You Can Always Compute Maximally Permissive Controllers Under Partial Observation When They Exist

Sophie Pinchinat and Stéphane Riedweg
IRISA-INRIA, F-35042, Rennes, France
{Sophie.Pinchinat,Stephane.Riedweg}@irisa.fr

Abstract— **The maximal permissivity property of controllers is an optimal criterion that is often taken for granted as the result of synthesis algorithms: the algorithms are designed for frameworks where the existence and the uniqueness of a maximal permissive controller is demonstrated apart, as it fulfills sufficient hypotheses ; these algorithms precisely compute this object. Still, maximally permissive solutions might exist in circumstances which do not fall into such identified frameworks, but there is no way to ensure that the algorithms deliver an optimal solution. In this paper, we propose a general synthesis procedure which always computes a maximal permissive controller when it exists.**

I. Introduction

The maximal permissivity property in control theory is a natural notion meaning that the controller only disallos what must be disallowed ; in other words, the controller should be *minimal restrictive controller* as called in the pioneer work of [1], or equivalently *maximally permissive controller* as we qualify it here.

To our knowledge, the maximally permissive property of controllers has never been treated on its own in the literature. The major reason lies on the nature of supervisory control problems that have been investigated. Either one proves apart the existence and the uniqueness of a maximally permissive solution; this is the case of the Ramadge and Wonham regular languages framework [2] where the existence and uniqueness of a supremal controllable and observable language is established (see also [3]) ; even if it means adding stronger hypothesis than observability such as normality [4], [5]. However, a few attempt to compute a maximally permissive controller in less favorable contexts have been proposed ; see for example [6]. On the contrary, the considered framework does not ensure the existence of maximally permissive solutions, in general; their computation is then evaded. This is the case in branching-time logic frameworks [7], [8].

And yet, as we shown here, the maximally permissive property of controllers under partial observation is decidable, even in the most general framework where the control objective is any mu-calculus definable property [9]. Our approach, based on labeling processes, makes it possible to characterize the maximally permissive controllers as models accepted by some tree automaton, so that their existence (and synthesis when possible) comes down to the non-emptiness problem for the tree automaton.

The paper is organized as follows: Section II presents the framework and Section III focuses on maximal permissiveness issues with the general decision procedure.

II. Basics of Control Problems

Models of systems are operational: they are standard state machines called here *processes*, composed of states, transitions between states that are triggered by events.

We assume given a finite set of events $\mathtt{Ev} = \{\alpha, \beta, \theta, \upsilon \dots\}$, a set of atomic propositions $AP = \{a, b, c, c' \dots\}$ which denote properties of states.

Definition 1: **Processes.** Given two finite sets $\Sigma \subseteq \mathtt{Ev}$ and $\Gamma \subseteq AP$, a *process on* (Σ, Γ) is a tuple $\mathcal{S} = \langle S, s^0, t, L\rangle$, where S is the set of states, $s^0 \in S$ is the initial state, $t : S \times \Sigma \to S$ is the *transition function* - it is partial -, and $L : S \to 2^\Gamma$ is a labeling function, labeling states with propositions - it is total. The set Σ is called *the type of* $\mathcal{S}$.

When $t(s, \alpha) = s'$, the pair (s, s') is called an α-*transition*, and we say that an α-transition is *firable in* s.

A process $\mathcal{S}$ of type Σ is *complete* if for all $\alpha \in \Sigma$, an α-transition is firable in any state of $\mathcal{S}$. A process $\mathcal{S}$ is *finite* if S is finite. We will use $\mathcal{S}, \mathcal{P}, \mathcal{E}, \mathcal{E}'$, ... as typical elements for processes.

Processes can be combined: we use the *weak synchronous product*. Two processes $\mathcal{S}_1$ and $\mathcal{S}_2$ must synchronize on the occurrence of a shared event, namely an event in $\Sigma_1 \cap \Sigma_2$:

Definition 2: **(Weak) Synchronous Product.** The *(weak) synchronous product* of $\mathcal{S}_1 = \langle S_1, s_1^0, t_1, L_1\rangle$ of type Σ_1 and $\mathcal{S}_2 = \langle S_2, s_2^0, t_2, L_2\rangle$ of type Σ_2 is the process $\mathcal{S}_1 \otimes \mathcal{S}_2 = \langle S_1 \times S_2, (s_1^0, s_2^0), t, L\rangle$ on $(\Sigma_1 \cup \Sigma_2, \Gamma_1 \cup \Gamma_2)$, hence of type $\Sigma_1 \cup \Sigma_2$, where $t((s_1, s_2), \alpha) = (s_1', s_2')$ whenever

- $(\alpha \in \Sigma_1 \cap \Sigma_2)$ and $(s_1' = t_1(s_1, \alpha))$ and $(s_2' = t_2(s_2, \alpha))$, or
- $(\alpha \in \Sigma_1 \setminus \Sigma_2)$ and $(s_1' = t_1(s_1, \alpha))$ and $(s_2' = s_2)$, or
- $(\alpha \in \Sigma_2 \setminus \Sigma_1)$ and $(s_1' = s_1)$ and $(s_2' = t_2(s_2, \alpha))$.

Moreover, $L(s_1, s_2) = L_1(s_1) \cup L_2(s_2)$.

A state in a (weak) synchronous product is then a pair (s_1, s_2) of states each of which will be called *a local state*. Following this line, a *local transition* of a synchronous product is a transition firable in a local state.

The weak synchronous product gives the formal meaning to the notion of controller: given a process $\mathcal{P}$ - for "plant" - of type Σ, a subset $O \subseteq \Sigma$ of *observable* events, and a property ψ on processes - which will be made clear further.

