Optimal Assumptions for Synthesis

Romain Brenguier[*]

University of Oxford, UK
`rbrengui@cs.ox.ac.uk`

Abstract

Controller synthesis is the automatic construction a correct system from its specification. This often requires assumptions about the behaviour of the environment. It is difficult for the designer to identify the assumptions that ensures the existence of a correct controller, and doing so manually can lead to assumptions that are stronger than necessary. As a consequence the generated controllers are suboptimal in terms of robustness. In this work, given a specification, we identify the weakest assumptions that ensure the existence of a controller. We also consider two important classes of assumptions: the ones that can be ensured by the environment and assumptions that restricts only inputs of the systems. We show that optimal assumptions correspond to strongly winning strategies, admissible strategies and remorse-free strategies depending on the classes. Using these correspondences, we then propose an algorithm for computing optimal assumptions that can be ensured by the environment.

1998 ACM Subject Classification D.2.4 Software/Program Verification

Keywords and phrases Controller synthesis, Parity games, Admissible strategies

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2016.8

1 Introduction

The goal of synthesis is the implementation of a correct reactive system from its specifications. A specification is given by a ω-regular language over input and output signals of the desired system. It is realisable if we can guarantee that the sequence of inputs and outputs belong to the language. For regular languages this can be done using finite memory and thus implemented using Moore machines. Several tools have been developed to solve this problem (see for example: [3, 14]).

In general, the realisation of a specification requires some assumptions about the environment. In this work, we look for the weakest assumption that makes it realisable. For us, an assumption is weaker if it allows more behaviours. We therefore use language inclusion to compare assumptions, which means there may be incomparable and there is not a unique weakest in general. Apart from looking for just an assumption, we also consider two important classes of assumptions: ensurable and input assumptions. *Ensurable assumptions* can be ensured by the environment; in other term they cannot be falsified by a strategy of the controller. These assumptions are natural to consider when faced with a reactive environment. On the other hand, *input assumptions* are independent of the sequence of output that is produced. They are better suited to ensurable assumptions when the behaviour of the environment does not depend on the outputs of our system.

Synthesis is in general achieved by the computation of winning strategies in a game. For instance, if the specification is given by a parity automaton, we can see it has a game where the controller chooses output symbols and the adversary controls input symbols. Winning

[*] Work supported by EPSRC grant EP/M023656/1.

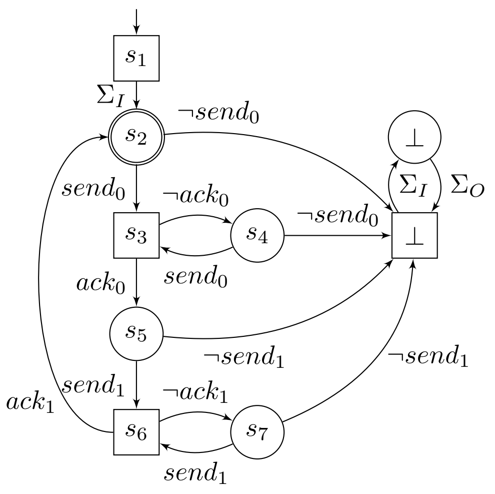

Figure 1 A Büchi automaton for the specification $\Sigma_I \cdot (send_0 \cdot (\neg ack_0 \cdot send_0)^* \cdot ack_0 \cdot send_1 \cdot (\neg ack_1 \cdot send_1)^* \cdot ack_1)^\omega$. Accepting states (with colour 0) are double lined. Square states mean that the next signal is an input, while circles mean it will be an output.

strategies in this game correspond to correct implementation of the system, and their existence implies realisability. When winning strategies do not exist, different classes have been introduced to characterise strategies that make their best effort to win. In particular, *strongly winning strategy* [10] play a winning strategy as soon as the current history (sequence of signals seen so far) makes it possible. *Admissible strategies* [1] are not *dominated* by other ones, in the sense that no strategy performs better than them against all adversary strategies. *Remorse-free strategies* [8] are such that no other strategy performs better than them against all *words* played by the adversary. We draw a link between classes of assumptions and these classes of strategies.

Example. As an example, assume we want to design a sender on a network where packets can be lost or corrupted, and our goal is to obtain a protocol similar to the classical alternating bit protocol. The outputs of the sender are actions $send_0$ and $send_1$, and the environment controls ack_0, ack_1 corresponding to acknowledgement of good reception of the packet. The specification is given by the ω-regular expression: $\Sigma_I \cdot (send_0 \cdot (\neg ack_0 \cdot send_0)^* \cdot ack_0 \cdot send_1 \cdot (\neg ack_1 \cdot send_1)^* \cdot ack_1)^\omega$. Intuitively, we have to send message with bit control 0 until receiving the corresponding acknowledgement, then do the same thing with the next message with bit control 1 and repeat this forever.

Although the implementation of the protocol seems straightforward, classical realisability fails here since if all packets are lost after some point the specification will not be satisfied, hence there is no winning strategy. To ensure realisability we have to make the assumption that a packet that is repeatedly sent will eventually be acknowledged. An admissible strategy for this specification can be implemented by a Moore machine which has the same structure as the automaton in Figure 1 with output function G such that $G(s_2) = G(s_4) = send_0$ and $G(s_5) = G(s_7) = send_1$. This implementation is natural for the given specification and corresponds indeed to the alternating bit protocol. The assumption corresponding to this strategy is the language recognised by the same automaton where we add $\bot$-states to the set of accepting states. As we will see in Theorem 23, it is an optimal ensurable assumption for the specification.

Table 1 Link between classes of assumptions and strategies.

Class of assumption:	Optimal achieved by:	
General ($\mathcal{A}$)	Strongly winning strategies	Theorem 12
Ensurable ($\mathcal{E}$)	Admissible strategies	Theorem 23
Input ($\mathcal{I}$)	Remorsefree strategies	Theorem 38

Scenarios. Specifications disallow behaviours that are not desirable. Dually, we may want to specify execution scenarios that should be possible in the synthesised system. We ask then for a system whose outcomes are all within the specifications and contains all the given scenarios. Scenarios are also a means for the user to provide feedback when the assumptions and strategies suggested by our algorithm are not the expected ones.

Generalisation. Sometimes, we already know a sufficient assumption but we want to synthesise a system which is as robust as possible by generalising this assumption. For instance, for the alternating bit protocol we could suggest as an initial assumption that two successive packets cannot be lost. With this assumption, we would offer no guarantee when than more two packets in a row are lost. By generalising the assumption, we ensure that the strategy synthesised works well under the assumption and for as many input sequences as possible. For instance, the alternating bit protocol works if an infinite number of packets are not lost.

Contribution. In this article we establish correspondences between class of assumptions and classical classes of strategies, which are summarised in Table 1.

We also show existence of optimal assumptions in most cases and give algorithms to compute optimal assumptions. In particular, we show the following properties. The existence of sufficient input assumptions compatible with a scenario is always true (Theorem 6). It is also true for safety assumptions if the scenario is itself a safety language (Theorem 7). There may exist an infinite number of optimal and ensurable-optimal assumptions (Theorem 14) and of input-optimal assumptions (Theorem 34). We can compute an optimal ensurable assumption in exponential time for parity specification and in polynomial time if we have an oracle to solve parity games (Theorem 27 and Theorem 30). There is an exponential algorithm that given a sufficient assumption, generalises it to an ensurable-optimal assumption (Theorem 33).

