Mean Payoff Supervisory Control Problem): Given system G, design a supervisor $S \in \mathbb{S}$ such that: $\mathscr{L}(S/G)$ is live and for all $r \in Run_{inf}(S/G)$, $V_{lim}(r) \geq 0$.

Since the limit mean payoff only depends on the mean payoff of cycles, the key task is to ensure all cycles have nonnegative payoffs in the supervised system. The main challenge concerning this problem is that the supervisor only has partial observation of the system. Thus it is essential to construct proper estimates for both the system's current state and the energy level for supervisor's decision making. We will discuss this issue in the following section.

IV. First Cycle Energy Inclusive Controller

In this section, we define the *First Cycle Energy Inclusive Controller* (FCEIC), which is a two-player game structure between the supervisor and the environment. The FCEIC characterizes the change of system's current state and energy level under control decisions of the supervisor. This structure is inspired by the Bipartite Transition System and All Inclusive Structure in [24], [25], which include supervisors enforcing several logical properties in discrete event systems.

A. Energy Information States

We first define some orders of vectors. Given two vectors $v_1 = [v_1(1), v_1(2), \cdots, v_1(n)]$, $v_2 = [v_2(1), v_2(2), \cdots, v_2(n)] \in \mathbb{Z}^n$, we denote by $v_1 \leq v_2$ (respectively $v_1 \geq v_2$) if $\forall 1 \leq i \leq n, v_1(i) \leq v_2(i)$ (respectively $v_1(i) \geq v_2(i)$). We also denote by $v_1 < v_2$ if $\forall 1 \leq i \leq n, v_1(i) \leq v_2(i)$ and $\exists 1 \leq j \leq n$, $v_1(j) < v_2(j)$ (respectively $\forall 1 \leq i \leq n$, $v_1(i) \geq v_2(i)$ and $\exists 1 \leq j \leq n$, $v_1(j) > v_2(j)$), i.e., at least one element in v_1 is strictly smaller or larger than an element in v_2.

Problem 1 requires that a supervisor be synthesized such that every infinite run in the supervised system has a nonnegative limit mean payoff. The supervisor only has partial observation and we hope to transform this problem into a problem under full observation. In order to track both the unobservable reach of states and the payoff of enabled events under control decisions, we give the definition of the *Energy Information State* as follows where we let $I = 2^X$ be the set of (current) state estimates and $|\cdot|$ be the cardinality of a set.

Definition 1 (Energy Information State): An energy information state is: $q^e = (q, [v(1), \cdots v(|q|)]) \in 2^X \times \mathbb{Z}^{|q|}$. Let $I(q^e)$ and $E_L(q^e)$ denote the state estimate and energy level components of q^e; hence, $q^e = (I(q^e), E_L(q^e))$.

Denote by Q^E the set of energy information states. Each $q^e \in Q^E$ induces a *belief function* $h_{q^e} : X \to \mathbb{Z}$. Specifically, for $q^e \in Q^E$ where $I(q^e) = q \in 2^X$, $E_L(q^e) = \{h_{q^e}(x) : x \in q\}$. We usually put $E_L(q^e)$ in a vector form: $[h_{q^e}(x_1), \cdots h_{q^e}(x_{|q|})]$ and by convention in this work, elements in $E_L(q^e)$ are placed in an increasing order w.r.t. state names in $I(q^e)$.

We define an order $\preccurlyeq$ over Q^E: for $q^e_1, q^e_2 \in Q^E$, $q^e_1 \preccurlyeq q^e_2$ if $I(q^e_1) = I(q^e_2)$ and $E_L(q^e_1) \leq E_L(q^e_2)$. We also say that q^e_2 *subsumes* q^e_1 if $q^e_1 \preccurlyeq q^e_2$. In other words, q^e_2 shares the same state estimate with q^e_1 and the energy level vector of q^e_2 is no less than that of q^e_1 in a point-wise sense. We define another order $\prec$ over Q_E: for $q^e_1, q^e_2 \in Q_E$, $q^e_1 \prec q^e_2$ if $I(q^e_1) = I(q^e_2)$, $E_L(q^e_1) < E_L(q^e_2)$. That is to say, q^e_1 and q^e_2 have the same state estimate and there exists $E_L(q^e_1)(i) < E_L(q^e_2)(i)$ at some state $I(q^e_1)(i)$ for some $i \geq 1$.

We call $q^{ae} \in Q^E \times \Gamma$ an *augmented energy information state* and let $I_E(q^{ae})$, $\Gamma(q^{ae})$ denote the energy information state and control decision components of q^{ae}, so $q^{ae} = (I_E(q^{ae}), \Gamma(q^{ae}))$. An augmented energy information state is an energy information state augmented with a control decision. With a slight abuse of notation, we use $h_{q^{ae}}$ to stand for h_{q^e} where $q^e = I_E(q^{ae})$. Then we give two concepts:

For $\gamma \in \Gamma$, $q^{ae} \in Q^E \times \Gamma$ is a *γ-successor* of $q^e \in Q^E$ if:

- $I(q^{ae}) = UR_\gamma(I(q^e))$;
- $\forall x' \in I(q^{ae})$, $h_{q^{ae}}(x') = \min_{\xi} \{h_{q^e}(x) + \omega(\xi) : \exists x \in I(q^e), \xi \in (E_{uo} \cap \gamma)^* \text{ s.t. } f(x,\xi) = x'\}$;

Overall, $q^{ae} = (I_E(q^{ae}), \gamma)$. Its state estimate component is the unobservable reach of $I(q^e)$ under γ. We also use the belief function to track the *minimum* energy level by some unobservable string ξ reaching a possible state in $I(I_E(q^{ae}))$.

