Games for Efficient Supervisor Synthesis

Daniel Hausmann, Prabhat Kumar Jha, and Nir Piterman

***Abstract*— In recent years, there has been an increasing interest in the connections between supervisory control theory and reactive synthesis. As the two fields use similar techniques there is great hope that technologies from one field could be used in the other. In this spirit, we provide an alternative reduction from the supervisor synthesis problem to solving Büchi games via games with a non-blocking objective. Our reduction is more compact and uniform than previous reductions. As a consequence, it gives an asymptotically better upper bound on the time complexity of the supervisory control synthesis problem. Our reduction also breaks a widely held belief about the impossibility of reducing the supervisory control synthesis problem to a game with a linear winning condition.**

Supervisory control; Game theory; Discrete event systems

I. Introduction

Supervisory control theory [1] provides a framework for reasoning about the control structure for transition systems that have some controllable and some uncontrollable components of behaviour. The desired behaviour of the plant is described as reachability to good states of plants, termed as *non-blocking* property. The control structure is implemented in the form of a supervisor that restricts the controllable events for the plant in order to yield the desired behaviour. We are interested in the synthesis of maximally permissive non-blocking supervisors.

In [1], an algorithm for synthesizing non-blocking supervisors has been described. The algorithm is based on repeated backward searches using language refinement of the generated language with the specification language. Each backward search can be computed in time $O(ne)$, where n is the number of states of the plant and e is the number of events. There can be at most n backward searches, so the overall worst-case runtime complexity of this algorithm is in $O(n^2e)$.

More or less at the same time that Ramadge and Wonham established the supervisory control theory [1], Pnueli and Rosner introduced the modern *reactive synthesis* [2]. Here, the control structure is an automaton and the specification is provided in some logic such as linear temporal logic (LTL), computation tree logic (CTL), etc. The two fields developed in isolation and with very little contact. However, it turns out that the techniques used are similar: iteration of backward and forward searches in graph structures. Practitioners in both fields started exploring the links between the two [3] with the hope of cross-fertilization.

Motivated by techniques used in [4] and [3], [5] considers the natural encoding of the supervisory control synthesis problem to a game, where the choices of the supervisor are encoded directly. Thus, their approach involves a reduction to a game graph that is of exponential size as all possible supervisor choices are encoded as separate game vertices. Technically, they are mostly interested in supervisory control where the specification is a language of infinite words and use a co-operative Büchi game to encode it. A fixpoint computation for the co-operative Büchi objective is used to solve the game. Due to the exponential size of the game graph, this reduction requires exponential time in the size of the event set.

In [3],[1] there is a more involved reduction to a game, which encodes the choices of the supervisor linearly. Thus, the construction results in a game with a graph of size $O(ne)$. In order to do that, special care has to be given to the connectivity of a state leading to three different encodings of states in the game depending on state properties. The winning condition is described as a non-linear condition, encoded in a branching time logic. The interest in [3] is mainly in establishing the connection between the two fields. However, using their approach to solve the supervisory control synthesis problem results in a runtime of $O(n^2e^3)$.

In this work, we provide a reduction from supervisor synthesis to solving Büchi games. While previous

This work is supported by the ERC consolidator grant D-SynMA (No. 772459) and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

D. Hausmann and N. Piterman are with the Department of Computer Science and Engineering, University of Gothenburg, Gothenburg, Sweden {hausmann|piterman}@chalmers.se

P. K. Jha is with the Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden prabhar@chalmers.se

[1]Note that, chronologically, the work of Ehlers et al. appears earlier.

approaches concentrated on the choice of the controller which events to allow, our approach concentrates on the choice of the plant whether to allow the controller to apply control at all. The result is a reduction where every state of the plant is represented by exactly two vertices in the game. In addition, the reduction is totally uniform and handles all states in the same way regardless of their properties. It follows that compared to [3], our reduction is more efficient and cleaner. In comparison to [5], our reduction (A) does not require cooperation of players in Büchi games and instead yields a conventional competitive Büchi game, and (B) gives a graph of size double of that of the plant compared to the exponential-sized graph obtained in the reduction mentioned above.

In addition, our reduction disproves the widely held belief (cf. [3], [6]) that the winning condition in non-blocking games cannot be expressed by a linear winning condition. Indeed, we show that the non-blocking winning condition is equivalent to a Büchi winning condition. The translation in [3] was intended to establish further connections between supervisory control and reactive synthesis. In this spirit, based on our reduction, we are able to use optimal analysis of Büchi games [7] to improve the upper bound on time for solving the supervisory control problem to $O(\max(n^2, ne))$.