Comparison on previous works on assumptions for synthesis. In [7], the study is focused on safety conditions defined by forbidding edges of the automaton defining specification L. This approach depends on the choice of the automaton representing L, while ours does not. In the setting of [7], comparison between assumptions is based on the number of edges, while we compare them based on language inclusion which we find more relevant. Consider the example of Figure 2 taken from [7]. There is no winning strategy from s_1. According to [7], there are 2 minimal sufficient assumptions which are to remove either (s_3, s_6) or (s_5, s_7). However if we remove the edge from s_3 to s_6, state s_5 is no longer reachable which means that the first assumption is in fact stronger than the second one from the point of view of language inclusion. Moreover, we show that even for this restricted class of safety assumption, the claim that there is a unique non-restrictive optimal assumption [7, Theorem 5] does not hold for our notion of ensurable assumption. Consider the example of Figure 3: removing (s_3, s_6) or (s_4, s_6) is sufficient for $L = \Sigma_I \cdot (o_1 \cdot i_1 + o_2 \cdot i_2) \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$, and both these assumptions are ensurable-optimal.

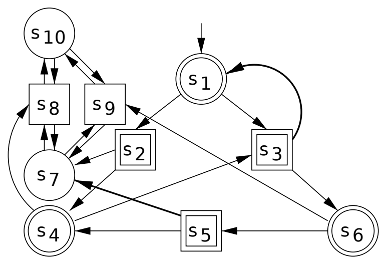

Figure 2 A game from [7].

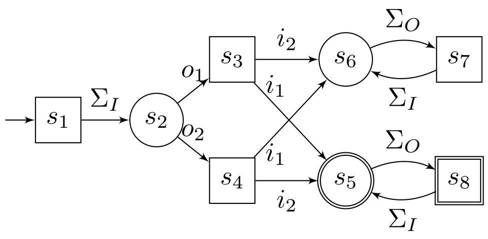

Figure 3 Automaton for specification $\Sigma_I \cdot (o_1 \cdot i_1 + o_2 \cdot i_2) \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$.

Other related works. Our goal is close to the work on assume-guarantee synthesis [6]. However assume-guarantee synthesis relies on equilibria concepts inspired by Nash equilibria. As such they assume rationality of the environment of the system and that its objective is known. By contrast, here we do not assume a rational environment and we look for the minimal assumptions about it. Closer to our work is [8] which relies on a notion of dominant strategy to obtain the weakest input assumption for which a component of the system can be implemented. A problem with this approach is that dominant strategies do not always exist. Here we characterise all minimal assumptions, and we look both at input assumptions and ensurable assumptions, which are more relevant in the context of a reactive environment.

2 Preliminaries

Given a finite alphabet Σ and an infinite word $w \in \Sigma^\omega$, we use w_i to denote the i-th symbol of w, and $w_{\leq i} = w_1 \cdots w_i$ the finite prefix of w of length i. We write $|w_{\leq i}| = i$ its length. A reactive system reads *input signals* in a finite alphabet Σ_I and produces *output signals* in a finite alphabet Σ_O. We fix these alphabets for the remainder of this paper. A *specification* of a reactive system is an ω-language $L \subseteq (\Sigma_I \cdot \Sigma_O)^\omega$. A *program* or *strategy* is a mapping $\sigma_\exists \colon (\Sigma_I \cdot \Sigma_O)^* \cdot \Sigma_I \to \Sigma_O$. An *outcome* of such a strategy $\sigma_\exists$ is a word w such that for all $i \in \mathbb{N}$, $w_{2\cdot i+2} = \sigma_\exists(w_{\leq 2\cdot i+1})$. We write $\mathsf{Out}(\sigma_\exists)$ for the set of outcomes of $\sigma_\exists$.

Given a specification L, the realizability problem [15] asks whether there exists a strategy $\sigma_\exists$ such that $\mathsf{Out}(\sigma_\exists) \subseteq L$. Such a strategy is said *winning* for L. The process of constructing such a strategy is called *synthesis*.

Parity automata. We assume that specifications are given by deterministic parity automata, which can recognise any ω-regular languages [13]. A *parity automaton* is given by $\langle S, s_0, \Delta, \chi\rangle$, where S is a finite set of states, $s_0 \in S$ is the initial state, $\Delta \in S \times (\Sigma_I \cup \Sigma_O) \times S$ is the transition relation, and $\chi \colon S \to \mathbb{N}$ is a colouring function. A path $\rho \in S^\omega$ is accepting if the smallest colour seen infinitely often is even (i.e. if $\min\{c \mid \forall i \in \mathbb{N}.\ \exists j \geq i.\ \chi(w_j) =$

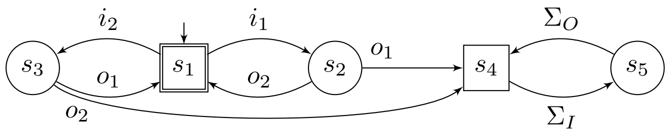

Figure 4 A Büchi automaton corresponding to specification $(i_1 \cdot o_2 + i_2 \cdot o_1)^\omega$.

$c\} \in 2 \cdot \mathbb{N})$. A word w is accepted if there is an accepting path whose labelling is w. A *Büchi automaton* is a parity automaton for which $\chi(S) \subseteq \{0, 1\}$. A *safety automaton* is a Büchi automaton where states of colour 1 are absorbing. The language recognised by an automaton is the set of words it accepts. Specification can also be given in temporal logics such as LTL before being translated to an automaton representation. In some examples, we will use LTL formulas with the syntax $\mathtt{X}\phi$ meaning ϕ holds in the next state, $\phi_1 \mathtt{U} \phi_2$ meaning ϕ_1 holds until ϕ_2 holds (and ϕ_2 must hold at some point), $\mathsf{F}\phi := \mathtt{true}\ \mathtt{U}\ \phi$ and $\mathsf{G}\phi := \neg\mathsf{F}(\neg\phi)$.

Strategies. The realizability problem is best seen as a game between two players [12]. The environment chooses the input signals and the controller the output signals. We therefore also define the concept of *environment-strategy* which is a mapping $\sigma_\forall \colon (\Sigma_I \cdot \Sigma_O)^* \to \Sigma_I$. Given an environment-strategy $\sigma_\forall$, we write $\mathsf{Out}(\sigma_\forall)$ the set of words w such that for all $i \in \mathbb{N}$, $w_{2\cdot i+1} = \sigma_\forall(w_{\le 2\cdot i})$. Given an input word $u \in {\Sigma_I}^\omega$, we write $\mathsf{Out}(\sigma_\exists, u)$ the unique outcome such that for all $i \in \mathbb{N}$, $w_{2\cdot i+1} = u_{i+1}$ and $w_{2\cdot i+2} = \sigma_\exists(w_{\le 2\cdot i+1})$. We also write $\mathsf{Out}(\sigma_\exists, \sigma_\forall) = \mathsf{Out}(\sigma_\exists) \cap \mathsf{Out}(\sigma_\forall)$, note that it contains only one outcome. A finite prefix of an outcome is called a *history*. Given a history h, we write $\mathsf{Out}_h(\sigma_\exists)$ a word w such that for all $i \le |h|$, $w_i = h_i$ and for all i such that $2\cdot i + 2 > |h|$, $w_{2\cdot i+2} = \sigma_\exists(w_{\le 2\cdot i+1})$. We write π_I and π_O the *samplings* over input and output signals respectively, that is $\pi_I \colon (\Sigma_I \cdot \Sigma_O)^\omega \to {\Sigma_I}^\omega$ is such that $\pi_I(w)_i = w_{2\cdot i-1}$ and $\pi_O \colon (\Sigma_I \cdot \Sigma_O)^\omega \to {\Sigma_O}^\omega$ is such that $\pi_O(w)_i = w_{2\cdot i}$.