Besides, for $e_o \in E_o$, q^e is an *e_o-successor* of q^{ae} if:

- $I(q^e) = Next_{e_o}(I(I_E(q^{ae})))$;
- $\forall x \in I(q^e)$, $h_{q^e}(x) = \min_{x'} \{h_{q^{ae}}(x') + \omega(e_o) : \exists x' \in I(I_E(q^{ae})), \text{ s.t. } f(x', e_o) = x\}$;

So the state estimate component of q^e is the observable reach of $I(I_E(q^{ae}))$ under e_o. Meanwhile, we use the belief function to track the *minimum* energy level by observable event e_o reaching a possible state in $I(q^{ae})$.

A *control-observation sequence* is a sequence of states, events and control decisions in the form of $\rho = y^e_1 \xrightarrow{\gamma_1} z^e_1 \xrightarrow{e_1} y^e_2 \xrightarrow{\gamma_2} z^e_2 \cdots \xrightarrow{\gamma_{n-1}} z^e_{n-1} \xrightarrow{e_{n-1}} y^e_n$ or $\rho' = y^e_1 \xrightarrow{\gamma_1} z^e_1 \xrightarrow{e_1} y^e_2 \xrightarrow{\gamma_2} z^e_2 \cdots \xrightarrow{\gamma_{n-1}} z^e_{n-1} \xrightarrow{e_{n-1}} y^e_n \xrightarrow{\gamma_n} z^e_n$ where $\forall i \leq n$, $\gamma_i \in \Gamma$, $e_i \in E_o$, $y^e_i \in Q^E$, $z^e_i \in Q^E \times \Gamma$, z^e_i is a γ-successor of y^e_i and y^e_{i+1} is an e_i-successor of z^e_i. This sequence just characterizes the update of both information state and energy under control decisions. By convention, we denote by $\rho_k = y^e_1 \xrightarrow{\gamma_1} z^e_1 \xrightarrow{e_1} y^e_2 \xrightarrow{\gamma_2} z^e_2 \cdots \xrightarrow{\gamma_{k-1}} z^e_{k-1} \xrightarrow{e_{k-1}} y^e_k$ and $\rho'_k = y^e_1 \xrightarrow{\gamma_1} z^e_1 \xrightarrow{e_1} y^e_2 \xrightarrow{\gamma_2} z^e_2 \cdots \xrightarrow{\gamma_{k-1}} z^e_{k-1} \xrightarrow{e_{k-1}} y^e_k \xrightarrow{\gamma_k} z^e_k$, for $1 \leq k \leq n$. With the supervisor making decisions, strings are generated in the supervised system and we define strings generated by control-observation sequence.

Definition 2: Given a control-observation sequence $\rho = y^e_1 \xrightarrow{\gamma_1} z^e_1 \xrightarrow{e_1} y^e_2 \xrightarrow{\gamma_2} z^e_2 \cdots \xrightarrow{\gamma_{n-1}} z^e_{n-1} \xrightarrow{e_{n-1}} y^e_n$ or $\rho' = y^e_1 \xrightarrow{\gamma_1}$

$z_1^e \xrightarrow{e_1} y_2^e \xrightarrow{\gamma_2} z_2^e \cdots \xrightarrow{e_{n-1}} y_n^e \xrightarrow{\gamma_n} z_n^e$, define the set of generated strings recursively: $\forall 1 \leq k \leq n$, let $Str(\rho_1) = \{\varepsilon\}$, $Str(\rho_1') = \{\xi_1 \in E_{uo}^* : \exists x \in I(y_1^e), x' \in I(I_E(z_1^e)), \xi_1 \in (\gamma_1 \cap E_{uo})^*$ s.t. $f(x,\xi_1) = x'\}$, then $Str(\rho_{k+1}) = \{s_k' e_k : \exists x \in I(y_1^e), x' \in I(I_E(z_k^e)), x'' \in I(y_{k+1}^e), s_k' \in Str(\rho_k'),$ s.t. $f(x,s_k') = x', f(x',e_k) = x''\}$ and $Str(\rho_{k+1}') = \{s_{k+1}\xi_{k+1} : \exists x \in I(y_1^e), x' \in I(y_{k+1}^e), x'' \in I(I_E(z_{k+1}^e)), s_{k+1} \in Str(\rho_{k+1}), \xi_{k+1} \in (\gamma_{k+1} \cap E_{uo})^*,$ s.t. $f(x,s_{k+1}) = x', f(x',\xi_{k+1}) = x''\}$.

Then we show belief functions always return the *minimum* payoff of strings reaching a state in the state estimate of an energy (or augmented energy) information state.