A controller of $\mathcal{P}$ for ψ under observation O is some non-empty process $\mathcal{C}$ on $(O, \emptyset)$ with additional properties we explain now. First of all, the controlled plant is the process $\mathcal{P} \otimes \mathcal{C}$ and it satisfies ψ. The weak synchronous product realizes our intuition of how a controller should act, for the following reasons: in $\mathcal{P} \otimes \mathcal{C}$, a local transition of $\mathcal{P}$, labeled by a non-observable event (in $\Sigma \backslash O$), remains firable in the product since no synchronization with a transition of $\mathcal{C}$ is required. Hence, $\mathcal{C}$ cannot prevent non-observable events from occurring. The controller $\mathcal{C}$ can neither take the occurrence of a non-observable event into account as its local state does not change. However, the controller $\mathcal{C}$ might stop $\mathcal{P}$ from doing an θ-transition when $\theta \in O$: this is the case when a θ-transition is firable in a local state of $\mathcal{P}$ but is not firable in the local state of $\mathcal{C}$; we then say that $\mathcal{C}$ *disallows* this θ-transition.

Nevertheless, disallowing transitions is subject to additional constraints that respect the characteristic of events: the events set Σ of $\mathcal{P}$ splits into the *uncontrollable* events set $\Sigma_{uc} \subseteq \Sigma$, on the one hand, and the *controllable* events set Σ_c, by complementary in Σ, on the other hand. The set Σ_{uc} denotes particular events which cannot be controlled by an internal device of the plant, since for example they arise from the outside world; we use υ as a typical uncontrollable event. A transition labeled by an uncontrollable event are called *an uncontrollable transition*, otherwise it is called a *controllable transition*... Consequently, realistic controllers should disallow only transitions labeled by controllable events, in which case the controllers are qualified *admissible*. In the following, an admissible controller of $\mathcal{P}$ for ψ under observation O is simply called *a controller for* $(\mathcal{P}, O, \psi)$.

Finally, it is natural to require *maximally permissive controllers*, that is controllers which disallow only transitions that must be disallowed. This qualitative notion is formalized by the partial pre-order of *simulation*:

Definition 3: **Simulation.** Given two processes $\mathcal{S}_1 = \langle S_1, s_1^0, t_1, L_1 \rangle$ of type Σ_1 and $\mathcal{S}_2 = \langle S_2, s_2^0, t_2, L_2 \rangle$ of type Σ_2, a *simulation of $\mathcal{S}_1$ by $\mathcal{S}_2$* is a binary relation $\rho \subseteq S_1 \times S_2$ such that $(s_1^0, s_2^0) \in \rho$ and for any $(s_1, s_2) \in \rho$ and for any $\alpha \in \Sigma_1$ such that $t_1(s_1, \alpha)$ is defined, $t_2(s_2, \alpha)$ is also defined and $(t(s_1, \alpha), t_2(s_2, \alpha)) \in \rho$.

Notice that simulations do not take atomic propositions into account. We write $\mathcal{S}_1 \preceq \mathcal{S}_2$ whenever there exists a simulation of $\mathcal{S}_1$ by $\mathcal{S}_2$, and we use $\mathcal{S}_1 \prec \mathcal{S}_2$ when $\mathcal{S}_1 \preceq \mathcal{S}_2$ and not $\mathcal{S}_2 \preceq \mathcal{S}_1$. One easily checks that $\preceq$ is a pre-order which corresponding order is partial. Basically, when processes are associated their *execution tree*, that is, their (possibly infinite) unfolding, $\mathcal{S}_1 \preceq \mathcal{S}_2$ means that $\mathcal{S}_2$ has a larger execution tree than $\mathcal{S}_1$ has.

Definition 4: **Maximally Permissive Controllers Under Partial Observation.** Given a process $\mathcal{P}$ of type Σ, a set of observations $O \subseteq \Sigma$ and a property ψ, a controller $\mathcal{C}$ for $(\mathcal{P}, O, \psi)$ is *maximally permissive* if for any other controller $\mathcal{C}'$ of type O for $(\mathcal{P}, O, \psi)$, we do not have $\mathcal{P} \otimes \mathcal{C} \prec \mathcal{P} \otimes \mathcal{C}'$.

The rest of the paper is dedicated to the computation of a maximally permissive controller for $(\mathcal{P}, O, \psi)$ when it exists, when property ψ is definable in any branching-time temporal logic. For this, we consider the mu-calculus logic of [9], [10], over processes, which subsumes any other branching-time logics.

We first recall what the mu-calculus is. Assume given a set of variables $Var = \{X, Y, \dots\}$, which is used for fix-points formulas of the mu-calculus, written L_μ.

Definition 5: **Syntax of L_μ.** The syntax of the mu-calculus is standard [10]. It is defined by:
$\varphi, \varphi' (\ni L_\mu) ::= \top \mid a \mid X \mid \neg\varphi \mid \varphi \vee \varphi' \mid \ <\alpha>\varphi \mid \mu X.\varphi(X)$
where $\alpha \in \mathtt{Ev}$, $a \in AP$ and $X \in Var$.

Fix-points formulas $\mu X.\varphi(X)$ can be properly interpreted whenever each occurrence of X in $\varphi(X)$ is under an even number of negation symbols $\neg$. This is standard. Using the classical terminology of the mu-calculus, we name *sentences* all the formulas where each occurrence of a variable X is binded by a fix-point symbol μ. Following the standards, we write $\bot$, $\varphi \wedge \varphi'$, $[\alpha]\varphi$, $\nu X.\varphi(X)$ respectively for $\neg\top$, $\neg(\neg\varphi \vee \neg\varphi')$, $\neg <\alpha> \neg\varphi$, $\neg\mu X.\neg\varphi(\neg X)$, as well as $\xrightarrow{\alpha}$, $[\ ]\varphi$ and $\varphi \Rightarrow \varphi'$ respectively for $<\alpha>\top$, $\bigwedge_{\alpha \in \mathtt{Ev}} [\alpha]\varphi$ and $\neg\varphi \vee \varphi'$. For $\varphi \in L_\mu$, $\mathbf{AG}\varphi$ is a notation for $\nu X.[\ ]X \wedge \varphi$ (for the reader who is familiar with the logic CTL of Clarke and Emerson): according to Def.6 further, it states the *invariance* of φ, namely "from now on, the property φ always holds".