Moore machine. Finite memory strategies are implemented as Moore machines. Note that the machines as we will define them read both inputs and outputs. In many application, reading outputs is unnecessary since the strategy can be left undefined on incompatible histories. However the definition we provide is coherent with our definition of strategies and makes it easier to combine strategies which may not be compatible with the same histories. A *Moore machine* is given by $\langle S_I, S_O, s_0, \delta, G \rangle$ where $S = S_I \cup S_O$ is a finite set of states, S_I is a set of input states and S_O of output states, $s_0 \in S_I$ is the initial state, $\delta \colon S \times (\Sigma_I \cup \Sigma_O) \to S$ is the transition function, and $G \colon S_O \to \Sigma_O$ is an output function. A Moore machine implements a strategy $\sigma_\exists$ where for all history $h \in (\Sigma_I \cdot \Sigma_O)^* \cdot \Sigma_I$, $\sigma_\exists(h) = G(s_{|h|})$ where for all $0 \le i < |h|$, $s_{i+1} = \delta(s_i, h_{i+1})$.

2.1 Assumptions

An assumption $A \subseteq (\Sigma_I \cdot \Sigma_O)^\omega$ is *sufficient* for specification L if there is a strategy $\sigma_\exists$ of the controller such that any outcome either satisfies L or is not in A, i.e. $\mathsf{Out}(\sigma_\exists) \cap A \subseteq L$. In this case we also say that A is *sufficient for* $\sigma_\exists$. We look for assumptions that are the least restrictive. We say that assumption A is *less restrictive* than B if $B \subseteq A$. We say it is *strictly less restrictive* if $B \subset A$ (i.e. $B \subseteq A$ and $A \neq B$). We consider the following classes:

- An *assumption* is an ω-regular language $A \subseteq (\Sigma_I \cdot \Sigma_O)^\omega$. We write the class of all assumptions $\mathcal{A}$.

- An *input-assumption* is an assumption which concerns only inputs and does not restrict outputs. We write this class $\mathcal{I} = \{A \in \mathcal{A} \mid \forall w, w' \in (\Sigma_I \cdot \Sigma_O)^\omega.\ \pi_I(w) = \pi_I(w') \land w \in A \Rightarrow w' \in A\}$.
- An *ensurable assumption* is an assumption for which the environment has a winning strategy, i.e. for each $w \in (\Sigma_I \cdot \Sigma_O)^*$, if $w \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap A \neq \varnothing$ then there exists an environment strategy $\sigma_\forall$ such that $w \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap \mathsf{Out}(\sigma_\forall) \neq \varnothing$ and $\mathsf{Out}(\sigma_\forall) \subseteq A$. We write this class $\mathcal{E}$. The fact that the environment can ensure the assumption is often a requirement in synthesis (see for instance [2]).
- An *output-restrictive assumption* A is an assumption which it restricts the strategies of the controller, that is there is $w \in (\Sigma_I \cdot \Sigma_O)^*$ and $\sigma_\exists$ a strategy, such that: $A \cap w \cdot (\Sigma_I \cdot \Sigma_O)^\omega \neq \varnothing$ and $\mathsf{Out}(\sigma_\exists) \cap w \cdot (\Sigma_I \cdot \Sigma_O)^\omega \neq \varnothing$ and $A \cap \mathsf{Out}(\sigma_\exists) \cap w \cdot (\Sigma_I \cdot \Sigma_O)^\omega = \varnothing$. Intuitively an output-restrictive assumption A forbids strategy $\sigma_\exists$, so playing $\sigma_\exists$ would be a trivial way to satisfy $A \Rightarrow L$. From the point of view of synthesis it is better to avoid such assumptions. We write this class $\mathcal{R}$.
- A *safety assumption* is an assumption A for which every word not in A has a bad prefix [11], i.e. $A = (\Sigma_I \cdot \Sigma_O)^\omega \setminus \{w \mid \exists k.\ w_{\leq 2 \cdot k} \in \mathsf{Bad}(A)\}$, where $\mathsf{Bad}(A) = \{h \in (\Sigma_I \cdot \Sigma_O)^* \mid h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap A = \varnothing\}$ is the set of bad prefixes of A. We write this class $\mathcal{S}$.

For a class $\mathcal{C}$ of assumptions, we say that assumption A is *$\mathcal{C}$-optimal* for L if A belongs to $\mathcal{C}$, is sufficient for L and there is no assumption $B \in \mathcal{C}$ that is strictly less restrictive than A and sufficient for L.

▶ Remark. Note that L is always a sufficient assumption, however it is too strong and will never be interesting for synthesis: if we assume that our specification always hold then any strategy would do. That is why we ask for assumptions that are as weak as possible.

We first note the following relationships between the classes of assumptions.

▶ **Lemma 1.** *Non-empty input assumptions are ensurable, i.e.* $(\mathcal{I} \setminus \{\varnothing\}) \subset \mathcal{E}$

▶ **Lemma 2.** *Ensurable assumptions are the non-output-restrictive ones:* $\mathcal{E} = \mathcal{A} \setminus \mathcal{R}$.

▶ **Example 3.** In all the examples of this article we will assume that the set of input signals is $\Sigma_I = \{i_1, i_2\}$ and the set output of output signals is $\Sigma_O = \{o_1, o_2\}$. The automaton for specification L given by formula $o_1 \mathsf{U} i_1$ is represented in Figure 5. This specification is not realisable, however several assumptions can be sufficient for it. Consider for instance the assumption A given by the LTL formula $\mathsf{F} o_1$. It is sufficient for L and is in fact sufficient for any specification since a strategy $\sigma_\exists$ which never plays o_1 has no outcome in A. To avoid this degenerate assumptions we focus on non-restrictive assumptions: $\mathsf{F} o_1$ is indeed output restrictive. On the other hand $\mathsf{F}(o_1) \Leftrightarrow \mathsf{F}(i_1)$ is ensurable because the environment can react to make the assumption hold, no matter the strategy $\sigma_\exists$ we chose. We can also check that it is sufficient for L: the strategy that always play o_1 is winning.

This assumption is fine in the context of a reactive environment, but if the environment behaves independently of the output of the system, o_1 should not appear in the assumption. Imagine the inputs are read from a file, then the environment cannot react to our outputs since the input word is already present on disk before we started producing outputs. Thus there is no way the assumption $\mathsf{F}(o_1) \Leftrightarrow \mathsf{F}(i_1)$ can be satisfied for all possible programs. In that case, the input-assumption $\mathsf{F} i_1$ which is independent from outputs is better suited.

2.2 Refinement using scenarios

As we will see in the next sections, in general there are an infinite number of incomparable optimal assumptions. This raises the problem of choosing one among all the possibilities.

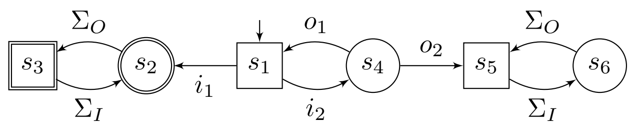

Figure 5 Büchi automaton recognising the language corresponding to LTL formula $o_1 \mathtt{U} i_1$.

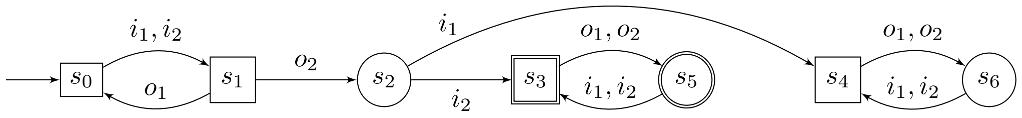

Figure 6 Büchi automaton for specification $(\neg o_2)\mathtt{U}(o_2 \wedge \mathtt{X} i_2)$.

A solution is to get some feedback from the user in the form of *scenarios*. A scenario is a behaviour that the strategy we produce should allow.