Theorem 1: For a control-observation sequence $\rho = y_1^e \xrightarrow{\gamma_1} z_1^e \xrightarrow{e_1} y_2^e \xrightarrow{\gamma_2} z_2^e \cdots \xrightarrow{\gamma_{n-1}} z_{n-1}^e \xrightarrow{e_{n-1}} y_n^e$ or $\rho' = y_1^e \xrightarrow{\gamma_1} z_1^e \xrightarrow{e_1} y_2^e \xrightarrow{\gamma_2} z_2^e \cdots \xrightarrow{e_{n-1}} y_n^e \xrightarrow{\gamma_n} z_n^e$, we have $\forall x \in I(y_n^e)$, $h_{y_n^e}(x) = \min_{s \in Str(\rho)} \{\omega(s) : \exists \tilde{x} \in I(y_1^e),$ s.t. $f(\tilde{x},s) = x\}$ and $\forall x' \in I(I_E(z_n^e))$, $h_{z_n^e}(x') = \min_{s \in Str(\rho')} \{\omega(s) : \exists \tilde{x} \in I(y_1^e),$ s.t. $f(\tilde{x},s) = x'\}$.

The idea for this theorem is that since we count the minimum string payoff in obtaining a single e_o-successor or γ-successor, we can show by induction that the belief function returns the minimum payoff along strings generated in a control-observation sequence. The detailed proof is omitted here due to space limitations. Since strings generated by such a sequence have the same observation, the minimum payoff is actually due to the unobservable substrings.

B. Build the First Cycle Energy Inclusive Controller

To consider both energy flow and information flow under control, we propose the *first cycle energy inclusive controller (FCEIC)*, defined by construction in Algorithm 1. It is of the form $(Q_Y^F, Q_Z^F, E, f_{yz}^F, f_{zy}^F, \Gamma, y_0^e, Q_l^F)$ where $Q_Y^F \subseteq Q^E$ is the set of energy information states (Y-states for short); $Q_Z^F \subseteq Q^E \times \Gamma$ is the set of augmented energy information states (Z-states for short) and for $z^e \in Q_Z^F$, $z^e = (I_E(z^e), \Gamma(z^e))$; $f_{yz}^F : Q_Y^F \times \Gamma \to Q_Z^F$ is the transition function from Q_Y^F states to Q_Z^F states; $f_{zy}^F : Q_Z^F \times E_o \to Q_Y^F$ is the transition function from Q_Z^F states to Q_Y^F states; Γ is the set of admissible control decisions; $y_0^e \in Q_Y^F$ is the initial energy information state where $I(y_0^e) = x_0$ and $E_L(y_0^e) = 0$; Q_l^F is the set of leaf states.

A Z-state z^e is *deadlock free* if $\forall x \in I(I_E(z^e)), \exists e \in \Gamma(z^e)$, s.t. $f(x,e)!$, i.e., there is an event enabled at every state in its state estimate. Otherwise, z^e is *deadlocking*. As there are no unobservable cycles in G by Assumption 2, a deadlock free Z-state always has f_{zy}^F transitions defined out of it.

Algorithm 1 builds the FCEIC recursively by adding feasible e_o-successors and γ-successors. The FCEIC describes a *reachability game* between the supervisor and the environment in a general manner. A Y-state is an energy information state where the supervisor issues control decisions. If the supervisor issues an admissible control decision γ, a f_{yz}^F transition is defined out of a Y-state, which follows the definition of γ-successor. While a Z-state is an energy information state augmented with a control decision, where the environment plays by selecting observable events to occur within the enabled events by the supervisor. When a particular observable event e_o is selected to occur by the environment, a f_{zy}^F transition is defined out of a Z-state, which follows the definition of e_o-successor. In this manner, the two players take turns to play and a game is formed.

Algorithm 1: Construction of the FCEIC

```
   Input  : G
   Output : FCEIC = (Q_Y^F, Q_Z^F, E, f_yz^F, f_zy^F, Γ, y_0^e, Q_l^F)
1  Q_Y^F = {y_0^e}, Q_Z^F = ∅, Q_l^F = ∅;
2  FirstCycle(y_0^e, FCEIC);
   Procedure: FirstCycle(y^e, FCEIC)
3  for γ ∈ Γ do
4      let z^e be a γ-successor of y^e;
5      if z^e is deadlock free then
6          add transition y^e --γ--> z^e to f_yz^F;
7          if z^e ∉ Q_Z^F then
8              Q_Z^F = Q_Z^F ∪ {z^e};
9              for e_o ∈ γ ∩ E_o do
10                 let ỹ^e be an e_o-successor of z^e;
11                 add transition z^e --e_o--> ỹ^e to f_zy^F;
12                 if ỹ^e ∉ Q_Y^A then
13                     Q_Y^F = Q_Y^F ∪ {ỹ^e};
14                     if there exists a run from y_0^e:
                       y_0^e --γ_0--> z_0^e --e_0--> y_1^e ··· --γ_{n-1}--> z_{n-1}^e --e--> ỹ^e
                       and ∃j < n, s.t. y_j^e ≼ ỹ^e then
15                         stop searching from ỹ^e,
                           Sub(ỹ^e) = y_j^e, Q_l^F = Q_l^F ∪ {ỹ^e},
                           Q_lg^F = Q_lg^F ∪ {ỹ^e};
16                     if there exists a run from y_0^e: y_0^e --γ_0-->
                       z_0^e --e_0--> y_1^e --γ_1--> z_1^e ··· --γ_{n-1}--> z_{n-1}^e --e--> ỹ^e
                       and ∃j < n, s.t. ỹ^e ≺ y_j^e then
17                         stop searching from ỹ^e,
                           Q_l^F = Q_l^F ∪ {ỹ^e},
                           Q_lb^F = Q_lb^F ∪ {ỹ^e};
18                     else
19                         FirstCycle(ỹ^e, FCEIC);
```