II. Preliminaries

A. Supervisory Control

A plant models possible behaviors of autonomous systems. It is described using states and transitions between the states, occurring as the result of some event. Controllable events are controlled by the controller part of the system and uncontrolled events are not controlled by the system. Supervisors restrict the plant's controllable events. A *closed-loop system* $G \parallel S$ is a combination of a supervisor S and a plant G. It restricts the plant by considering only the controllable actions allowed by the supervisor. We are interested in maximally permissive and non-blocking supervisors. We define the set-up formally as follows.

A *plant* P is a 5-tuple $P = (X, x_0, X_m, E = E_c \uplus E_{uc}, \delta)$ where X is a set of states, $x_0 \in X$ is the initial state, $X_m \subseteq X$ is a set of marked states, E is a set of events partitioned into the set E_c of controllable events and the set E_{uc} of uncontrollable events, and $\delta : X \times E \rightharpoonup X$ is a partial transition function. For $x \in X$, we denote the plant P with x as the initial state by P_x. A *supervisor* for plant G is a function $S : E^* \to 2^{E_c}$, assigning sets $S(w) \subseteq E_c$ of permitted controllable events to finite words $w \in E^*$. Let ϵ denote the empty word. Given a plant G and a supervisor S, the induced *closed-loop system* is $G \parallel S = (X \times E^*, (x_0, \epsilon), X_m \times E^*, E, \delta')$, where δ' is defined by putting $\delta'((x, \sigma), e) = (\delta(x, e), \sigma e)$ for $x \in X$, $\sigma \in E^*$ and $e \in S(\sigma) \cup E_{uc}$ such that $\delta(x, e)$ is defined; otherwise, $\delta'((x, \sigma), e)$ is left undefined. We co-define the *transition closure* $\delta^*(x, w)$ and *run* $\bar{\delta}(x, w)$ of a plant on a word $w \in E^*$ from a state $x \in X$ inductively with $\delta^*(x, \epsilon) = \bar{\delta}(x, \epsilon) = x$, $\delta^*(x, we) = \delta(\delta^*(x, w), e)$ and $\bar{\delta}(x, we) = \bar{\delta}(x, w) \cdot \delta^*(x, we)$, where $e \in E$. We say that there is a *path* from x to x' if there is a word $w \in E^*$ such that $\delta^*(x, w) = x'$. A supervisor S is said to be *non-blocking* for G, if for all states (x, σ) in $G \parallel S$, if $\delta'^*((x_0, \epsilon), \sigma) = (x, \sigma)$, then there exists a marked state $x' \in X_m$ and a (possibly empty) word $\sigma' \in E^*$ such that $\delta'^*((x, \sigma), \sigma') = (x', \sigma\sigma')$. If there is a function $f : X \to 2^{E_c}$ such that for every word $w \in E^*$, $S(w) = f(\delta^*(x_0, w))$, then we say that S is *memoryless*. S is said to be *maximally-permissive non-blocking* for G if it is non-blocking and for all non-blocking supervisors S' for G and all words $\sigma \in E^*$, we have $S'(\sigma) \subseteq S(\sigma)$. The supervisory control problem is to construct a maximally-permissive non-blocking supervisor.

Definition II.1 (Supervisory control synthesis problem)**.** Given a plant G, construct the maximally-permissive non-blocking supervisor S for G, if it exists, or state "no" if it does not exist.

Lemma II.1 ([1])**.** *For every plant, if there exists a non-blocking supervisor then there exists a memoryless and maximally-permissive non-blocking supervisor.*

B. Two-player Games

We are going to reduce the problem of supervisory control synthesis to the solution of two-player games.

A *game arena* is a graph $\mathcal{G} = (V = V_c \uplus V_{uc}, \Delta \subseteq V \times V)$ where V_c denotes the set of vertices controlled by the controller, V_{uc} denotes the set of vertices not controlled by the controller, and Δ denotes transitions. A *play* is an infinite sequence of vertices $\pi : \mathbb{N} \to V$ such that $\forall i.(\pi(i), \pi(i+1)) \in \Delta$. An edge (v, v') *occurs* in play π if for some i we have $\pi(i) = v$ and $\pi(i+1) = v'$. A *non-blocking* game is played on a game arena in which a set $V_m \subseteq V$ of vertices is marked. The controller's objective is to keep marked vertices reachable. A *Büchi* game is played on a game arena in which a set $B \subseteq \Delta$ of edges is marked. The controller's objective is to use edges from B infinitely often. Notice that a plant has *states* and a game has *vertices*. The *unravelling* of a game arena $\mathcal{G}$ starting at vertex v_i is a labelled tree $T_{v_i} \subseteq v_i \cdot V^*$ such that $v_i \in T_{v_i}$ and for each node $n = w \cdot v_j \in T_{v_i}$ we have $n \cdot v_k \in T_{v_i}$ if and only if $(v_j, v_k) \in \Delta$. Given a game