The logic can now be interpreted. Given $\mathcal{S} = \langle S, s^0, t, L \rangle$ a process on (Σ, Γ), a formula $\varphi \in L_\mu$ denotes a subset of states, those which "satisfy" it. The semantics is given by induction over φ, hence the need to interpret variable formulas $X \in Var$: a valuation $val : Var \rightarrow 2^S$ is set which defines the subset of states denoted by each variable formula.

Definition 6: **Semantics of L_μ.** Given a process $\mathcal{S} = \langle S, s^0, t, L \rangle$ on (Σ, Γ), and a valuation $val : Var \rightarrow 2^S$, the interpretation of an L_μ-formula φ, written $[\![\varphi]\!]_{\mathcal{S}}^{[val]}$, is a subset of S defined inductively on the structure of φ by:

$$[\![\top]\!]_{\mathcal{S}}^{[val]} = S \qquad [\![a]\!]_{\mathcal{S}}^{[val]} = \{s \in S \,|\, a \in L(s)\}$$
$$[\![X]\!]_{\mathcal{S}}^{[val]} = val(X) \quad [\![\neg\varphi]\!]_{\mathcal{S}}^{[val]} = S \setminus [\![\varphi]\!]_{\mathcal{S}}^{[val]}$$
$$[\![\varphi \vee \varphi']\!]_{\mathcal{S}}^{[val]} = [\![\varphi]\!]_{\mathcal{S}}^{[val]} \cup [\![\varphi']\!]_{\mathcal{S}}^{[val]}$$
$$[\![<\alpha>\varphi]\!]_{\mathcal{S}}^{[val]} = \{s \in S \,|\, \exists s', t(s,\alpha) = s', s' \in [\![\varphi]\!]_{\mathcal{S}}^{[val]}\}$$
$$[\![\mu X.\varphi(X)]\!]_{\mathcal{S}}^{[val]} = \bigcap\{V \subseteq S \,|\, [\![\varphi]\!]_{\mathcal{S}}^{[val(V/X)]} \subseteq V\}$$

As the interpretation of sentences is independent of the valuation val, we simply write $[\![\, \varphi \,]\!]_{\mathcal{S}}$, and write $\mathcal{S} \models \varphi$, read "$\mathcal{S}$ *satisfies* φ," whenever $s^0 \in [\![\, \varphi \,]\!]_{\mathcal{S}}$. With these conventions of notations, the assertions "$\mathcal{S} \models \varphi \vee \varphi'$", "$\mathcal{S} \models \neg\varphi$", ... respectively mean "$\mathcal{S} \models \varphi$ or $\mathcal{S} \models \varphi'$ ", "not $\mathcal{S} \models \varphi$" (also written "$\mathcal{S} \not\models \varphi$"), ...

From now on, assume given a process $\mathcal{P} = \langle P, p^0, t, L\rangle$ on (Σ, Γ), an observation set $O \subseteq \Sigma$, and a mu-calculus sentence ψ (with atomic propositions in Γ).

Maximally Permissive Control Under Partial Observation Problem

$\texttt{MP}(\mathcal{P}, O, \psi)$:*Given* $(\mathcal{P}, O, \psi)$*, does there exist a maximally permissive controller for* $(\mathcal{P}, O, \psi)$*, and if any, compute one ?*

In the next secttion, Theo.14 characterizes a maximally permissive controller for $(\mathcal{P}, O, \psi)$, and we derive an algorithm, according to the results of Theo.18.

III. Characterizing and Computing Maximally Permissive Controllers Under Partial Observation

Technically, controllers are represented in an extended form. They become complete processes by a kind of completion procedure: informally, a fresh atomic proposition c is chosen which labels original states of the controller, whereas new states, not labeled by c (hence labeled by $\neg c$) are added in order to make the result a complete process. Given a controller $\mathcal{C}$, we shall write $\mathcal{E}_{\mathcal{C}}$ the process obtained by the completion procedure. $\mathcal{E}_{\mathcal{C}}$ is called a *labeling process* which formal definition follows:

Definition 7: **Labeling Processes.** Given $c \in AP$, a *c-labeling process of type O* is a complete process $\mathcal{E} = \langle E, e^0, t, L\rangle$ on $(O, \{c\})$, with $L(e^0) = \{c\}$. We let Lab_c^O be the set of c-labeling processes, and we use $\mathcal{E}$, $\mathcal{E}'$, ... for typical elements. A *labeling of some process $\mathcal{S}$ by the proposition c*, or a *c-labeling of $\mathcal{S}$*, is a product $\mathcal{S} \otimes \mathcal{E}$ with $\mathcal{E} \in Lab_c^O$.

Basically, the process $\mathcal{S} \otimes \mathcal{E}$ is in general an unfolding of $\mathcal{S}$, with some states labeled by c.

The relationship between a controller $\mathcal{C}$ and the labeling process $\mathcal{E}_{\mathcal{C}}$, obtained by the completion procedure, is formalized by the notion of *pruning*:

Definition 8: **Pruning.** Given a labeling process $\mathcal{E} = \langle E, e^0, t, L\rangle$ on $(O, \{c\})$, the *c-pruning of $\mathcal{E}$*, written $\mathcal{E}_{\to c}$, is the restriction of $\mathcal{E}$ to the set of states $E \cap L^{-1}(\{c\})$. Namely, it is like $\mathcal{E}$ but only on its c-labeled part. Notice that because e^0 is labeled by c, process $\mathcal{E}_{\to c}$ is non-empty.

Now, for a controller $\mathcal{C}$, we have $\mathcal{E}_{\mathcal{C}\to c} = \mathcal{C}$. So that talking about controllers or talking about labeling processes is equivalent.