Scenario. Formally a *scenario* is given by a language $S \subseteq (\Sigma_I \cdot \Sigma_O)^\omega$. A strategy $\sigma_\exists$ is *compatible* with the set of scenarios S when $S \subseteq \mathsf{Out}(\sigma_\exists)$. Similarly S is *compatible* with specification L if $S \subseteq L$. Assumption A is *sufficient* for L and S, if there exists $\sigma_\exists$ such that $S \subseteq A \cap \mathsf{Out}(\sigma_\exists) \subseteq L$. We say that an assumption A is $\mathcal{C}$-optimal for L and S if it is sufficient for L and S and there is no $A' \in \mathcal{C}$ strictly less restrictive than A and sufficient for L and S. A scenario S is *coherent* if there is no words $w, w' \in S$ such that $w_{\le 2\cdot i+1} = w'_{\le 2\cdot i+1}$ and $w_{2\cdot i+2} \neq w'_{2\cdot i+2}$ for some $i \in \mathbb{N}$. If S is not coherent, then as no strategy can play both $w_{\le 2\cdot i+2}$ and $w'_{\le 2\cdot i+2}$, no strategy can be compatible with S. Coherence is in fact a necessary and sufficient condition for the existence of a compatible strategy.

▶ **Lemma 4.** *Scenario S is coherent if, and only if, there exists a strategy compatible with S. In particular, given a coherent scenario S and a strategy $\sigma_\exists$, the strategy $[S \to \sigma_\exists]$ is compatible with S, where:* $[S \to \sigma_\exists](h) = \begin{cases} w_{h+1} & \textit{if there is } w \in S \textit{ such that } h \textit{ is a prefix of } w \\ \sigma_\exists(h) & \textit{otherwise} \end{cases}$.

▶ **Example 5.** Consider the example in Figure 6. There are many different possible assumptions we could chose from. However if we give the scenario $\Sigma_I \cdot o_2 \cdot i_2 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ then the only ensurable-optimal assumption is $(\Sigma_I \cdot o_2 \cdot i_2 + \Sigma_I \cdot o_1 \cdot \Sigma_I) \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$. The corresponding winning strategy consists in playing o_2 for the first output.

2.3 Existence of a sufficient assumption with scenario

We show that given a scenario, there exists an input assumption which is sufficient and compatible with it. A safety assumption also exists if the scenario is itself a safety language.

▶ **Theorem 6.** *Let L be a specification and S a scenario. If S is compatible with L, then there exists an input assumption which is sufficient for L and compatible with S.*

Proof. Assume S is compatible with L and let $A = \{w \in (\Sigma_I \cdot \Sigma_O)^\omega \mid \exists w' \in S.\ \pi_I(w) = \pi_I(w')\}$. A is clearly an input assumption. We prove that for any strategy $\sigma_\exists$, A is sufficient for $[S \to \sigma_\exists]$. Let $w \in A \cap \mathsf{Out}([S \to \sigma_\exists])$. Since $w \in A$, there is $w' \in S$ such that $\pi_I(w') = \pi_I(w)$. Since $[S \to \sigma_\exists]$ is compatible with S (Lemma 4), it is compatible with w', and therefore $w' = \mathsf{Out}(\sigma_\exists, \pi_I(w)) = w$. This proves $w \in S$, and thus A sufficient for L. ◀

▶ **Theorem 7.** *Let L be a specification and S a scenario compatible with L. If S is a safety language, then there exists a safety assumption sufficient for L and compatible with S.*

▶ Remark. There are examples of scenarios which are not safety languages for which there is no sufficient safety assumption. Consider L and S both given by $\mathsf{F} i_1$. A safety assumption A compatible with S has to contain $\mathsf{F} i_1$. Assume towards a contradiction that there is $w \not\in A$. Since A is a safety assumption, w has a bad prefix, that is there is i such that $w_{\leq 2 \cdot i} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap A = \varnothing$. As $w_{\leq 2 \cdot i} \cdot (i_1 \cdot o_1)^\omega \in S$, this is contradiction with the fact that A is compatible with S. Therefore $A = (\Sigma_I \cdot \Sigma_O)^\omega$ and this is not sufficient for L.

3 General assumptions

In this section we study general assumptions, without concern for whether there are ensurable or not. Properties established here will be useful when studying ensurable assumptions.

Necessary and sufficient assumptions. Given a specification L and a strategy $\sigma_\exists$, we say that an assumption A is *necessary* for $\sigma_\exists$ if B sufficient for $\sigma_\exists$ implies $B \subseteq A$.

▶ **Lemma 8.** *Given a strategy $\sigma_\exists$, the assumption $GA(\sigma_\exists) = L \cup ((\Sigma_I \cdot \Sigma_O)^\omega \setminus \mathsf{Out}(\sigma_\exists))$ is sufficient and necessary for $\sigma_\exists$.*

▶ **Corollary 9.** *If A is optimal, then there exists $\sigma_\exists$ such that $A = GA(\sigma_\exists)$.*

Link with strongly winning strategies. Our goal is to establish a link with the notion of strongly winning strategy. Intuitively this corresponds to the strategies that play a winning strategy whenever it is possible from the current history.

▶ **Definition 10** ([10, 4]). Strategy $\sigma_\exists$ is *strongly winning* when for any history h, if there exists $\sigma'_\exists$ such that $\varnothing \neq (\mathsf{Out}(\sigma'_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega) \subseteq L$, then $(\mathsf{Out}(\sigma_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega) \subseteq L$. A *subgame winning* strategy (called subgame perfect in [10]), is such that for any history h, if there exists $\sigma'_\exists$ such that $\mathsf{Out}_h(\sigma'_\exists) \subseteq L$ then $\mathsf{Out}_h(\sigma_\exists) \subseteq L$.

▶ **Lemma 11** ([10, Lemma 1]). *For every specification, there exists strongly winning and subgame winning strategies.*

▶ **Theorem 12.** *Let $GA(\sigma_\exists) = L \cup ((\Sigma_I \cdot \Sigma_O)^\omega \setminus \mathsf{Out}(\sigma_\exists))$. If strategy $\sigma_\exists$ is strongly winning for L, then $GA(\sigma_\exists)$ is an optimal assumption for L. Reciprocally, if A is an optimal assumption for L, then there is a strongly winning strategy $\sigma_\exists$ such that $A = GA(\sigma_\exists)$.*

Proof. Assume that $\sigma_\exists$ is strongly winning. First notice that by Lemma 8, $\mathsf{GA}(\sigma_\exists)$ is sufficient for $\sigma_\exists$ and thus sufficient for L. Let A be an assumption which is sufficient for L, we will prove that $\mathsf{GA}(\sigma_\exists) \not\subset A$, which shows that $\mathsf{GA}(\sigma_\exists)$ is optimal. Let $\sigma'_\exists$ be a strategy for which A is sufficient. If $A \setminus \mathsf{GA}(\sigma_\exists) = \varnothing$, then $A \subseteq \mathsf{GA}(\sigma_\exists)$ which shows the property. Otherwise there exists $w \in A \setminus \mathsf{GA}(\sigma_\exists)$. Since $w \not\in \mathsf{GA}(\sigma_\exists)$ and $L \subseteq \mathsf{GA}(\sigma_\exists)$, $w \not\in L$, i.e. w is losing. Since $w \not\in \mathsf{GA}(\sigma_\exists)$ and $(\Sigma_I \cdot \Sigma_O)^\omega \setminus \mathsf{Out}(\sigma_\exists) \subseteq \mathsf{GA}(\sigma_\exists)$, $w \in \mathsf{Out}(\sigma_\exists)$, i.e. it is an outcome of $\sigma_\exists$. Since $A \cap \mathsf{Out}(\sigma'_\exists) \subseteq L$ and $w \in A \setminus L$, $w \not\in \mathsf{Out}(\sigma'_\exists)$, i.e. it is not an outcome of $\sigma'_\exists$. Let $w_{\leq k}$ be the longest prefix of w that is compatible with $\sigma'_\exists$. Since $\sigma_\exists$ is strongly winning and w is an outcome of $\sigma_\exists$ which is losing, for all strategies $\sigma''_\exists$, either $w_{\leq k} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap \mathsf{Out}(\sigma''_\exists) = \varnothing$ or $w_{\leq k} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap \mathsf{Out}(\sigma''_\exists) \setminus L \neq \varnothing$. Since $w_{\leq k}$ is compatible with $\sigma'_\exists$, $w_{\leq k} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap \mathsf{Out}(\sigma'_\exists) \neq \varnothing$, and therefore there is an outcome w' of $\sigma'_\exists$ which is losing. Since A is sufficient for $\sigma'_\exists$, $w' \not\in A$. Note that w' is not an outcome