arena $\mathcal{G} = (V = V_c \uplus V_{uc}, \Delta \subseteq V \times V)$, we define the *computation forest* $\mathcal{F} = \bigcup_{v_i \in V} T_{v_i}$ as the union of the unravellings with respect to all the different starting vertices. We say that $u \in \mathcal{F}$ is a *child* of $w \in \mathcal{F}$ if there is a vertex $v \in V$ such that $u = wv$. It follows that for $w \cdot v_1 \cdot v_2 \in \mathcal{F}$ such that $(v_1, v_2) \in \Delta$, $w \cdot v_1 \cdot v_2$ is a child of $w \cdot v_1$. We say that $u \in \mathcal{F}$ is a *descendant* of $w \in \mathcal{F}$ if there is a word $w' \in V^*$ such that $u = ww'$. A *strategy* for controller is a sub-forest F of the computation forest $\mathcal{F}$ of the game such that for all $u \in (V^* \cdot V_{uc}) \cap F$ if $v \in \mathcal{F}$ is a child of u then $v \in F$. A play π is *contained* in a strategy F if for every i we have $\pi(0), \ldots, \pi(i) \in F$. Given a node $wv \in F$ we say that wv is marked (in a non-blocking game) if $v \in V_m$. Strategy F is *winning* for the non-blocking objective if every unmarked node in the strategy has a descendant in F that is marked. Given a node $wv \in F$ and its child $wvu \in F$ we say that the edge (wv, wvu) is marked if $(v, u) \in B$. Strategy F is *winning* for the Büchi objective if every vertex in F has at least one child in F and every infinite play contained in F has infinitely many marked edges. A winning strategy F is said to be *maximal* if for all winning strategies F', we have $F' \subseteq F$. F is said to be *memoryless* if for all sequences $\sigma \in V^*$ and for all vertices $u, v \in V$, we have $\sigma uv \in F$ if and only if $uv \in F$ and $\sigma u \in F$.

Lemma II.2 (Existence and memorylessness of maximal strategies in non-blocking games)**.** *Given a non-blocking game, there is a memoryless and maximal (potentially empty) winning strategy.*

Proof: The proof is based on adaptation of Lemma II.1 for non-blocking games. Given a game arena $\mathcal{G} = (V = V_c \uplus V_{uc}, \Delta)$ and a set $V_m \subseteq V$ of marked vertices, we define the *plant induced by* $\mathcal{G}$ and starting in state $v_i \in V$ by $P_{v_i} = (X = V, x_0 = v_i, X_m = V_m, E = E_c \uplus E_{uc}, \delta)$, where $E_c = \{(v, v') \in \Delta \mid v \in V_c\}$ and $E_{uc} = \{(v, v') \in \Delta \mid v \in V_{uc}\}$ and δ is defined by putting $\delta(x, (x, y)) = y$ for $x \in V$ and $(x, y) \in \Delta$. In the case that there is no non-blocking supervisor for P_{v_i}, we put $\tau_{v_i} = \{\}$. Otherwise, let S_{v_i} be the maximally-permissive non-blocking and memoryless supervisor for P_{v_i} that exists by Lemma II.1, let $P_{v_i} \parallel S_{v_i} = (V \times E^*, (v_i, \epsilon), V_m \times E^*, E, \delta_i')$ be the induced closed-loop system and put $\tau_{v_i} = \{\bar{\delta}(v_i, w) \mid \exists v \in V \,.\, \delta_i'^*((v_i, \epsilon), w) = (v, w)\}$. The tree τ_{v_i} contains all sequences of vertices corresponding to paths originating from (v_i, ϵ) in $P_{v_i} \parallel S_{v_i}$. Notice that for all $w \in E^*$ and $(u, v) \in E$ we have $S_{v_i}(w \cdot (u, v)) = \{(v, v') \in E_c \mid \bar{\delta}(v_i, w).uvv' \in \tau_{v_i}\}$.