Theo.14 further relates the maximally permissive property with a single logical formula. The result relies on the *adjustment* of the sentence describing the control objective, see Def.9, and on a formula which ensures the admissibilty of the controller, as stated by Lem.12:

Definition 9: **Adjustment.** For all $\varphi \in L_\mu$ and $b \in AP$, the *b-adjustment of* φ is $\varphi_{\to b} \in L_\mu$, inductively defined by:

$$\begin{array}{|lll} \top_{\to b} = \top & a_{\to b} = a & X_{\to b} = X \\ (<\alpha>\varphi)_{\to b} = <\alpha>(b \wedge \varphi_{\to b}) & (\neg\varphi)_{\to b} = \neg\varphi_{\to b} \\ (\varphi \vee \varphi')_{\to b} = \varphi_{\to b} \vee \varphi'_{\to b} & (\mu X.\varphi)_{\to b} = \mu X.\varphi_{\to b} \end{array}$$

The fundamental result of Lem.10, easily proved by induction on the structure of the sentence, means that the b-adjustment of a sentence φ holds of a process whenever φ holds of the process derived by keeping only states labeled by b:

Lemma 10: Given a process $\mathcal{S}$ on (Σ, Γ), a sentence φ and $b \in \Gamma$, $\mathcal{S}_{\to b} \models \varphi$ if and only if $\mathcal{S} \models \varphi_{\to b}$.

Applying Lem.10 to the control problem $(\mathcal{P}, O, \psi)$, we get the following:

Lemma 11: Given a fresh proposition $c \notin \Gamma$ and a labeling process $\mathcal{E} \in Lab_c^O$, $\mathcal{P} \otimes \mathcal{E}_{\to c} \models \psi$ if and only if $\mathcal{P} \otimes \mathcal{E} \models \psi_{\to c}$.

Proof: Observe that process $\mathcal{P} \otimes \mathcal{E}_{\to c}$ is isomorphic to process $(\mathcal{P} \otimes \mathcal{E})_{\to c}$ and use Lem.10. ■

Let us write $\texttt{Admissible}(c)$ the sentence

$$\mathbf{AG}(c \Rightarrow \bigwedge_{v \in \Sigma_{uc}} [v]c)$$

$\mathcal{P} \otimes \mathcal{E} \models \texttt{Admissible}(c)$ tells that $\mathcal{E}_{\to c}$ is indeed a controller: in particular, in $\mathcal{P}$ no uncontrollable transition is disallowed by the controller $\mathcal{E}_{\to c}$.

For convenience let us simply write $\texttt{Controller}(c, \psi)$ instead of $\texttt{Admissible}(c) \wedge \psi_{\to c}$. Now, $\mathcal{P} \otimes \mathcal{E}$ satisfying $\texttt{Controller}(c, \psi)$ gives a logical manner to express that $\mathcal{E}_{\to c}$ is a controller for $(\mathcal{P}, O, \psi)$:

Lemma 12: Let $\mathcal{E} \in Lab_c^O$. $\mathcal{P} \otimes \mathcal{E} \models \texttt{Controller}(c, \psi)$ if and only if $\mathcal{E}_{\to c}$ is a controller for $(\mathcal{P}, O, \psi)$.

Proof: Assume $\mathcal{E}_{\to c}$ is a controller for $(\mathcal{P}, O, \psi)$; hence $\mathcal{P} \otimes \mathcal{E}_{\to c} \models \psi$, which entails $\mathcal{P} \otimes \mathcal{E} \models \psi_{\to c}$, by Lem.11. Moreover, because $\mathcal{E}_{\to c}$ is admissible, no uncontrollable transition of $\mathcal{P}$ is disallowed by $\mathcal{E}_{\to c}$, hence, in $\mathcal{P} \otimes \mathcal{E}$, the target states of any uncontrollable transition necessarily are labeled by c (for this transition to be kept after pruning by c) ; $\mathcal{P} \otimes \mathcal{E}$ therefore satisfies the invariant property $\mathbf{AG}(c \Rightarrow \bigwedge_{v \in \Sigma_{uc}} [v]c)$, that is $\texttt{Admissible}(c)$.

Reciprocally, let $\mathcal{E}$ be such that $\mathcal{P} \otimes \mathcal{E} \models \texttt{Controller}(c, \psi)$. By assumption, because $\mathcal{E} \in Lab_c^O$, $\mathcal{E}_{\to c}$ has type O; it is also non-empty. According to Lem.11, $\mathcal{P} \otimes \mathcal{E}_{\to c} \models \psi$. It remains to show that $\mathcal{E}_{\to c}$ is admissible: let (p, v, p') be an uncontrollable transition of $\mathcal{P}$ and let (p, e) be a global state of $\mathcal{P} \otimes \mathcal{E}$ with e labeled by c, hence so is (p,e). For $\mathcal{E}_{\to c}$ to be admissible, we must exhibit a local state $e' \in E$ s.t. $((p,e),v,(p',e'))$ is a transition in $\mathcal{P}\otimes\mathcal{E}$ and e' is labeled by c. *De facto*, $((p,e),v,(p',e'))$ will also be a transition of the controlled plant $\mathcal{P}\otimes\mathcal{E}_{\to c}$. If $v \notin O$, it is obviously the case by taking $e' = e$. Otherwise ($v \in O$), because $\mathcal{E}$ is complete on O, the local transition (e,v,e') exists; e' must be labeled by c since $\mathcal{P}\otimes\mathcal{E} \models \mathbf{AG}(c \Rightarrow \bigwedge_{v\in\Sigma_{uc}}[v]c)$, which concludes. ∎

We now turn to permissiveness issues.

Let us write $c \sqsubset c'$ for $c \sqsubseteq c' \wedge \neg(c' \sqsubseteq c)$ where $c \sqsubseteq c'$ is a notation for the invariant property $([\,]\mathbf{AG}(c'))_{\to c}$.

$c \sqsubseteq c'$ simply means the following: when interpreted on some process $\mathcal{S}$ (w.l.o.g. with all states reachable), $\mathcal{S} \models c \sqsubseteq c'$ whenever $[\![c]\!]_{\mathcal{S}} \subseteq [\![c']\!]_{\mathcal{S}}$. That is, the label c covers less states than the label c' does. Reformulated in terms of labeling processes, we get Lem.13:

Lemma 13: For any $\mathcal{E} \in Lab_c^O$ and $\mathcal{E}' \in Lab_{c'}^O$ $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \models c \sqsubset c'$ if and only if $\mathcal{P}\otimes\mathcal{E}_{\to c} \prec \mathcal{P}\otimes\mathcal{E}'_{\to c'}$.