The procedure *FirstCycle* constructs the state space of the FCEIC by a depth-first search like process. In this process, we only add deadlock free Z-states to the structure. Then we make sure that there are events enabled at every state in the state estimate of any Z-state and the system keeps operating until we stop searching. In Lines 14 and 16, if the newly added state $\tilde{y}^e$ subsumes or is subsumed by an existing state on the run from the initial state, then the two energy information states share the same state estimate but the new state may have nondecreasing or decreasing energy level vectors compared with the existing state. We also know that some simple cycles with nonnegative or negative payoffs are formed in the system for the first time. Then we terminate searching and add the new state as a leaf state of the FCEIC.

We partition the leaf states as: $Q_l^F = Q_{lg}^F \cup Q_{lb}^F$ where Q_{lg}^F represents *good leaf states* and Q_{lb}^F represents *bad leaf states*. If a good leaf state is reached, we know there exist simple cycles with a nonnegative payoff in the system; while if a bad leaf state is reached, there exist simple cycles with a negative payoff. We use $Sub(y^e)$ to store the state subsumed by good

leaf state y^e. The goal of the supervisor is to reach good leaf states but to avoid bad ones in the game, which indicates the limit mean payoff of runs in the supervised system remain nonnegative. Finally, if no state subsumes another, we call *FirstCycle* in line 19 recursively until no new states are added to the FCEIC. Furthermore, Algorithm 1 converges in finite steps and returns a finite and acyclic structure.

Theorem 2: Algorithm 1 returns a finite structure.

Proof: Prove by contradiction. Assume that the FCEIC is infinite. Since E, $\Gamma \subseteq 2^E$ and E_o are finite, the number of transitions defined at each state in the structure is finite. Then by König's lemma (see, e.g., [11]), there exists an infinite run $y_0^e \xrightarrow{\gamma_0} z_0^e \xrightarrow{e_0} y_1^e \xrightarrow{\gamma_1} z_1^e \cdots$ in the FCEIC such that it is neither the case that $\exists y_i^e, y_j^e$, $i<j$, s.t. $y_i^e \preccurlyeq y_j^e$ nor the case that $y_j^e \prec y_i^e$. Therefore, there exist y_i^e, y_j^e $(i<j)$ s.t. $I(y_i^e) = I(y_j^e)$, $E_L(y_i^e)(k) \leq E_L(y_j^e)(k)$ and $E_L(y_i^e)(l) > E_L(y_j^e)(l)$ for some $k \neq l$. Therefore, there are two simple cycles in G: $x_1 \xrightarrow{e_1} x_2 \cdots \xrightarrow{e_n} x_1$ and $x_1' \xrightarrow{e_1'} x_2' \cdots \xrightarrow{e_n'} x_1'$ s.t. $x_1, x_1' \in I(y_i^e)$, $P(e_1 \cdots e_n) = P(e_1' \cdots e_n')$, $\omega(e_1 \cdots e_n) \geq 0$ and $\omega(e_1' \cdots e_n') < 0$. However, this contradicts with Assumption 1 that G is with unambiguous cycle payoffs. ■

Example 1: We construct the FCEIC w.r.t. to the system G in Figure 1. Let G be with $E_o = \{o_1, o_2, o_3, o_4\}$, $E_{uo} = \{a_1, a_2, a_3, a_4, b_1, b_2, c_1, c_2, c_3, c_4.c_5\}$, $E_c = \{c_1, c_2, c_3, c_4, c_5\}$ and $E_{uc} = \{a_1, a_2, a_3, a_4, b_1, b_2, o_1, o_2, o_3, o_4\}$. The weight of each event is shown in the figure. We let $\{uc\}$ represent the set of uncontrollable events and use subscripts to distinguish different sets of uncontrollable events that occur from certain states. Notice that some uncontrollable events are not defined at some states in G, thus cannot occur even if always enabled. It is easy to see that G satisfies both Assumptions 1 and 2.

Then we follow Algorithm 1 to build the FCEIC in Figure 2. All admissible control decisions γ_0 through γ_6 are shown in Figure 2. For simplicity of the graph, we do not put energy level vectors in the figure but list them in the next paragraph. The elements in each energy level vector are placed in the same order as the states in the state estimate.