That is, for $v \in V_c$ the supervisor defines exactly the strategy and for $v \in V_{uc}$ the set is empty. This correspondence between τ_{v_i} and S_{v_i} implies that there is a path from (v_i, ϵ) to a marked vertex in the closed-loop system iff there is a path from v_i to a marked node in τ_{v_i}. Hence, S_{v_i} is non-blocking iff τ_{v_i} is winning. The same correspondence also provides that for w ending in a vertex in V_c we have $(u, v) \in S_{v_i}(w)$ iff $\bar{\delta}(v_i, w) \cdot v \in \tau_{v_i}$. Hence S_{v_i} is maximally-permissive iff τ_{v_i} is the maximal winning sub-tree of T_{v_i} which is the unravelling of the game arena with v_i as the starting vertex. Now consider $F = \bigcup_{v_i \in X} \tau_{v_i}$. Since each τ_{v_i} is the maximal winning sub-tree of the respective T_{v_i}, F is the maximal winning sub-forest of $\mathcal{F}$ i.e. maximal winning strategy.

It remains to show that F is memoryless. For each $v_i \in V$, let $f_{v_i} : X \to 2^{E_c}$ be a function such that $f_{v_i}(\bar{\delta}(v_i, w)) = S_{v_i}(w)$ for all $w \in E^*$; such a function exists since S_{v_i} is memoryless. Let us consider two vertices $u, v \in V$. We note that $v \in f_{v_i}(u) \iff \sigma u \in \tau_{v_i} \implies \sigma uv \in \tau_{v_i}$ from the definitions of τ_{v_i} and f_{v_i}. Let a vertex $v_j \in V$. Assume towards a contradiction that $f_{v_i} \neq f_{v_j}$. Then we construct a tree τ'_{v_i} by putting $v \in f_{v_i}(u) \cup f_{v_j}(u) \iff \sigma u \in \tau'_{v_i} \implies \sigma uv \in \tau'_{v_i}$ and $v_i \in \tau'_{v_i}$. We construct τ'_{v_j} analogously. We note that τ'_{v_i} and τ'_{v_j} are winning and either τ_{v_i} is a strict sub-tree of τ'_{v_i} or τ_{v_j} is a strict sub-tree of τ'_{v_j}. This contradicts maximality of τ_{v_i} and τ_{v_j}, implying that $f_{v_i} = f_{v_j}$. Now $uv \in F$ iff $v \in f_{v_i}(u)$ iff $\sigma u \in F \implies \sigma uv \in F$. Hence F is memoryless. ∎

Lemma II.3 (Existence of memoryless strategies for Büchi games [8])**.** *Given a game arena and a marked subset of edges B, if a vertex v is in any winning strategy for controller then there is a memoryless winning strategy F_B for controller such that $v \in F_B$.*

Definition II.2 (Winning region)**.** A vertex $v \in V$ is said to be in the *winning region* of controller if v is contained in some winning strategy. We denote the set of winning vertices as V_{win}.

Lemma II.4 ([7])**.** *The winning region in a Büchi game with n vertices can be computed in time $O(n^2)$.*

III. Games for Supervisory Control

In this section we give a reduction from supervisory control to non-blocking games and Büchi games. Interestingly, the reduction uses the same game arena for both cases based on an appropriate definition of marked vertices (for the non-blocking game) and of marked edges (for the Büchi game). As mentioned, our reduction is different from existing reductions by diverging from the natural paradigm of "supervisor needs to choose the controllable events to implement". Instead we base our reduction on "environment chooses when to allow supervisor to implement controllable events".

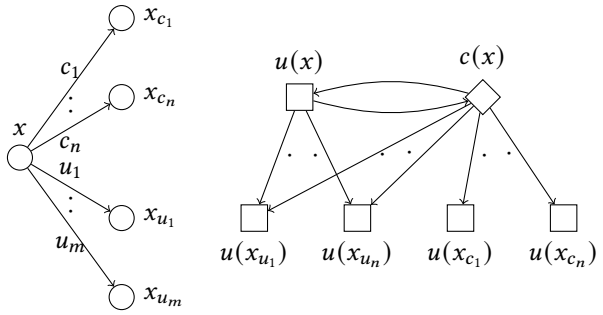

Fig. 1: Converting a state x of a plant to a pair of vertices $u(x)$ and $c(x)$ of a game

A. Plant to Game Arena

Next, we introduce the transformation that constructs a game arena from a plant.