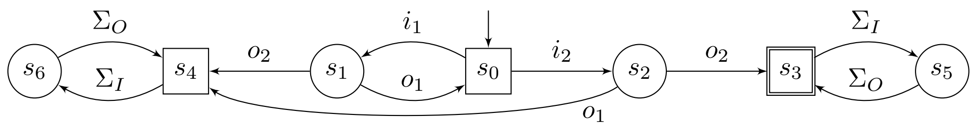

Figure 7 Büchi automaton for expression $(i_1 \cdot o_1)^* \cdot i_2 \cdot o_2 \cdot (\Sigma_I \cdot \Sigma_O)^\omega$.

of $\sigma_\exists$: $w'_{k+1} = \sigma'_\exists(w_{\le k}) \neq \sigma_\exists(w_{\le k})$. Hence, $w \in (\Sigma_I \cdot \Sigma_O)^\omega \setminus \mathsf{Out}(\sigma_\exists) \subseteq \mathsf{GA}(\sigma_\exists)$. Therefore $w' \in \mathsf{GA}(\sigma_\exists) \setminus A$ which proves $\mathsf{GA}(\sigma_\exists) \not\subset A$.

Let now A be an optimal assumption for L and $\sigma_\exists$ a strategy for which A is sufficient. Note that by Corollary 9, $A \subseteq \mathsf{GA}(\sigma_\exists)$. We show that $\sigma_\exists$ is strongly winning. Let h be a history such that there is $\sigma'_\exists$ such that $\varnothing \neq \mathsf{Out}(\sigma'_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq (L)$. We prove that $\mathsf{Out}(\sigma_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq L$ which shows the result. If $\mathsf{Out}(\sigma_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega = \varnothing$ the property holds and otherwise consider the strategy $\sigma_\exists\,[h \leftarrow \sigma'_\exists]$ that plays according to $\sigma_\exists$ and when h is reached shifts to $\sigma'_\exists$. Formally, given a history h':

$$\sigma_\exists\,[h \leftarrow \sigma'_\exists] = \begin{cases} \sigma'_\exists(h') & \text{if } h \text{ is a prefix of } h' \\ \sigma_\exists(h') & \text{otherwise} \end{cases}$$

Since h is compatible with $\sigma'_\exists$ and $\mathsf{Out}(\sigma'_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq L$, we also have $\sigma_\exists\,[h \leftarrow \sigma'_\exists] \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq L$. Moreover all outcomes not in $h \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ are compatible with $\sigma_\exists$. Hence $\mathsf{GA}(\sigma_\exists) \cup h \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ is sufficient for $\sigma_\exists\,[h \leftarrow \sigma'_\exists]$. By the optimality of assumption $\mathsf{GA}(\sigma_\exists)$, $\mathsf{GA}(\sigma_\exists) \not\subset \mathsf{GA}(\sigma_\exists) \cup h \cdot (\Sigma_I \cdot \Sigma_O)^\omega$. Hence $h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq \mathsf{GA}(\sigma_\exists)$. Since $\mathsf{GA}(\sigma_\exists)$ is sufficient for $\sigma_\exists$, $\mathsf{Out}(\sigma_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq L$, which shows the result. ◀

▶ **Example 13.** Consider the specification L given by expression $(i_1 \cdot o_1)^* \cdot i_2 \cdot o_2 \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ and for which a Büchi automaton is given in Figure 7. There is no winning strategy in this game, since if the input is always i_1 there is no way to satisfy the specification. However, if the current history is of the form $(i_1 \cdot o_1)^* \cdot i_2$, then controller has a winning strategy which consists in replying o_2. Strongly winning strategies must therefore present this behaviour for all $(i_1 \cdot o_1)^* \cdot i_2$ that are compatible with it. Consider strategy $\sigma_\exists$ such that if the first input is i_2, then $\sigma_\exists$ plays the winning strategy we described and otherwise always play o_2. This is a strongly winning strategy since for histories beginning with i_2 it is winning and for any other history compatible with $\sigma_\exists$ there is no winning strategy. The assumption corresponding to this strategy is $\mathsf{GA}(\sigma_\exists) = i_2 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega + i_1 \cdot (\Sigma_O \cdot \Sigma_I)^* \cdot o_1 \cdot (\Sigma_I \cdot \Sigma_O)^\omega$.

Infinity of optimal assumptions. As Figure 8 illustrates, there can be an infinite number of optimal assumptions.

▶ **Theorem 14.** *There is a specification for which there are an infinite number of optimal assumptions and an infinite number of optimal ensurable assumptions.*

Proof. Consider the game of Figure 8. In this game there are an infinite number of strongly winning strategies. They must all play o_2 in s_5 but have the choice of how long to stay in s_2. We write $\sigma^n_\exists$ the strategy that plays o_1, n times before playing o_2 (note that we could also consider strategies that depend on the choice of input in s_1, but this will not be necessary here). The sufficient hypothesis for $\sigma^n_\exists$ is $\mathsf{GA}(\sigma^n_\exists) = (\Sigma_I \cdot \Sigma_O)^\omega \setminus (\Sigma_I \cdot o_1)^n \cdot \Sigma_I \cdot o_2 \cdot i_1 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$. They are incomparable and since $\sigma^n_\exists$ are strongly winning they are all optimal. This shows that there is an infinite number of optimal assumptions. Note that these assumptions are ensurable, and therefore there also is an infinite number of ensurable-optimal assumptions. ◀

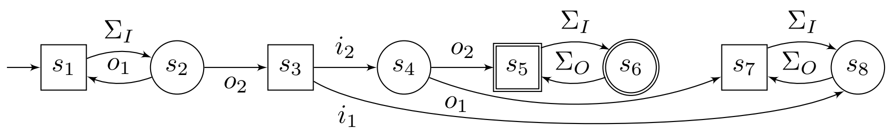

Figure 8 A parity automaton for a specification with an infinity of optimal assumptions.

Scenarios. Strategy $[S \to \sigma_\exists]$ corresponds to an optimal assumption when $\sigma_\exists$ is winning.

▶ **Lemma 15.** *If $\sigma_\exists$ is subgame winning strategy for L, then $\mathsf{GA}([S \to \sigma_\exists])$ is optimal for L and S.*

Generalisation. Assume now we are given a sufficient assumption A and want to generalise it, that is find A' optimal and such that $A \subseteq A'$. We compute $\sigma_\exists$ winning for $A \Rightarrow L$ (i.e. such that $\mathsf{Out}(\sigma_\exists) \cap A \subseteq L$ and $\sigma'_\exists$ subgame winning for L. We then define $\sigma'_\exists[A \setminus W \to \sigma_\exists]$ to be the function that maps h to $\sigma'_\exists(h)$ if h is not a prefix of a word $w \in A$ or $h \in W = \{h \mid \mathsf{Out}(\sigma'_\exists) \cap h \cdot (\Sigma_I \cdot \Sigma_O)^\omega \subseteq L\}$, and maps h to $\sigma_\exists(h)$ otherwise.

▶ **Lemma 16.** *If $\mathsf{Out}(\sigma_\exists) \cap A \subseteq L$ and $\sigma'_\exists$ is subgame winning for L, then $\mathsf{GA}(\sigma'_\exists[A \setminus W \to \sigma_\exists])$ is an optimal assumption for L and contains A.*

4 Ensurable assumptions

Consider again the solution given in Example 13. Assumption $i_2 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega + i_1 \cdot (\Sigma_O \cdot \Sigma_I)^* \cdot o_1 \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ is indeed an optimal, but it may not be what we would expect because the expression $i_1 \cdot (\Sigma_O \cdot \Sigma_I)^* \cdot o_1$ is an assumption about the controller rather than the environment. A controller which falsifies the assumption would then be considered correct. Instead of this, we would prefer an assumption which only restrict the behaviour environment. This motivates the search for nonrestrictive assumptions.