Here we have the following states: $y_0^e = \{\{x_0\}, 0\}$, $z_0^e = \{\{x_0, x_1, x_2\}, [0, -2, -3], \gamma_0\}$, $y_1^e = \{\{x_3, x_4\}, [-1, -2]\}$, $z_1^e = \{\{x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}, [-1, -2, 2, -1, 4, -1, 3, 2], \gamma_1\}$, $y_2^e = \{\{x_{12}\}, 6\}$, $z_2^e = \{\{x_{12}\}, 6, \gamma_2\}$, $y_{2-2}^e = \{\{x_{12}\}, 8\}$, $z_8^e = \{\{x_9, x_{12}, x_{14}\}, [2, 6, 5], \gamma_2''\}$, $y_{2-3}^e = \{\{x_{12}\}, 5\}$, $y_{2-4}^e = \{\{x_{12}\}, 8\}$, $y_3^e = \{\{x_{13}\}, 1\}$, $z_3^e = \{\{x_{13}\}, 1, \gamma_3\}$, $y_{3-2}^e = \{\{x_{13}\}, 3\}$, $z_9^e = \{\{x_{10}, x_{13}, x_{15}\}, [0, 1, 0], \gamma_3'\}$, $y_{3-3}^e = \{\{x_{13}\}, -1\}$, $y_{3-4}^e = \{\{x_{13}\}, 3\}$, $z_4^e = \{\{x_3, x_4, x_5, x_7\}, [-1, -2, 2, 4], \gamma_4\}$, $y_{1-2}^e = \{\{x_3, x_4\}, [0, -1]\}$, $z_5^e = \{\{x_3, x_4, x_6, x_8\}, [-1, -2, -1, -1], \gamma_5\}$, $y_{1-3}^e = \{\{x_3, x_4\}, [0, -1]\}$, $z_6^e = \{\{x_3, x_4\}, [-1, -2], \gamma_6\}$, $y_{1-4}^e = \{\{x_3, x_4\}, [0, -1]\}$, $y_{1-5}^e = \{\{x_3, x_4\}, [0, -1]\}$.

In the FCEIC, the game is initiated from y_0^e where the only feasible control decision is γ_0. If the supervisor plays γ_0, a Z-state z_0^e is reached where the environment can select observable event o_1 to occur. The rest of the structure is interpreted in a similar way. Notice that at y_2^e, the supervisor cannot issue control decision γ_2' to enable c_3 but to disable c_5. Otherwise, a deadlocking Z-state z_7^e is reached since no event can occur at x_{14} if c_5 is disabled.

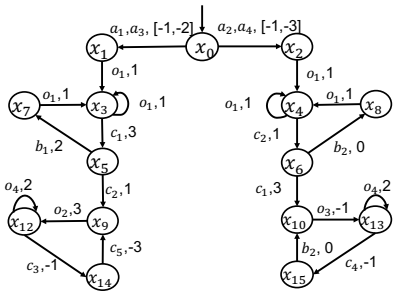

Fig. 1. The automaton G in Example 2

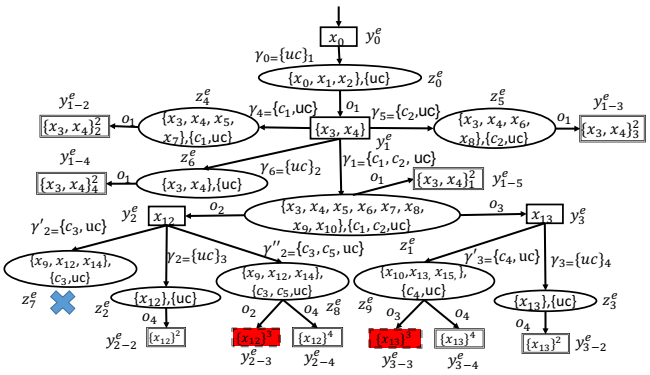

Fig. 2. The First Cycle Energy Inclusive Controller (without z_7^e)

Thus z_7^e is not included in the FCEIC. Meanwhile, we calculate the energy level vector of each state. For example, $I(y_0^e) = \{x_0\}$, $E_L(y_0^e) = 0$; then since z_0^e is γ_0-successor of y_0^e, $I(I_E(z_0^e)) = UR_{\gamma_0}(I(y_0^e)) = \{x_0, x_1, x_2\}$, $h_{z_0^e}(x_1) = \min\{\omega(a_1), \omega(a_3)\} = -2$, $h_{z_0^e}(x_2) = \min\{\omega(a_2), \omega(a_4)\} = -3$ and $z_0^e = \{\{x_0, x_1, x_2\}, [0, -2, -3], \gamma_0\}$; next since y_1^e is o_1-successor of z_1^e, $I(y_1^e) = Next_{o_1}(\{\{x_0, x_1, x_2\}) = \{x_3, x_4\}$, $h_{y_1^e}(x_3) = h_{z_0^e}(x_1) + \omega(o_1) = -1$, $h_{y_1^e}(x_4) = h_{z_0^e}(x_2) + \omega(o_1) = -2$, so $y_1^e = \{\{x_3, x_4\}, [-1, -2]\}$.