Definition III.1. Given a plant $P = (X, x_0, X_m, E = E_c \uplus E_{uc}, \delta)$, we define the game arena $\mathcal{G}_P = (V = V_c \uplus V_{uc}, \Delta)$, where V is a set of vertices containing two vertices $c(x)$ and $u(x)$ for each state $x \in X$.

$$\begin{array}{rcl} V_c & := & \{c(x) \mid x \in X\} \qquad V_{uc} := \{u(x) \mid x \in X\} \\ \Delta & := & \{(u(x), c(x)), (c(x), u(x)) \mid x \in X\} \cup \\ & & \{(u(x), u(x')) \mid \exists e_u \in E_{uc} \,.\, \delta(x, e_u) = x'\} \cup \\ & & \{(c(x), u(x')) \mid \exists e \in E \,.\, \delta(x, e) = x'\} \end{array}$$

The construction of $\mathcal{G}_P$ is illustrated in Figure 1. We note that $|V| = 2|X|$. Given a run $\bar{\delta}(x, w) = x_1, \ldots x_n$ of P, where $x_1 = x$, let $seq(\bar{\delta}(x, w))$ be $u(x_1), c(x_1), u(x_2), c(x_2), \ldots, u(x_n), c(x_n)$.

We note that for an uncontrollable action u and a transition $\delta(x, u) = x'$ it is possible to follow u from x to x' in the game arena by either moving directly from $u(x)$ to $u(x')$ or by moving from $u(x)$ to $c(x)$ and from there to $u(x')$. The sequence $seq(\bar{\delta}(x, w))$ is the fixed sequence where we do the latter.

B. Supervisory Control to Non-Blocking Games

Next, we show that the supervisory control problem reduces to a non-blocking game over $\mathcal{G}_P$ with an appropriate set of marked vertices.

The non-blocking game $\mathcal{G}_{P_m}$ is defined over the game arena $\mathcal{G}_P$, using the set $V_m = \{c(x_m), u(x_m) \mid x_m \in X_m\}$ of marked vertices. We show that the maximal winning strategy for $\mathcal{G}_{P_m}$ yields the maximally-permissive non-blocking supervisor for plant P (if it exists).

Lemma III.1 (From plant to non-blocking game)**.** *The following are equivalent:*

1) *The vertex $u(x_0)$ is in the maximal winning strategy F_{max} in the non-blocking game $\mathcal{G}_{P_m}$.*
2) *The supervisor $S(w) = \{e_j \in E_c \mid seq(\bar{\delta}(x_0, we_j)) \in F_{max}\}$ is the maximally permissive non-blocking supervisor.*
3) *There exists a non-blocking supervisor for P.*

Proof: We begin by proving (1) $\implies$ (2). First we show that S is a non-blocking supervisor for P. In $P \,||\, S = (X \times E^*, (x_0, \epsilon), X_m \times E^*, E, \delta')$, consider a state $(x, w) \in X \times E^*$, such that $\delta'^*((x_0, \epsilon), w) = (x, w)$. We have to show that if (x, w) is unmarked then there is a word $w' \in E^*$ such that $\delta'^*((x, w), w')$ is marked. Since $seq(\bar{\delta}(x_0, w)) = \sigma u(x)$, we have that $u(x)$ is in F_{max} as F_{max} is memoryless. Then either $u(x) \in V_m$ or there is a word $\sigma u(x) \sigma' v_m \in V^*$ such that $v_m \in V_m$. In the first case, we have $x \in X_m$ and are done. In the second case, consider the word $w' \in E^*$ such that $seq(\bar{\delta}(x_0, w')) = \sigma u(x) \sigma' v_m$. Since $seq(\bar{\delta}(x_0, w)) = \sigma u(x)$, there is a word $w'' \in E^*$ such that $seq(\bar{\delta}(x, w'')) = \sigma' v_m$. Then we have $\delta'^*((x, w), w'') = (x'_m, w')$ such that either $v_m = u(x'_m)$ or $v_m = c(x'_m)$. In both cases, $x'_m \in X_m$ and hence (x'_m, w') is a marked state of $P \,||\, S$, as required. Hence S is a non-blocking supervisor.

Suppose S is not maximally permissive. Then let $S' \neq S$ be the maximally permissive non-blocking supervisor for P. There is a word $w \in E^*$ such that $S'(w) \neq S(w)$. Let $e_j \in E_c$ be an event such that $e_j \in S'(w)$ and $e_j \notin S(w)$. According to the definition of S, $seq(\bar{\delta}(x_0, we_j)) \notin F_{max}$. The definition of non-blocking supervisors and the fact that S' is a non-blocking supervisor implies that there exists $w' \in E^*$ such that $\delta'^*(\delta'^*((x_0, \epsilon), we_j), w')$ is marked. Hence $\sigma = seq(\bar{\delta}(x_0, we_j w')) \notin F_{max}$. Put $F' = F_{max} \cup \{\sigma' \in V^* | \exists \sigma'' \in V^* \,.\, \sigma = \sigma' \sigma''\}$. The strategy F' is winning as σ is marked (recall that σ is marked if the last vertex in σ is marked).