Proof: Actually, it is enough to show that: $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \models c \sqsubseteq c'$ if and only if $\mathcal{P}\otimes\mathcal{E}_{\to c} \preceq \mathcal{P}\otimes\mathcal{E}'_{\to c'}$.

Assume $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \models c \sqsubseteq c'$. By definition of labeling processes, $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'$ is some unfolding of $\mathcal{P}$ labeled by both c and c', in a such a way that any state labeled by c is necessarily also labeled by c'. Clearly, the c-pruning of $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'$ is a sub-process of the c'-pruning of $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'$. Hence $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'_{\to c} \preceq \mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'_{\to c'}$. We conclude by observing the following: since $\mathcal{E}'$ is complete, $(\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}')_{\to c}$ is isomorphic to $\mathcal{P}\otimes\mathcal{E}_{\to c}$, up to the atomic proposition c', and $(\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}')_{\to c'}$ is isomorphic to $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}'_{\to c'}$, up to the atomic proposition c.

The converse follows the same reasonning backward ∎

We can now state the fundamental result that:

Theorem 14: **Maximally Permissive Controllers Under Partial Observation.**

Let $\mathcal{E} \in Lab_c^O$. $\mathcal{E}_{\to c}$ is a solution of $\texttt{MP}(\mathcal{P},O,\psi)$ if and only if for all $\mathcal{E}' \in Lab_{c'}^O$,

$$\begin{array}{ll}\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' & \models \texttt{Controller}(c,\psi) \\ & \wedge(\texttt{Controller}(c',\psi) \;\Rightarrow\; \neg(c \sqsubset c')).\end{array}$$

Proof: Assume $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \models \texttt{Controller}(c,\psi) \wedge (\texttt{Controller}(c',\psi) \Rightarrow \neg(c \sqsubset c'))$. Since $\mathcal{E}'$ is complete and not labeled by c, from $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \models \texttt{Controller}(c,\psi)$ we get $\mathcal{P}\otimes\mathcal{E} \models \texttt{Controller}(c,\psi)$. By Lem.12, this is equivalent to say that $\mathcal{E}_{\to c}$ is a controller for $(\mathcal{P},O,\psi)$. $\mathcal{E}_{\to c}$ is moreover maximally permissive: indeed, consider a controller $\mathcal{C}'$ for $(\mathcal{P},O,\psi)$, and its corresponding labeling process $\mathcal{E}_{\mathcal{C}'}$, obtained by using proposition c'. By Lem.12, $\mathcal{P}\otimes\mathcal{E}_{\mathcal{C}'} \models \texttt{Controller}(c',\psi)$, which entails $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}_{\mathcal{C}'} \models \texttt{Controller}(c',\psi)$, as $\mathcal{E}$ is complete and not labeled by c'. Now by assumption, necessarily $\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}_{\mathcal{C}'} \models \neg(c \sqsubset c')$. Lem.13 then involves not $\mathcal{P}\otimes\mathcal{E}_{\to c} \prec \mathcal{P}\otimes\mathcal{E}_{\mathcal{C}'\to c'}$. Because $\mathcal{E}_{\mathcal{C}'\to c'}$ is simply $\mathcal{C}'$, we conclude.

The converse follows the same line. ∎

We now investigate a decision procedure $\texttt{MP}(\mathcal{P},O,\psi)$. By Theo.18, the solutions of $\texttt{MP}(\mathcal{P},O,\psi)$ coincide with the models of some *parity automaton*. As already shown by [11], parity automata can actually be exploited to characterize the controllers; moreover, the controller synthesis reduces to the computation of a model of some parity automata. Moreover, computing a model of a parity automata is known to be decidable since [12], see also [10].

We first recall what *parity (tree) automata* are:

Definition 15: **Parity Tree Automata.** A *parity tree automaton* on (Σ,Γ) (with $\Sigma \subseteq \texttt{Ev}$ and $\Gamma \subseteq AP$) is a tuple $\mathcal{A} = \langle Q, Q^{\exists}, Q^{\forall}, q^0, \delta, r\rangle$ where Q is a finite set of states partitioned into two subsets $Q^{\exists}$ and $Q^{\forall}$ of *existential* and *universal* states, $q^0 \in Q$ is the initial state, *the transition function* δ which assigns to each state q and each subset of Γ an a set of pairs in $((\Sigma\cup\{\varepsilon\})\times Q)\cup(\Sigma\times\{\to,\not\to\})$. Formally,

$$\delta : Q \times 2^{\Gamma} \to 2^{((\Sigma\cup\{\varepsilon\})\times Q)\cup(\Sigma\times\{\to,\not\to\})}$$

Finally, $r : Q \to \mathbb{N}$ is the *parity condition*.

In the following, we simply say "automaton" instead of "parity tree automaton".

The automata semantics is *parity games*, introduced by [12]. We refer to [13] for a survey on parity games, and we give here a short presentation before Def.17.

A *parity game* is a directed graph G with a partition (V_I, V_{II}) of the vertices, an initial vertex v^0, and a partial mapping r from the vertices to a given finite set of integers; when it is convenient, the edges of the graph can have labels, as it is the case of Def.17. The game involves two players Player I and Player II.

A *play from some vertex* v of G proceeds as follows: if $v \in V_I$, then Player I is the current player and she chooses a successor vertex v' in the graph G, otherwise ($v \in V_{II}$) Player II chooses a successor vertex v'. The play goes on now from vertex v'. This way, a play can define an infinite sequence of vertices or a finite sequence ending in a current vertex which has no successor in the graph G. The *play is winning for player I* if either it is finite and ends up in a vertex of V_{II}, or if it is infinite and the upper bound of the set of ranks $r(v_i)$ of vertices v_i infinitely often covered in the play is even.

A *strategy for Player I* is a function σ assigning to every sequence $\vec{v}$ of vertices ending in a vertex of V_I, a successor vertex. A strategy σ is a *memoryless* if $\sigma(\vec{v})$ only depends on the last vertecs of $\vec{v}$. *A play follows a strategy* σ for Player I if each choice of Player I in any vertex $v \in V_I$ is $\sigma(\vec{v})$; in general, a given strategy σ is followed by several plays as the choice of Player II is left free by σ. A strategy σ for Player I is *winning (for Player I)* if each play from the initial vertex following σ is winning for Player I. Winning strategies for Player II are defined similarly. The fundamental result of parity games is the *Memoryless Determinacy Theorem*, established in [12] (see also [10]).