From the table, we find that $y_1^e \preccurlyeq y_{1-2}^e$, $y_1^e \preccurlyeq y_{1-3}^e$, $y_1^e \preccurlyeq y_{1-4}^e$, $y_1^e \preccurlyeq y_{1-5}^e$, $y_2^e \preccurlyeq y_{2-2}^e$, $y_2^e \preccurlyeq y_{2-4}^e$, $y_3^e \preccurlyeq y_{3-2}^e$ and $y_3^e \preccurlyeq y_{3-4}^e$ by evaluating their energy level vectors. We also find $y_{2-3}^e \prec y_2^e$ and $y_{3-3}^e \prec y_3^e$ since $E_L(y_{2-3}^e) = 5 < E_L(y_2^e) = 6$ and $E_L(y_{3-3}^e) = -1 < E_L(y_3^e) = 1$. We stop searching from the leaf states in Figure 2, then collect good leaf states $Q_{lg}^F = \{y_{1-2}^e, y_{1-3}^e, y_{1-4}^e, y_{1-5}^e, y_{2-2}^e, y_{2-4}^e, y_{3-2}^e, y_{3-4}^e\}$ and bad leaf states $Q_{lb}^F = \{y_{2-3}^e, y_{3-3}^e\}$. For example, when we reach y_{1-2}^e, we find there are three simple cycles with nonnegative payoffs in automaton G: $x_3 \xrightarrow{c_1} x_5 \xrightarrow{b_1} x_7 \xrightarrow{o_1} x_3$ with payoff 6, $x_3 \xrightarrow{o_1} x_3$ with payoff 1 and $x_4 \xrightarrow{o_1} x_4$ with payoff 1. The bad leaf states come from the two simple cycles with negative payoffs in G: $x_9 \xrightarrow{o_2} x_{12} \xrightarrow{c_3} x_{14} \xrightarrow{c_5} x_9$ with payoff -1 and $x_{10} \xrightarrow{o_3} x_{13} \xrightarrow{c_4} x_{15} \xrightarrow{b_2} x_{10}$ with payoff -2. Those two cycles should be avoided if we want to solve Problem 1.

V. Solving the Mean Payoff Control Problem

In this section, we first discuss some properties of the FCEIC and then solve Problem 1 based on it. We will also illustrate how "winning strategies" in the FCEIC lead to sound solutions for Problem 1.

By definition, the runs in the FCEIC are just finite control-observation sequences discussed in the last section. We

denote by $Run(F)$ the set of runs in the FCEIC. Given $r_f \in Run(F)$, we denote by $y^e \in r_f$ and $z^e \in r_f$ if y^e (respectively z^e) is a Y-state (respectively Z-state) in r_f. Let $Last_Y(r_f)$ and $Last_Z(r_f)$ be the last Y-state and Z-state of r_f and we denote by $Run_y(F)$ (respectively $Run_z(F)$) the set of runs whose last states are Y-states (respectively Z-states).

Then we discuss the strategies of both players in the FCEIC. We define the *supervisor's strategy (control strategy)* as $\pi_s : Run_y(F) \to \Gamma$ and *environment's strategy* as $\pi_e : Run_z(F) \to E_o$. Both players select a transition according to their strategies when it is their turn to play. Since the supervisor only has partial observation of the system and makes decisions from state estimates, we call its strategy *observation based*. Denote the set of all supervisor's strategies by Π_s and the set of all environment's strategies by Π_e. If the supervisor plays π_s while the environment plays π_e from the initial state y_0^e, then a unique initial run, denoted by $r_f(\pi_s, \pi_e)$, is generated. We also define $Run(\pi_s, y^e) = \{y^e \xrightarrow{\gamma_1} z_1^e \xrightarrow{e_1} y_2^e \cdots \xrightarrow{\gamma_{n-1}} z_{n-1}^e \xrightarrow{e_{n-1}} y_n^e : \forall i < n, \gamma_i = \pi_s(y^e \xrightarrow{\gamma_1} z_1^e \xrightarrow{e_1} y_2^e \cdots \xrightarrow{\gamma_{i-1}} z_{i-1}^e \xrightarrow{e_{i-1}} y_i^e)\}$ as the set of runs starting from y^e and *consistent* with control strategy π_s, i.e., the control decisions in the run are just specified by π_s.

In the FCEIC, we say the supervisor wins the game if only good leaf states are reached, otherwise, the environment wins the game if bad leaf states are reached. So the game on the FCEIC is a *zero sum* reachability game. From the construction of the FCEIC, the game is of full observation after introducing the energy information states, so either the supervisor or the environment has a *winning strategy* [1].

We define *the supervisor's winning region* Win_s as the set of states from which the supervisor has a strategy to reach good leaf states *for sure* regardless of the environment's strategy. Then we present Algorithm 2 to compute it.

Algorithm 2: Compute the winning region of the FCEIC

Input : $FCEIC$
Output : Win_s
while $\exists y^e \in Q_Y^F \setminus Q_{lg}^F$, *s.t.* y^e *has no successor* **do**
  Remove y^e and all $z^e \in Q_Z^F$, s.t. $f_{zy}^F(z^e, e_o) = y^e$ for some $e_o \in E_o$;
  Take the accessible part of the structure;
Denote the remaining structure by $FCEIC_w$ and return the states in it;

In Algorithm 2, bad leaf states as well as their preceding Z-states are removed. Then we further prune away Y-states that have no successors and their preceding Z-states in an iterative manner until no more states are to be removed. This process is similar to calculating the *supremal controllable sublanguage* in nonblocking supervisory control problem under full observation [3]: the bad leaf states are viewed as undesirable marked states while the good leaf states are viewed as desirable ones; besides, f_{yz}^F transitions are viewed as controllable while f_{zy}^F transitions are viewed as uncontrollable. In this way, we make sure that only good leaf states are reached under certain control strategy. In other words, any control strategy in the $FCEIC_w$ is a winning control strategy in the FCEIC, and vice versa. It is possible that Algorithm 2 returns an empty set thus the environment always wins the game regardless of the supervisor's strategies.