For (2) $\implies$ (3) we note that the maximally permissive non-blocking supervisor is in particular a non-blocking supervisor.

In order to prove (3) $\implies$ (1), we construct a winning strategy F for the game $\mathcal{G}_{P_m}$ from S. Note that constructing some winning strategy suffices since if $u(x_0)$ is in some winning strategy then $u(x_0)$ is in the maximal winning strategy.

Put $u(x_0) \in F$. For all $x_i \in X$ and all $(u(x_i), v) \in \Delta$ such that $w \cdot u(x_i) \in F$, we require $w \cdot u(x_i) \cdot v \in F$. For all $x_i \in X$, all $h \in E^*$ and all $e \in S(h)$ such that $\delta^*(x_0, h) = x_i$, we require that $w \cdot c(x_i) \cdot u(\delta(x_i, e)) \in F$ whenever $w \cdot c(x_i) \in F$. For all $x_i \in X$ and all $h \in E^*$ such that $S(h) = \varnothing$ and $\delta^*(x_0, h) = x_i$, we require that $w \cdot c(x_i) \cdot u(x_i) \in F$ whenever $w \cdot c(x_i) \in F$.

It remains to show that F is a winning strategy for the non-blocking objective. Since S is a non-blocking supervisor for P, from every unmarked vertex (x_j, σ)

that is reachable from (x_0, ϵ) there is a path to a marked vertex $(x_m, \sigma\sigma')$ in $P \mid\mid S$. By definition of closed-loop systems and by the construction of F, if there is a path from (x_0, ϵ) to (x_j, σ) in $P \mid\mid S$ then for $u(x_0)$ there is some descendant $w \cdot u(x_j) \in F$, where $w \in V^*$. Furthermore, if there is a path from (x_j, σ) to $(x_m, \sigma\sigma')$ then there is a descendant $w \cdot u(x_j) \cdot w' \cdot u(x_m) \in F$, where $w' \in V^*$ and $u(x_m) \in V_m$, as required.

Consider a node $w \cdot c(x) \in F$. By definition $w \cdot c(x)$ has a descendant $w \cdot c(x) \cdot u(x') \in F$. From the previous paragraph, it follows that some marked vertex is reachable from $u(x')$ and we are done. ∎

C. Supervisory Control to Büchi Games

We now show that the same game arena captures the same supervisor control problem but this time as a Büchi game. This goes against the widely held belief that the supervisory control synthesis problem is inherently non-linear [3], [6].

The Büchi game $\mathcal{G}_{P_B}$ is defined over the game arena $\mathcal{G}_P$ with the following set of marked edges:

$$B = \{(u(x), u(x')) \in \Delta\} \cup \{(u(x), c(x)), (c(x), u(x)) \mid x \in X_m\}$$

We show that the winning region for the non-blocking condition can be computed from the winning region for the Büchi condition.

Lemma III.2 (From non-blocking game to Büchi game)**.** *The following are equivalent:*

1) *The vertex $u(x_0)$ is in the maximal winning strategy F_{max} in the non-blocking game $\mathcal{G}_{P_m}$.*
2) *The vertex $u(x_0)$ is in the winning region V_{win} in the Büchi game $\mathcal{G}_{P_B}$.*

Proof: We prove (1) $\implies$ (2) by constructing a winning strategy F_B for the Büchi game from F_{max}. In order to construct F_B, we inductively define a distance function $d : F_{max} \to \mathbb{N} \cup \{0\}$ such that if $v \in V_m$ and $\sigma v \in F_{max}$, then $d(\sigma v) = 0$ and otherwise $d(\sigma) = \min(\{d(\sigma v) \mid \sigma v \in F_{max}, v \in V\}) + 1$.

Since every $\sigma \in F_{max}$ has a descendant that is marked, d is well-defined. We define F_B to contain $V \cap F_{max}$. Furthermore, we require that for all $v_{uc} \in V_{uc}$ and all $\sigma \in V^*$ such that $\sigma v_{uc} \in F_B$ and $\sigma v_{uc} v \in F_{max}$, we also have $\sigma v_{uc} v \in F_B$. For every vertex $v_c \in V_c$, if $\sigma v_c \in F_B$ and $d(\sigma v_c) > 0$, then for all $v \in V$ and $\sigma \in V^*$ such that $\sigma v_c v \in F_{max}$ and $d(\sigma v_c v) < d(\sigma v_c)$, we require $\sigma v_c v \in F_B$. For all $v_c \in V_c$ and $\sigma \in V^*$ such that $\sigma v_c \in F_B$ and $d(\sigma v_c) = 0$ and all $v \in V$ such that $\sigma v_c v \in F_{max}$ and $d(\sigma v_c v) = d(\sigma v_c)$, we require $\sigma v_c v \in F_B$. Denote $v_c = c(x)$ for some x, this is particularly the case for $u(x)$ that is marked as well.