Theorem 16: **Memoryless determinacy [12].** In any parity game G, one of the two players has a (memoryless) winning strategy.

As announced earlier, the automata semantics is parity games: given a process $\mathcal{S}$, we say that $\mathcal{S}$ is *accepted by* $\mathcal{A}$ whenever there exists a winning strategy in the corresponding *(acceptance) parity game* $G(\mathcal{A},\mathcal{S})$ which depends on both $\mathcal{S}$ and $\mathcal{A}$.

Definition 17: **(Acceptance) Parity Game** $G(\mathcal{A},\mathcal{S})$**.** For $\mathcal{A}=\langle Q,Q^{\exists},Q^{\forall},q^0,\delta,r\rangle$ an automaton on (Σ,Γ) and $\mathcal{S}=\langle S,s^0,t,L\rangle$ a process on (Σ,Γ), *the (acceptance) parity game* $G(\mathcal{A},\mathcal{S})$ is defined as follows: V_I is $Q^{\exists}\times S\cup\{\perp\}$, V_{II} is $Q^{\forall}\times S\cup\{\top\}$, and v^0 is (q^0,s^0). The remaining of the graph $G(\mathcal{A},\mathcal{S})$ is defined incrementally. Vertices $\top$ and $\perp$ have no successor. For any vertex (q,s) and any $\alpha\in\Sigma$,

- there is an ε-edge from (q,s) to a successor vertex (q',s) if $(\varepsilon,q')\in\delta(q,L(s))$,
- there is an α-edge from (q,s) to a successor vertex (q',s') if $(\alpha,q')\in\delta(q,L(s))$ and $t(s,\alpha)=s'$,
- there is an α-edge from (q,s) to $\top$, if $(\alpha,\rightarrow)\in\delta(q,L(s))$ and $t(s,\alpha)$ is defined, or $(\alpha,\not\rightarrow)\in\delta(q,L(s))$ and $t(s,\alpha)$ is not defined,
- there is an α-edge from (q,s) to $\perp$, if $(\alpha,\rightarrow)\in\delta(q,L(s))$ and $t(s,\alpha)$ is not defined, or $(\alpha,\not\rightarrow)\in\delta(q,L(s))$ and $t(s,\alpha)$is defined.

Formally, the automaton $\mathcal{A}$ *accepts* $\mathcal{S}$, written $\mathcal{S}\in\mathcal{L}(\mathcal{A})$, if there exists a winning strategy for Player I in $G(\mathcal{A},\mathcal{S})$.

We recall that the non-emptiness of parity automata and the computation of a model, if any, are decidable problems, as showed [12] with an optimal algorithm. When the size of $\mathcal{A}$ is n, deciding and computing a model (if any) $\mathcal{S}$ of $\mathcal{A}$ is done in time $2^{n^{O(1)}}$.

We can now state the theorem to obtain the decision procedure for $\mathtt{MP}(\mathcal{P},O,\psi)$, since computing a model of a parity automata is decidable:

Theorem 18: There exists an automaton $A_{(\psi,\mathcal{P})}$ on $(O,\{c\})$ s.t. $\mathcal{E}\in\mathcal{L}(A_{(\psi,\mathcal{P})})$ if and only if $\mathcal{E}\in Lab^O_c$ and for all $\mathcal{E}'\in Lab^O_{c'}$,

$$\begin{aligned}\mathcal{P}\otimes\mathcal{E}\otimes\mathcal{E}' \quad &\models \mathtt{Controller}(c,\psi)\\ &\wedge(\mathtt{Controller}(c',\psi)\ \Rightarrow\ \neg(c\sqsubset c')).\end{aligned}$$

Proof: (Sketch) We construct $\mathcal{A}_{(\psi,\mathcal{P})}$ in tree steps.

First, we construct the automaton $\mathcal{A}_1$ on $(\Sigma,\Gamma\cup\{c,c'\})$ equivalent to the mu-calculus formula $\mathtt{Controller}(c,\psi)\wedge(\mathtt{Controller}(c',\psi)\ \Rightarrow\ \neg(c\ \sqsubset\ c'))$: the construction is standard (see [10] for example).

Next, in the same spirit as [8] but for the weak synchronous product, we construct the automaton $\mathcal{A}_1 /\!\!/ \mathcal{P}$ on $(O,\{c,c'\})$ s.t. for any *complete* process $\mathcal{S}$ on $(O,\{c,c'\})$, $\mathcal{S}\in\mathcal{L}(\mathcal{A}_1 /\!\!/ \mathcal{P})$ if and only if $\mathcal{P}\otimes\mathcal{S}\in\mathcal{L}(\mathcal{A}_1)$; the construction of $\mathcal{A}_1 /\!\!/ \mathcal{P}$ is given in the Appendix, see Def.20.

Lastly, we construct $A_{(\psi,\mathcal{P})}$, on the basis of $\mathcal{A}_1 /\!\!/ \mathcal{P}$, as in [11] - the construction uses complementation and projection of automata -, so that $\mathcal{E}\in\mathcal{L}(A_{(\psi,\mathcal{P})})$ if and only if $\mathcal{E}\in Lab^O_c$ and $\forall\mathcal{E}'\in Lab^O_{c'}$, $\mathcal{E}\otimes\mathcal{E}'\in\mathcal{L}(\mathcal{A}_1 /\!\!/ \mathcal{P})$. ■

Corollary 19: The problem $\mathtt{MP}(\mathcal{P},O,\psi)$ is decidable and the proposed algorithm has complexity 2EXP-TIME($|\psi|\times|\mathcal{P}|$).