Then we argue that if there exists a winning strategy for the supervisor in the FCEIC, i.e., its winning region is not empty, then there always exists a supervisor solving Problem 1. The idea behind this result is quite straightforward. If a good leaf state y^e is reached under a wining control strategy π_s in the FCEIC, then for any state $x \in I(y^e)$, there forms a simple cycle from x in the supervised system with nonnegative payoff. Since a belief function in an energy information state returns the minimum string payoff by Theorem 1, the payoffs of all strings with the same observation and reaching the same state are nonnegative if the minimum string payoff is nonnegative.

We let the supervisor make the same control decision whenever the state estimate of a good leaf state is reached again. Intuitively speaking, the supervisor "ignores" the actual energy level of the system and just views the game starting from the good leaf state as the same game that starts from the state subsumed by the good leaf state. We can imagine that every good leaf state y^e is "merged" with $Sub(y^e)$ by letting all transitions going to y^e lead to $Sub(y^e)$ instead. In that way, the supervisor perpetually completes cycles with nonnegative payoffs in the supervised system since every simple cycle has a nonnegative payoff. So the limit mean payoff of every infinite run in the supervised system is also nonnegative. Furthermore, we can follow a similar argument as in [25] to show that the supervised system by any control strategy in the $FCEIC_w$ is live since there are no deadlocking Z-states in the $FCEIC_w$ and the supervisor can issue decisions indefinitely. Overall, any winning control strategy in the FCEIC solves Problem 1.

Theorem 3: If the supervisor has a winning strategy in the FCEIC, then there exists a supervisor solving Problem 1.

Proof: Suppose π_s is a winning control strategy in the FCEIC. We follow Algorithm 2 and obtain Win_s and $FCEIC_w$, so π_s is also in the $FCEIC_w$. In the following discussion, we assume that all transitions leading to a leaf state y^e in the $FCEIC_w$ lead to $Sub(y^e)$ so that the game on the $FCEIC_w$ becomes infinite-duration. That is, $\forall r_f = y_0^e \xrightarrow{\gamma_0} z_0^e \xrightarrow{e_0} y_i^e \cdots \xrightarrow{\gamma_{n-1}} z_{n-1}^e \xrightarrow{e_{n-1}} y_n^e \in Run(\pi_s, y_0^e)$ where y_0^e is the initial state of the FCEIC, if $y_n^e \in Q_{lg}^F$, we extend the domain of π_s by letting $\pi_s(r_f) = \pi_s(y_0^e \xrightarrow{\gamma_0} z_0^e \xrightarrow{e_0} y_i^e \cdots \xrightarrow{e_{m-1}} y_m^e)$ for some $m < n$ and $y_m^e \preceq y_n^e$. In the logical sense, whenever $I(y_n^e)$ is reached again, the control strategy (supervisor) makes the same decision as $I(y_n^e)$ is reached for the first time.

First, the supervised system under π_s is live following a similar argument as [25]. Then we show that the limit mean payoff of any infinite run in the supervised system is nonnegative. By perpetually making the same decision whenever a state estimate is reached, the supervisor guarantee that the payoff of any run in the supervised system never goes to negative infinity since every simple cycle in the supervised system has a nonnegative payoff. Therefore, the limit mean payoff of any infinite run in the supervised system under π_s is also nonnegative, which means π_s solves Problem 1. ■

Therefore, we have transfered Problem 1 into a reachabil-

ity game and solved it based on the FCEIC. The soundness of our approach is proven in Theorem 3. However, completeness still remains open, i.e., we do not know yet whether no solution exists for Problem 1 if we cannot find any winning control strategy in the FCEIC.

Example 2: We revisit Example 2 and find the winning region of the FCEIC following Algorithm 2. The FCEIC$_w$ is shown in Figure 3, where we use green lines to connect each good leaf state with the state subsumed by it, indicating that the supervisor will always make the same decision from them. So the game is extended to be infinite-duration. In building the FCEIC$_w$, red states y^e_{2-3} and y^e_{3-3} in Figure 2 are bad leaf states, thus they are pruned by Algorithm 2. Meanwhile, good leaf states y^e_{2-4} and y^e_{3-4} are also removed as they become no longer accessible from the initial state y^e_0 after their preceding Z-states z^e_8 and z^e_9 are removed. That means the supervisor should not choose γ''_2 at y^e or γ'_3 at y^e_2, otherwise, the environment can choose o_2 at z^e_8 or o_3 at z^e_9 to reach a bad leaf state and wins the game.

Then we locate a winning control strategy indicated by blue lines in Figure 3. The supervisor S issues γ_0 at y^e_0, γ_1 at y^e_1, γ_2 at y^e_2 and γ_3 at y^e_3. If the supervisor plays this strategy infinitely often, then only cycles with nonnegative payoffs are formed in the supervised system shown in Figure 4. Compared with the original system in Figure 1, the cycles with a negative payoff are broken. Then it is easy to verify that the supervised system is live and all infinite runs have a positive limit mean payoff. Therefore S solves Problem 1.