To see that F_B is a winning strategy, let ρ be a play that is contained in F_B. We distinguish cases: (1) There are infinitely many $i \in \mathbb{N}$ such that $\rho(i) \in V_{uc}$ and $\rho(i+1) \in V_{uc}$. (2) There are finitely many $i \in \mathbb{N}$ such that $\rho(i) \in V_{uc}$ and $\rho(i+1) \in V_{uc}$. In case (1), ρ contains infinitely occurrences of marked edges as all edges of form $(u(x'), u(x''))$ are marked. In case (2), consider $j \in \mathbb{N}$ such that for every $i > j$, if $\rho(i) \in V_{uc}$ then $\rho(i+1) \notin V_{uc}$, that is, $\rho(i+1) \in V_c$. We have $d(\sigma u(x)) \geq d(\sigma u(x) c(x))$ since for every $\sigma u(x) v \in F_{max}$ such that $v \neq c(x)$, we also have $\sigma u(x) c(x) v \in F_{max}$ (due to F_{max} being the maximal strategy). From i on, the play ρ is of the shape $u(x_i), c(x_i), u(x_{i+1}), c(x_{i+1}), \ldots$, where the distance function d is by construction guaranteed to decrease on moves of the shape $(c(x_j), u(x_{j+1}))$ and does not increase on moves of the shape $(u(x_j), c(x_j))$. Therefore, the distance eventually becomes zero. Since $d(v) = 0$ only when v is marked, the play reaches a marked vertex and since every outgoing edge of the marked vertex is marked, the play has an occurrence of as many marked edges as many times d evaluates to 0. Since d is a non-increasing function if no consecutive pair of vertices are uncontrollable, it does not increase and has value 0 infinitely often so ρ contains infinitely many edges from B. We get that in both cases (1) and (2), the play has infinite occurrences of marked edges, hence F_B is winning for the game $\mathcal{G}_{P_B}$.

In order to show (2) $\implies$ (1), we construct a strategy F in $\mathcal{G}_{P_m}$ from the winning region V_{win} of $\mathcal{G}_{P_B}$. For every $v \in V_{win}$, we put $v \in F$. For all $u, v \in V_{win}$ such that $(u, v) \in \Delta$, we require $\sigma u v \in F$ whenever $\sigma u \in F$.

Since there is no vertex outside V_{win} in F, it suffices to prove that from every vertex $v \in V_{win}$, there is a descendant in F that is marked. Suppose there is a vertex $v' = u(x) \in V_{win}$ that has no marked descendant in F. Consider a winning Büchi strategy F_B such that $V_{win} \subseteq F_B$. By construction $F_B \subseteq F$. For all $x \in X$, $u(x) \in V_{uc}$, and $(u(x), c(x)) \in \Delta$, if $\sigma \cdot u(x) \in F_B$ then we have $\sigma \cdot u(x) \cdot c(x) \in F_B$. Consider a play π in F such that $\pi(1) = v'$ and for every $i \in \mathbb{N}, i \geq 1$, if $\pi(i) = u(x)$ for some $x \in X$ then $\pi(i+1) = c(x)$ and if $\pi(i) = c(x)$ for some $x \in X$ then $\pi(i+1) = v''$ such that $\pi(1)\pi(2)\ldots\pi(i)v'' \in F_B$. In case there are multiple candidates for v'', choose one of those. Then π is contained in F_B. Note that we choose π such that there is no i such that both $\pi(i)$ and $\pi(i+1)$ are in V_{uc}. In addition, from our assumption, v' has no marked descendant in F. It follows that there are no marked edges in π. Hence, π is not Büchi winning, yielding a contradiction. We conclude that the strategy F is a winning strategy. ∎

We note that the strategy F constructed in the second part of the above proof is maximal: Every larger strategy necessarily contains some play that leaves V_{win} which is in contradiction to the claim of the lemma.