Proof: According to Theo.14 and Theo.18, we can decide $\mathtt{MP}(\mathcal{P},O,\psi)$ by computing the non-emptiness of the parity automaton $A_{(\psi,\mathcal{P})}$. The standard algorithm for non-emptiness in addition delivers a model (if any) which achieves the controller synthesis; as a model $\mathcal{E}$ is computed, the controller is $\mathcal{E}_{\rightarrow c}$.

It is well known that the size of $\mathcal{A}_1$ is $|\psi|$. By Def.20, $\mathcal{A}_1 /\!\!/ \mathcal{P}$ has size $n_2=|\psi|\times|\mathcal{P}|$, and according to [11], $A_{(\psi,\mathcal{P})}$ has size $2^{(|\psi|\times|\mathcal{P}|)^{O(1)}}$.

Finally, the non-emptiness of $A_{(\psi,\mathcal{P})}$ is computed in time $2^{2^{(|\psi|\times|\mathcal{P}|)^{O(1)}}}$ which concludes. ■

IV. Conclusion

We have presented a uniform approach to compute maximally permissive controllers, when they exist, in the framework of deterministic systems under partial observation with mu-calculus definable behaviors.

To our knowledge no such vast result has ever been established in the literature, as it subsumes existing works in more restrictive frameworks, such as regular languages, and it clarifies the notion of maximally permissive controllers in the branching-time setting.

We insist on the elegance of the approach which uses the weak synchronous product to combine the plant and the controller in order to treat partial observation. We believe the framework is a lot more natural than those based on strong synchronous product, where "unobservation" is represented by state-looping transitions in the controller, as in [8], although equivalent in essence.

As shown here, the existence of maximally permissive controllers is decidable, and by extending the approach, their uniqueness can be decided as well: it suffices to require the equality of the labels c and c' in the key formula of Theo.14.

The decision procedure gives an upper bound to the complexity of the Maximally Permissive Control Problem. As shown by Theo.18, this complexity is independent of O, the observation events set. Whether this complexity is optimal is still an open problem, but it can surely be improved for the restricted case of total observation (when $O=\Sigma$): indeed, as shown in [14], the Maximally Permissive Control Problem for total observation reduces to model-check a formula which does not depend on the plant $\mathcal{P}$, hence its polynomial complexity in the size of the plant. On the other hand, the presented decision procedure for partial observation relies on the non-emptiness decision of an automaton which size depends on the plant, hence doubly exponential in the size of the plant.

APPENDIX

Details on $\mathcal{A}_1 /\!\!/ \mathcal{P}$ for Theorem 18:

Assume the process is $\mathcal{P} = \langle P, p^0, t, L\rangle$ and the automaton is $\mathcal{A}_1 = \langle \Gamma \cup \{c, c'\}, Q, Q^\exists, Q^\forall, q^0, \delta, r\rangle$.

Definition 20: $\mathcal{A}_1 /\!\!/ \mathcal{P}$ is the automaton on $\{c, c'\}$ which set of states is $(Q \times P) \cup \{\top, \bot\}$, with respectively $(Q^\exists \times P) \cup \{\bot\}$ and $(Q^\forall \times P) \cup \{\top\}$ its existential and universal sets, which initial state is (q^0, p^0) and which parity for any state $(q, p) \in Q \times P$ is $r(q)$. Finally, its transition function is $\delta_{/\!\!/} : ((Q \times P) \cup \{\top, \bot\}) \times 2^{\{c,c'\}} \to 2^{(Q\times P)\cup\{\top,\bot\}}$ defined by: for any $(q, p) \in Q \times P$, $l \subseteq \{c, c'\}$ and any $\alpha \in \Sigma$,

- $(\varepsilon, \top) \in \delta_{/\!\!/}((q, p), l)$ whenever
$$\begin{cases} (\alpha, \to) \in \delta(q, L(p) \cup l) \text{ and } t(p, \alpha) \text{ defined.} \\ \quad \text{or} \\ (\alpha, \not\to) \in \delta(q, L(p) \cup l) \text{ and } t(p, \alpha) \text{ undefined.} \end{cases}$$
- $(\varepsilon, \bot) \in \delta_{/\!\!/}((q, p), l)$ whenever
$$\begin{cases} (\alpha, \to) \in \delta(q, L(p) \cup l) \text{ and } t(p, \alpha) \text{ undefined} \\ \quad \text{or} \\ (\alpha, \not\to) \in \delta(q, L(p) \cup l) \text{ and } t(p, \alpha) \text{ defined.} \end{cases}$$
- $(\alpha, (q', p')) \in \delta_{/\!\!/}((q, p), l)$ whenever $\alpha \in O$ and $(\alpha, q') \in \delta(q, L(p) \cup l)$ and $t(p, \alpha) = p'$.
- $(\varepsilon, (q', p')) \in \delta_{/\!\!/}((q, p), l)$ if and only if
$$\begin{cases} (\varepsilon, q') \in \delta(q, L(p) \cup l) \text{ and } p' = p \\ \quad \text{or} \\ (\alpha, q') \in \delta(q, L(p) \cup l) \text{ and } t(p, \alpha) = p' \text{ and } \alpha \notin O \end{cases}$$

We now establish the correction of Def.20:

Proposition 21: For any complete process $\mathcal{S}$ on $(O, \{c, c'\})$, $\mathcal{S} \in \mathcal{L}(\mathcal{A}_1 /\!\!/ \mathcal{P})$ if and only if $\mathcal{P} \otimes \mathcal{S} \in \mathcal{L}(\mathcal{A}_1)$.

Proof: Assume $\mathcal{S}$ is (S, s^0, t, L). Note that we freely use t and L for $\mathcal{S}$, as for $\mathcal{P}$, and later for $\mathcal{P} \otimes \mathcal{S}$ as well, but it is clear in what follows -.