4.1 Necessary and sufficient non-restrictive assumptions

In this section, we show properties of assumptions that are not restrictive. As we have seen in Lemma 2, this coincide with ensurable assumptions for ω-regular objectives.

Given a strategy $\sigma_\exists$, the word w is *doomed for* $\sigma_\exists$ if there is an index k such that one outcome of $\sigma_\exists$ has prefix $w_{\le k}$ and all outcome of $\sigma_\exists$ that have prefix $w_{\le k}$ do not satisfy L. We write $\mathsf{Doomed}(\sigma_\exists)$ for the set of words that are doomed for $\sigma_\exists$ i.e. $\mathsf{Doomed}(\sigma_\exists) = \{w \mid \exists k \in 2 \cdot \mathbb{N}.\ \mathsf{Out}(\sigma_\exists) \cap w_{\le k} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \ne \varnothing$ and $\mathsf{Out}(\sigma_\exists) \cap w_{\le k} \cdot (\Sigma_I \cdot \Sigma_O)^\omega \cap L = \varnothing\}$. We consider the assumption $\mathsf{EA}(\sigma_\exists) = \mathsf{GA}(\sigma_\exists) \setminus \mathsf{Doomed}(\sigma_\exists)$.

▶ **Lemma 17.** *Let $\sigma_\exists$ be a strategy, we have the following properties:* **1.** *$\mathsf{EA}(\sigma_\exists)$ is sufficient for $\sigma_\exists$, and nonrestrictive;* **2.** *for all assumption A sufficient for $\sigma_\exists$ and not output-restrictive, we have that $A \subseteq \mathsf{EA}(\sigma_\exists)$.*

▶ **Example 18.** For the strategy $\sigma_\exists$ we defined in Example 13, the set of doomed histories is $i_1 \cdot o_2 \cdot (\Sigma_I \cdot \Sigma_O)^\omega$. Then $\mathsf{EA}(\sigma_\exists)$ is $i_2 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ which is nonrestrictive. This assumption describes better than $\mathsf{GA}(\sigma_\exists)$ the assumptions on the environment necessary to win. However it is not optimal among nonrestrictive assumptions, and we will now characterise the strategies for which $\mathsf{EA}(\sigma_\exists)$ is optimal.

4.2 Link between non-dominated strategies and optimal assumptions

We use the notion of weak dominance classical in game theory. Intuitively a strategy dominates another one if it performs at least as well against any strategy of the environment.

▶ **Definition 19** ([5]). Strategy $\sigma_\exists$ is *very weakly dominated* from history h by strategy $\sigma'_\exists$ if for all strategy $\sigma_\forall$ of the environment, $\mathsf{Out}_h(\sigma_\exists, \sigma_\forall) \in L \Rightarrow \mathsf{Out}_h(\sigma'_\exists, \sigma_\forall) \in L$. It is *weakly dominated* from h by $\sigma'_\exists$ if moreover $\sigma'_\exists$ is not very weakly dominated by $\sigma_\exists$ from h. A strategy is said *non-dominated* if no strategy weakly-dominates it from the empty history ε. A strategy is *non-subgame-dominated* if there is no strategy that weakly-dominates it from any history h. A strategy is said *dominant* if it very weakly dominates all strategies.

We can draw a link between optimal assumptions and non-dominated strategies.

▶ **Lemma 20.** *If $\mathsf{EA}(\sigma_\exists) \subseteq \mathsf{EA}(\sigma'_\exists)$ then $\sigma_\exists$ is very weakly dominated by $\sigma'_\exists$.*

▶ **Lemma 21.** *If $\sigma_\exists$ is very weakly dominated by $\sigma'_\exists$ then $\mathsf{EA}(\sigma_\exists) \subseteq \mathsf{EA}(\sigma'_\exists)$.*

▶ **Example 22.** Consider again Example 13 and a strategy $\sigma'_\exists$ which plays o_1 in s_1 and o_2 in s_2. We have that $\sigma'_\exists$ weakly dominates $\sigma_\exists$. The assumption necessary to $\sigma'_\exists$ is $\mathsf{GA}(\sigma'_\exists) = (i_1 \cdot o_1)^* \cdot (i_1 \cdot o_2 + i_2 \cdot \Sigma_O) \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ which is incomparable with $\mathsf{GA}(\sigma_\exists)$: it does not contain $(i_1 \cdot o_1)^\omega$ for instance. But $\mathsf{Doomed}(\sigma'_\exists) = \varnothing$ while $\mathsf{Doomed}(\sigma_\exists) = i_1 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega$ (which rules out $(i_1 \cdot o_1)^\omega$). So we indeed have $\mathsf{EA}(\sigma_\exists) \subset \mathsf{EA}(\sigma'_\exists)$.

▶ **Theorem 23.** *Let L be an ω-regular specification. If $\sigma_\exists$ is a non-dominated strategy for L, then $\mathsf{EA}(\sigma_\exists)$ is ensurable optimal for L. Reciprocally if A is an ensurable optimal assumption for L, then there is a non-dominated strategy $\sigma_\exists$ for L such that $A = \mathsf{EA}(\sigma_\exists)$.*

Proof. Let $\sigma_\exists$ be a non-dominated strategy. By Lemma 17, the assumption $\mathsf{EA}(\sigma_\exists)$ is sufficient for L and not output-restrictive. We now prove it is optimal. Let A be a nonrestrictive assumption sufficient for L and $\sigma'_\exists$ for which A is sufficient. By Lemma 8, $A \subseteq \mathsf{GA}(\sigma'_\exists)$. As $\sigma_\exists$ is not weakly dominated by $\sigma'_\exists$, either: **1.** $\sigma_\exists$ is not very weakly dominated by $\sigma'_\exists$. Then by Lemma 20 $\mathsf{EA}(\sigma_\exists) \not\subseteq \mathsf{EA}(\sigma'_\exists)$. Or **2.** $\sigma_\exists$ very weakly dominates $\sigma'_\exists$ then by Lemma 21, $\mathsf{EA}(\sigma'_\exists) \subseteq \mathsf{EA}(\sigma_\exists)$. Therefore $\mathsf{EA}(\sigma_\exists) \not\subset \mathsf{EA}(\sigma'_\exists)$ and by Lemma 17 $A \subseteq \mathsf{EA}(\sigma'_\exists)$, so $\mathsf{EA}(\sigma_\exists) \not\subset A$. This shows that $\mathsf{EA}(\sigma_\exists)$ is $\mathcal{E}$-optimal for L.

Now let $\sigma_\exists$ be a strategy such that $\mathsf{EA}(\sigma_\exists)$ is $\mathcal{E}$-optimal, we show that $\sigma_\exists$ is non-dominated. Let $\sigma'_\exists$ be a strategy which very weakly dominates $\sigma_\exists$, we prove that $\sigma_\exists$ very weakly dominates $\sigma'_\exists$. By Lemma 21, $\mathsf{EA}(\sigma_\exists) \subseteq \mathsf{EA}(\sigma'_\exists)$. Since $\mathsf{EA}(\sigma_\exists)$ is optimal, $\mathsf{EA}(\sigma_\exists) \not\subset \mathsf{EA}(\sigma'_\exists)$. Therefore $\mathsf{EA}(\sigma_\exists) = \mathsf{EA}(\sigma'_\exists)$. By Lemma 20 this implies that $\sigma'_\exists$ is very weakly dominated by $\sigma_\exists$ and shows that $\sigma_\exists$ is not weakly dominated. ◀

▶ **Corollary 24.** *If $\sigma_\exists$ is dominant, then $\mathsf{EA}(\sigma_\exists)$ is the unique $\mathcal{E}$-optimal assumption.*

4.3 Computation of optimal ensurable assumptions

In parity games, deciding the existence of a winning strategy from a particular state is in the complexity class $\mathsf{NP} \cap \mathsf{coNP}$ [9]. We will show that if an algorithm for solving parity games is available, then the other operations to obtain optimal assumptions can be performed efficiently. We first construct a representation of one arbitrary non-dominated strategy. Our construction is based on the notion of memoryless strategies: given L as a parity automaton, a strategy is said *memoryless* if it only depends on the current state of the automaton, in other words it can be implemented with a Moore machine with the same structure as the given automaton.