Theorem III.3. *Given a plant $P = (X, x_0, X_m, E = E_c \uplus E_{uc}, \delta)$, a maximally-permissive and non-blocking supervisor, if it exists, can be constructed in time $O(max(n^2, ne))$ where $n = |X|$ and $e = |E|$.*

Proof: Using Lemma III.1, we convert a plant to a non-blocking game of twice the size of the plant. The construction of the maximally-permissive supervisor from the maximal strategy is described in the lemma. Lemma III.2 provides a reduction of a non-blocking game to a Büchi game with the same arena. The construction of the maximal strategy for the non-blocking game from the winning region of the Büchi game is given in the proof. Computing winning regions for Büchi games can be done in time $O(n^2)$ as per Lemma II.4. Hence the maximally-permissive supervisor can be constructed in time $O(n^2)$ from the description of the plant. The description of the plant is of size $|X| \cdot |E| = ne$ leading to overall time complexity in $O(\max(n^2, ne))$. ∎

D. A simple example

Here we describe a simple example of a plant with three states in Fig. 2, in which $X = \{x0, x1, x2\}, E_c = \{c1, c2, c3\}, E_{uc} = \{u1\}, X_m = \{x2\}$ and transitions are as depicted in Fig. 2. Using our reduction we construct a game arena as shown in Fig. 3. Vertices corresponding to marked vertex $x2$ i.e. $u(x2)$ and $c(x2)$ are marked for the non-blocking game. Accordingly for the Büchi game, Büchi edges are labelled with B as shown in Fig. 3. Vertices $u(x1)$ and $c(x1)$ are not winning for the controller due to the $u(x1) - c(x1)$ loop which has no Büchi edge. The winning region for the Büchi game, $\{u(x0), c(x0), u(x2), c(x2)\}$ is colored. A winning strategy for the Büchi game can be described as a function of choices from each controllable vertex in the winning region due to memorylessness of strategy. In this example, from vertex $c(x0)$ controller can take the edge $(c(x0), u(x2))$ and from vertex $c(x2)$ controller can take the edge $(c(x2), u(x0))$. From the winning region, we can obtain the maximally-permissive supervisor for the plant by allowing all controllable events which result in states corresponding to the winning region. Since this is memoryless it can also be described using a function from states to subsets of controllable events, i.e. $f(x0) = \{c2, c3\}$ and $f(x1) = f(x2) = \varnothing$ where the maximally-permissive supervisor is defined by $S(w) = f(\bar{\delta}(x0, w))$. Note that the edge corresponding to event $c3$ from state $x0$ in the game is not in the winning strategy for the Büchi game, however it is enabled by the maximally-permissive supervisor.

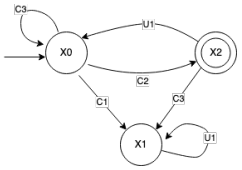

Fig. 2: A simple plant

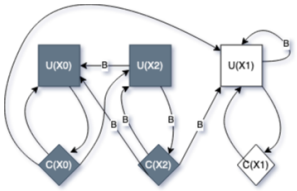

Fig. 3: Game corresponding to the plant in Fig. 2

IV. Conclusion

We have provided a reduction of the supervisor synthesis problem for terminating processes to solving Büchi games. Our reduction is more compact and uniform than previous reductions and establishes a better complexity bound for supervisor synthesis. It will be interesting to investigate whether the supervisory control problem for non-terminating processes [4] can be reduced to linear games as well. Another direction to extend this work is towards composition-based and modular synthesis of non-blocking supervisors.

References

[1] P.J. Ramadge and W.M. Wonham. The control of discrete event systems. *Proc. IEEE*, 77(1):81–98, 1989.

[2] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *POPL '89*, page 179–190, 1989. ACM.

[3] R. Ehlers, S. Lafortune, S. Tripakis, and M.Y. Vardi. Supervisory control and reactive synthesis: a comparative introduction. *Discret. Event Dyn. Syst.*, 27(2):209–260, 2017.

[4] R. Majumdar and A.-K. Schmuck. Supervisory controller synthesis for non-terminating processes is an obliging game. *IEEE Transactions on Automatic Control*, pages 385–392, 2022.

[5] A.-K. Schmuck, T. Moor, and K.W. Schmidt. A reactive synthesis approach to supervisory control of terminating processes. *IFAC-PapersOnLine*, 53(2):2149–2156, 2020. 21st IFAC World Congress.

[6] R. Ehlers. Personal communication.

[7] K. Chatterjee and M. Henzinger. An $O(n^2)$ time algorithm for alternating büchi games. In *SODA '12*, pages 1386–1399. SIAM, 2012.

[8] E.A. Emerson and C.S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS '91*, pages 368–377. IEEE Computer Society, 1991.