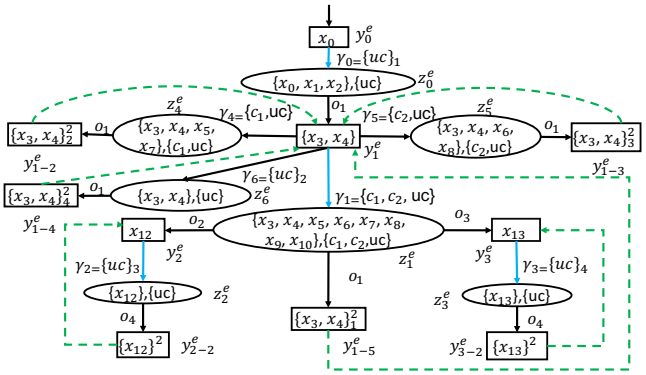

Fig. 3. The FCEIC$_w$ with dashed green lines connecting good leaf states with their subsumed states, Win_s is the set of all states

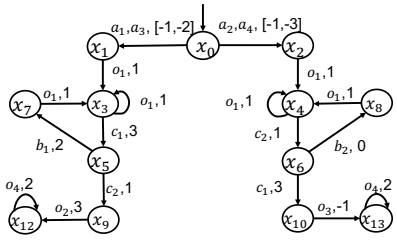

Fig. 4. A supervised system that satisfies the requirements of Problem 1

VI. Conclusion

We presented an approach for synthesizing partial observation supervisors that enforce nonnegative limit mean payoff of the system's long run behaviors. This work is the first one to investigate such a problem. To this end, we defined the energy information state and a novel bipartite information structure called First Cycle Energy Inclusive Controller (FCEIC). Based on the FCEIC, the problem was transformed into a finite reachability game with perfect information. We showed that the winning strategies for the supervisor in the FCEIC lead to valid solutions for the proposed problem.

References

[1] K. R. Apt and E. Grädel. *Lectures in game theory for computer scientists*. Cambridge University Press, 2011.

[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, 2015.

[3] C. G. Cassandras and S. Lafortune. *Introduction to discrete event systems – 2nd Edition*. Springer, 2008.

[4] K. Chatterjee, L. Doyen, T. A. Henzinger, and J.-F. Raskin. Algorithms for omega-regular games with imperfect information. In *Inter. Work. on Comp. Sci. Logic*, volume 6, pages 287–302. Springer, 2006.

[5] A. Degorre, L. Doyen, R. Gentilini, J.-F. Raskin, and S. Toruńczyk. Energy and mean-payoff games with imperfect information. In *Computer Science Logic*, pages 260–274. Springer, 2010.

[6] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8(2):109–113, 1979.

[7] A. Giua and C. Seatzu. A systems theory view of Petri nets. In *Advances in control theory and application*. Springer, 2007.

[8] P. Hunter, A. Pauly, G. A. Pérez, and J.-F. Raskin. Mean-payoff games with partial observation. *Theore. Computer Sci.*, 735:82–110, 2018.

[9] Y. Ji, X. Yin, and S. Lafortune. Opacity enforcement by insertion functions under energy constraints. In *Proc. of the 14th International Workshop on Discrete Event Systems*, pages 302–308, 2018.

[10] J. Komenda and T. Masopust. Computation of controllable and coobservable sublanguages in decentralized supervisory control via communication. *Discrete Event Dynamic Systems: Theory and Application*, 27(4):585–608, 2017.

[11] A. Levy. *Basic set theory*, volume 13. Courier Corporation, 2002.

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

[13] 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.

[14] V. Pantelic and M. Lawford. Optimal supervisory control of probabilistic discrete event systems. *IEEE Transactions on Automatic Control*, 57(5):1110–1124, 2012.

[15] K.M. Passino and P.J. Antsaklis. On the optimal control of discrete event systems. In *Proceedings of the 28th IEEE Conference on Decision and Control*, pages 2713–2718. IEEE, 1989.

[16] S. Pruekprasert and T. Ushio. Supervisory control of partially observed quantitative discrete event systems for fixed-initial-credit energy problem. *IEICE trans. on Info. and Systems*, 100(6):1166–1171, 2017.

[17] S. Pruekprasert, T. Ushio, and T. Kanazawa. Quantitative supervisory control game for discrete event systems. *IEEE Transactions on Automatic Control*, 61(10):2987–3000, 2016.

[18] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM journal on control and optimization*, 25(1):206–230, 1987.

[19] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Diagnosability of discrete-event systems. *IEEE Transactions on automatic control*, 40(9):1555–1575, 1995.

[20] R. Sengupta and S. Lafortune. An optimal control theory for discrete event systems. *SIAM Journal on con. and opt.*, 36(2):488–541, 1998.

[21] S. Shu and F. Lin. Supervisor synthesis for networked discrete event systems with communication delays. *IEEE Transactions on Automatic Control*, 60(8):2183–2188, 2015.

[22] R. Su, J. H. Van Schuppen, and J. E. Rooda. The synthesis of time optimal supervisors by using heaps-of-pieces. *IEEE Transactions on Automatic Control*, 57(1):105–118, 2012.

[23] S. Takai and T. Ushio. Effective computation of an L_m (G)-closed, controllable, and observable sublanguage arising in supervisory control. *Systems & Control Letters*, 49(3):191–200, 2003.

[24] X. Yin and S. Lafortune. Synthesis of maximally permissive supervisors for partially-observed discrete-event systems. *IEEE Transactions on Automatic Control*, 61(5):1239–1254, 2016.

[25] X. Yin and S. Lafortune. A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems. *IEEE Trans. on Automatic Control*, 61(8):2140–2154, 2016.