▶ **Lemma 25.** *Given a parity automaton and a memoryless strategy $\sigma_\exists$ which ensures we are winning from each state in the winning region, we can compute in polynomial time a Moore machine implementing a memoryless non-dominated strategy $\sigma'_\exists$.*

By combining this construction with a parity automaton for L, we can build an automaton for $\mathsf{GA}(\sigma'_\exists) = L \cup ((\Sigma_I \cdot \Sigma_O)^\omega \setminus \mathsf{Out}(\sigma_\exists))$. We can then exclude doomed histories by removing transitions going to states from which there is no winning path, and obtain an automaton which recognises $\mathsf{EA}(\sigma_\exists)$.

▶ **Lemma 26.** *Given a specification as a parity automaton, and a strategy $\sigma_\exists$ as a Moore machine, we can compute in polynomial time a parity automaton recognising $\mathsf{EA}(\sigma_\exists)$.*

▶ **Theorem 27.** *Given a specification as a parity automaton, we can compute in exponential time a parity automaton of polynomial size recognising an ensurable optimal assumption. Moreover, if we have access to an oracle for computing memoryless winning strategies in parity games, our algorithm works in polynomial time.*

Proof. We first need to obtain a Moore machine for a memoryless winning strategy $\sigma_\exists$, this can be done in exponential time or constant time if we have an oracle. Then by Lemma 25, we can compute a Moore machine implementing a memoryless non-dominated strategy $\sigma'_\exists$. By Lemma 26, we can construct a parity automaton recognising $\mathsf{EA}(\sigma'_\exists)$. By Lemma 23, the language of this automaton is an ensurable optimal assumption. ◀

4.4 Scenarios

Assume now, we are given a scenario and asked for a correct system compatible with the scenario. We first characterise optimal ensurable assumptions that are needed for this.

▶ **Theorem 28.** *Let L be a specification, and S a coherent scenario compatible with L. If $\sigma_\exists$ is a non-subgame-dominated, then $\mathsf{EA}([S \to \sigma_\exists])$ is $\mathcal{E}$-optimal for L and S.*

▶ **Lemma 29.** *Given a parity automaton for a coherent scenario S and a strategy $\sigma_\exists$ we can compute in polynomial time a Moore machine for $[S \to \sigma_\exists]$.*

We can now compute an optimal ensurable assumption compatible with the scenario.

▶ **Theorem 30.** *Given a specification L and a scenario S as parity automata, we can compute in exponential time a parity automaton of polynomial size recognising an $\mathcal{E}$-optimal assumption for L and S.*

Proof. We can compute in exponential time a memoryless strategy in parity game and as seen in Lemma 25, deduce in polynomial time a memoryless non-dominated strategy $\sigma'_\exists$. From the definition of non-subgame-dominated, this strategy is in fact non-subgame-dominated. Then by Lemma 29, we can compute a Moore machine for $[S \to \sigma_\exists]$. By Theorem 28, the corresponding assumption $\mathsf{EA}([S \to \sigma_\exists])$ is ensurable optimal for L and S. By Lem 26, $\mathsf{EA}([S \to \sigma_\exists])$ can be computed in polynomial time. ◀

4.5 Generalisation

We have seen in Lemma 16 that from a winning strategy for $A \Rightarrow L$ and a strongly winning strategy for L, we could obtain a strategy $\sigma_\exists$ that has both properties. Furthermore, we can compute $\sigma'_\exists$ that is strongly non-dominated for L and define a strategy that is both non-dominated for L and winning for $A \Rightarrow L$. We define $\sigma'_\exists[A \to \sigma_\exists]$ to be the function that maps h to $\sigma_\exists(h)$ if $h \cdot \sigma_\exists(h)$ is a prefix of some $w \in A$ and maps h to $\sigma'_\exists(h)$ otherwise.

▶ **Lemma 31.** *If $\sigma_\exists$ is winning for $A \Rightarrow L$ and strongly winning for L and $\sigma'_\exists$ is strongly non-dominated for L, then $\mathsf{EA}(\sigma'_\exists[A \to \sigma_\exists])$ contains A and is ensurable-optimal for L.*

▶ **Lemma 32.** *Given strategies $\sigma'_\exists$, $\sigma_\exists$ as Moore machines and assumption A as a parity automaton, can construct in polynomial time a Moore machine for $\sigma'_\exists[A \to \sigma_\exists]$.*

▶ **Theorem 33.** *There is an exponential algorithm that given L and A sufficient for L as parity automata, computes a parity automaton whose language A' is such that $A \subseteq A'$ and A' ensurable-optimal for L.*

Proof. Assume we are given automata $\mathcal{A}_A$ for A, and $\mathcal{A}_L$ for L. We construct $\mathcal{A}_{A \Rightarrow L}$ recognising $L \cup (\Sigma_I \cdot \Sigma_O)^\omega \setminus A$, and $\sigma_\exists$ a winning strategy in $\mathcal{A}_{A\Rightarrow L}$, then compute in exponential time a memoryless strategy $\sigma'_\exists$ which is winning in $\mathcal{A}_L$ from all states from which there is a winning strategy [13]; it is in fact strongly winning. We can construct a Moore machine for $\sigma'_\exists[A \to \sigma_\exists]$ (Lemma 32), and also an automaton for $\mathsf{EA}(\sigma'_\exists[A \to \sigma_\exists])$ (Lemma 26). $\mathsf{EA}(\sigma'_\exists[A \to \sigma_\exists])$ is ensurable optimal for L and contains A (Lemma 31). ◀

5 Input-assumptions

We now focus on input assumptions. There can be an infinite number of incomparable ones that are sufficient. This can be seen in the example of Figure 6, where the specification was $(\neg o_2)\mathtt{U}(o_2 \wedge \mathtt{X} i_2)$. There, we need the assumption to tell us when exactly the first i_2 will occur. This corresponds to assumptions of the form $A_n = (\Sigma_I \cdot \Sigma_O)^\omega \setminus \{(\Sigma_I \cdot \Sigma_O)^n \cdot i_1 \cdot \Sigma_O \cdot (\Sigma_I \cdot \Sigma_O)^\omega\}$. Any such assumption will be sufficient and they are all incomparable.

▶ **Theorem 34.** *There is a specification L for which there are an infinite number of optimal input assumptions.*

We show a link between input assumptions and a class of strategies called remorsefree.

▶ **Definition 35** (Remorsefree)**.** Given a specification L, a strategy $\sigma_\exists$ is *remorsefree* if for all $\sigma'_\exists$ and $w \in \Sigma_I^\omega$, $\mathsf{Out}(\sigma'_\exists, w) \models L$ implies $\mathsf{Out}(\sigma_\exists, w) \models L$. This is the notion used in [8] for dominance. A strategy $\sigma_\exists$ is *remorsefree-admissible* if for all $\sigma'_\exists$ either $\forall w \in \Sigma_I^\omega$. $\mathsf{Out}(\sigma'_\exists, w) \models L \Rightarrow \mathsf{Out}(\sigma_\exists, w) \models L$ or $\exists w \in \Sigma_I^\omega$. $\mathsf{Out}(\sigma_\exists, w) \models L \not\Rightarrow \mathsf{Out}(\sigma'_\exists, w) \models L$.