Consider the two parity games $G = G(\mathcal{A}_1, \mathcal{P} \otimes \mathcal{S})$ and $G_{/\!\!/} = G(\mathcal{A}_1 /\!\!/ \mathcal{P}, \mathcal{S})$. There is an obvious one-to-one correspondence between the positions $(q, (p, s))$ in G and the positions $((q, p), s)$ in $G_{/\!\!/}$; the positions also have the same parity value $r(q)$. Moreover, by associating the positions $\top$ and $\bot$ in G respectively to all the positions $(\top, s)$ and $(\bot, s)$ in $G_{/\!\!/}$, we show that any play in G has a counterpart as a play in $G_{/\!\!/}$; the converse holds as well and its proof uses the same arguments. This way, the existence of a winning strategy holds for both games exactly at the same time, hence $\mathcal{S} \in \mathcal{L}(\mathcal{A}_1 /\!\!/ \mathcal{P})$ if and only if $\mathcal{P} \otimes \mathcal{S} \in \mathcal{L}(\mathcal{A}_1)$.

Let $(q, (p, s))$ be a position of G, we study each possible move from $(q, (p, s))$ in G.

If there is an ε-edge from $(q, (p, s))$ to $\top$. By Def.17, we obtain **Condition (1)**:
$$\begin{cases} (\alpha, \to) \in \delta(q, L(p, s)) \text{ and } t((p, s), \alpha) \text{ is defined} \\ \quad \text{or } (\alpha, \not\to) \in \delta(q, L(p, s)) \text{ and } t((p, s), \alpha) \text{ is undefined} \end{cases}$$

Since $\mathcal{S}$ is complete on O, "$t((p, s), \alpha)$ is defined" is equivalent to "$t(p, \alpha)$ is defined". Hence Condition (1) is equivalent to **Condition (2)**:
$$\begin{cases} (\alpha, \to) \in \delta(q, L(p) \cup L(, s)) \text{ and } t(p, \alpha) \text{ is defined,} \\ \quad \text{or } (\alpha, \not\to) \in \delta(q, L(p) \cup L(s)) \text{ and } t(p, \alpha) \text{ is undefined.} \end{cases}$$

By Def.20, Condition (2) entails $(\varepsilon, \top) \in \delta_{/\!\!/}((q, p), L(s))$, which shows an ε-edge from $(q, (p, s))$ to $(\top, s)$ in the game $G_{/\!\!/}$.

In the same manner, we can show that the presence an ε-edge from $(q, (p, s))$ to $\bot$ in G implies the presence an ε-edge from $(q, (p, s))$ to $(\bot, s)$ in $G_{/\!\!/}$,

If now there is an ε-edge from $(q, (p, s))$ to some $(q', (p', s'))$, then $p' = p$ and $s' = s$. We can show that $(\varepsilon, (q', p)) \in \delta_{/\!\!/}((q, p), L(s))$, and necessarily there is an ε-edge from $((q, p), s)$ to $(q', (p, s))$ in $G_{/\!\!/}$.

If there is an α-edge, where $\alpha \in O$, from $(q, (p, s))$ to some $(q', (p', s'))$, then $(\alpha, q') \in \delta(q, L(p) \cup L(s), t(p, \alpha) = p'$, and $t(s, \alpha) = s'$. By definition of $\delta_{/\!\!/}$, $(\alpha, (q', p')) \in \delta_{/\!\!/}((q, p), L(s))$, and because moreover $t(s, \alpha) = s'$, there is an α-edge from $((q, p), s)$ to $((q', p'), s')$ in $G_{/\!\!/}$,

If there is an α-edge from $(q, (p, s))$ to $(q', (p', s'))$, where $\alpha \notin O$, then $(\alpha, q') \in \delta(q, L(p) \cup L(s))$ and $s = s'$. By definition of $\delta_{/\!\!/}$, we can show that there is an ε-edge from $((q, p), s)$ to $((q', p'), s)$ in $G_{/\!\!/}$. ■

REFERENCES

[1] P. J. Ramadge and W. Wonham, "Supervisory control of a class of discrete event processes," *SIAM Journal of Control and Optimization*, vol. 25, no. 1, pp. 206–230, 1987.

[2] Ramadge and Wonham, "The control of discrete event systems," in *Proc. of the IEEE*, vol. 77(1), 1989, pp. 81–98.

[3] Thistle and Wonham, "Control of of infinite behavior of finite automata," *SIAM Control and Optimization*, vol. 32(4), pp. 1075–1097, 1994.

[4] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya, "Supervisory control of discreteevent processes with partial observations," *IEEE Transactions on Automatic Control*, vol. 33, no. 3, pp. 249–260, 1988.

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

[6] S. Takai and T. Ushio, "Effective computation of an lm(g)-closed, controllable, and observable sublanguage arising in supervisory control," in *Workshop on Discrete Event Systems, Zaragoza, Spain*, October 2002.

[7] O. Kupferman, P. Madhusudan, P. S. Thiagarajan, and M. Y. Vardi, "Open systems in reactive environments: Control and synthesis," in *Proc. 11th Int. Conf. on Concurrency Theory*, ser. LNCS, vol. 1877. Springer-Verlag, 2000, pp. 92–107.

[8] Arnold, Vincent, and Walukiewicz, "Games for synthesis of controllers with partial observation," *Theoretical Computer Science*, vol. 1, pp. 7–34, 2003.

[9] Kozen, "Results on the propositional μ-calculus," *Theoretical Computer Science*, vol. 27(3), pp. 333–354, 1983.

[10] A. Arnold and D. Niwinski, *Rudiments of mu-calculus*. North-Holland, 2001.

[11] Riedweg and Pinchinat, "Quantified mu-calculus for control synthesis," in *Mathematical Foundations of Computer Science*, ser. LNCS, vol. 2747, 2003, pp. 642–651.

[12] E. A. Emerson and C. S. Jutla, "Tree automata, mu-calculus and determinacy," in *Proceedings 32nd Annual IEEE Symp. on Foundations of Computer Science, FOCS'91, San Jose, Puerto Rico, 1–4 Oct 1991*. Los Alamitos, California: IEEE Computer Society Press, 1991, pp. 368–377.

[13] J. Mycielski, "Games with perfect information," in *Handbook of Game Theory*, R. Aumann and S. Hart, Eds. Elsevier Science Publisher, 1992, vol. 1, pp. 41–70.

[14] S. Riedweg and S. Pinchinat, "Maximally permissive controllers in all contexts," in *Workshop on Discrete Event Systems*, Reims, France, sep 2004.