▶ **Lemma 36.** *Given a strategy $\sigma_\exists$, if $L \neq \varnothing$ then the input-assumption $\mathsf{IA}(\sigma_\exists) = \{w \in (\Sigma_I \cdot \Sigma_O)^\omega \mid \pi_I(w) \notin \pi_I(\mathsf{Out}(\sigma_\exists) \setminus L)\}$ is sufficient for $\sigma_\exists$. Moreover if A is an input-assumption which is sufficient for $\sigma_\exists$ then $A \subseteq \mathsf{IA}(\sigma_\exists)$.*

▶ **Example 37.** Consider the game of Figure 9. In this game the only remorsefree strategy is to output o_1 at the first step. The corresponding assumption is $A = \Sigma_I \cdot \Sigma_O \cdot ((i_2 + i_3) \cdot \Sigma_O)^\omega$ while the assumption corresponding to the strategy outputting o_2 is $\Sigma_I \cdot \Sigma_O \cdot (i_3 \cdot \Sigma_O)^\omega$ which is more restrictive. The assumption A is indeed the unique optimal input-assumption.

In [8], Damm and Finkbeiner show that there is a remorsefree strategy if, and only if, there is a unique minimal assumption for L. We generalise this result using the associated notion of admissibility to characterise the minimal assumptions that are sufficient to win.

▶ **Theorem 38.** *If $\sigma_\exists$ is a remorsefree admissible strategy for L, then $\mathsf{IA}(\sigma_\exists)$ is an optimal input-assumption for L. Reciprocally if A is an optimal input-assumption for L, then there is a remorsefree admissible strategy $\sigma_\exists$, such that $A = \mathsf{IA}(\sigma_\exists)$.*

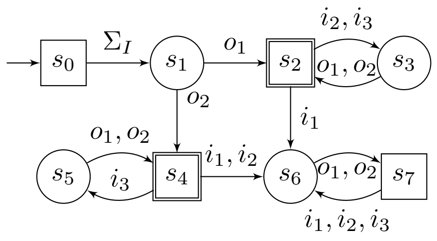

Figure 9 Büchi automaton for which the remorsefree strategy consists in outputting o_1.

Proof. Let $\sigma_\exists$ be a remorsefree-admissible strategy and A the corresponding environment assumption. Let B be such that $A \subset B$. We prove that B is not sufficient for L which will show that A is optimal. Assume towards a contradiction that B is sufficient for a strategy $\sigma'_\exists$. Since $\sigma_\exists$ is remorsefree-admissible, one of those two cases occurs: **1.** $\forall w' \in \Sigma_I^\omega.\ \mathsf{Out}(\sigma'_\exists, w') \models L \Rightarrow \mathsf{Out}(\sigma_\exists, w') \models L$. Let $w \in B \setminus A$. Since $A = \mathsf{IA}(\sigma_\exists)$, we have that $w \in \pi_I(\mathsf{Out}(\sigma_\exists) \setminus L)$. Hence $\mathsf{Out}(\sigma_\exists, w) \not\models L$, and we have that $\mathsf{Out}(\sigma'_\exists, w) \not\models L$ which shows that B is not sufficient for $\sigma'_\exists$; or **2.** $\exists w' \in \Sigma_I^\omega.\ \mathsf{Out}(\sigma_\exists, w') \models L \wedge \mathsf{Out}(\sigma'_\exists, w') \not\models L$. We have that w' belongs to A by definition, thus it belongs to B by hypothesis, and since $\mathsf{Out}(\sigma'_\exists, w') \not\models L$, B is not sufficient for $\sigma'_\exists$.

Let now A be an optimal assumption for L and $\sigma_\exists$ the corresponding strategy. We show that $\sigma_\exists$ is remorsefree-admissible. Let $\sigma'_\exists$ be another strategy. Since A is optimal, it is not strictly included in $B = \mathsf{IA}(\sigma'_\exists)$, so either $A = B$ or $A \backslash B \neq \varnothing$. **1.** If $A = B$ then we show that $\forall w \in \Sigma_I^\omega.\ \mathsf{Out}(\sigma'_\exists, w) \models L \Rightarrow \mathsf{Out}(\sigma_\exists, w) \models L$: **a.** if $w \in A = \mathsf{IA}(\sigma_\exists)$, then $\mathsf{Out}(\sigma_\exists, w) \models L$, and the implication holds. **b.** if $w \notin A = B = \mathsf{IA}(\sigma'_\exists)$, then $\mathsf{Out}(\sigma'_\exists, w) \not\models L$, and the implication holds. **2.** Otherwise $A \backslash B \neq \varnothing$ then let $w \in A \backslash B$. Since $w \in A$, $\mathsf{Out}(\sigma_\exists, w) \models L$ and since $w \notin B = \mathsf{IA}(\sigma'_\exists)$, $\mathsf{Out}(\sigma'_\exists, w) \not\models L$. Hence $\mathsf{Out}(\sigma_\exists, w) \models L \not\Rightarrow \mathsf{Out}(\sigma'_\exists, w) \models L$. ◀

References

1 Dietmar Berwanger. Admissibility in infinite games. In *STACS*, volume 4393 of *LNCS*, pages 188–199. Springer, February 2007.

2 Roderick Bloem, Rüdiger Ehlers, Swen Jacobs, and Robert Könighofer. How to handle assumptions in synthesis. In *SYNT*, pages 34–50, 2014. `doi:10.4204/EPTCS.157.7`.

3 Aaron Bohy, Véronique Bruyère, Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. Acacia+, a tool for LTL synthesis. In *Computer Aided Verification*, pages 652–657. Springer, 2012.

4 Romain Brenguier, Jean-François Raskin, and Ocan Sankur. Assume-admissible synthesis. In *CONCUR*, volume 42 of *LIPIcs*, pages 100–113. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. `doi:10.4230/LIPIcs.CONCUR.2015.100`.

5 Romain Brenguier, Jean-François Raskin, and Mathieu Sassolas. The complexity of admissibility in omega-regular games. In *CSL-LICS '14, 2014*. ACM, 2014. `doi:10.1145/2603088.2603143`.

6 Krishnendu Chatterjee and Thomas A Henzinger. Assume-guarantee synthesis. In *TACAS'07*, volume 4424 of *LNCS*. Springer, 2007.

7 Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Environment assumptions for synthesis. In *CONCUR*, pages 147–161. Springer, 2008.

8 Werner Damm and Bernd Finkbeiner. Does it pay to extend the perimeter of a world model? In *FM 2011: Formal Methods*, pages 12–26. Springer, 2011.

9 E Allen Emerson and Charanjit S Jutla. Tree automata, mu-calculus and determinacy. In *32nd Annual Symposium on Foundations of Computer Science*, pages 368–377. IEEE, 1991.

10 Marco Faella. Admissible strategies in infinite games over graphs. In *MFCS 2009*, volume 5734 of *Lecture Notes in Computer Science*, pages 307–318. Springer, 2009.

11 Rachel Faran and Orna Kupferman. Spanning the spectrum from safety to liveness. In *Automated Technology for Verification and Analysis*, pages 183–200. Springer, 2015.

12 Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. An antichain algorithm for LTL realizability. In *Computer Aided Verification*, pages 263–277. Springer, 2009.

13 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games: A Guide to Current Research*. Springer-Verlag, New York, NY, USA, 2002.

14 Swen Jacobs, Roderick Bloem, Romain Brenguier, Rüdiger Ehlers, Timotheus Hell, Robert Könighofer, Guillermo A. Pérez, Jean-François Raskin, Leonid Ryzhyk, Ocan Sankur, Martina Seidl, Leander Tentrup, and Adam Walker. The first reactive synthesis competition (SYNTCOMP 2014). *Int J Softw Tools Technol Transfer*, pages 1–24, 2016. `doi:10.1007/s10009-016-0416-3`.

15 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 179–190. ACM, 1989.