Does It Pay to Extend the Perimeter of a World Model?*

Werner Damm[1] and Bernd Finkbeiner[2]

[1] Carl von Ossietzky Universität Oldenburg
[2] Universität des Saarlandes

Abstract. Will the cost for observing additional real-world phenomena in a world model be recovered by the resulting increase in the quality of the implementations based on the model? We address the quest for optimal models in light of industrial practices in systems engineering, where the development of control strategies is based on combined models of a system and its environment. We introduce the notion of remorsefree dominance between strategies, where one strategy is preferred over another if it outperforms the other strategy in comparable situations, even if neither strategy is guaranteed to achieve all objectives. We call a world model optimal if it is sufficiently precise to allow for a remorsefree dominating strategy that is guaranteed to remain dominant even if the world model is refined. We present algorithms for the automatic verification and synthesis of dominant strategies, based on tree automata constructions from reactive synthesis.

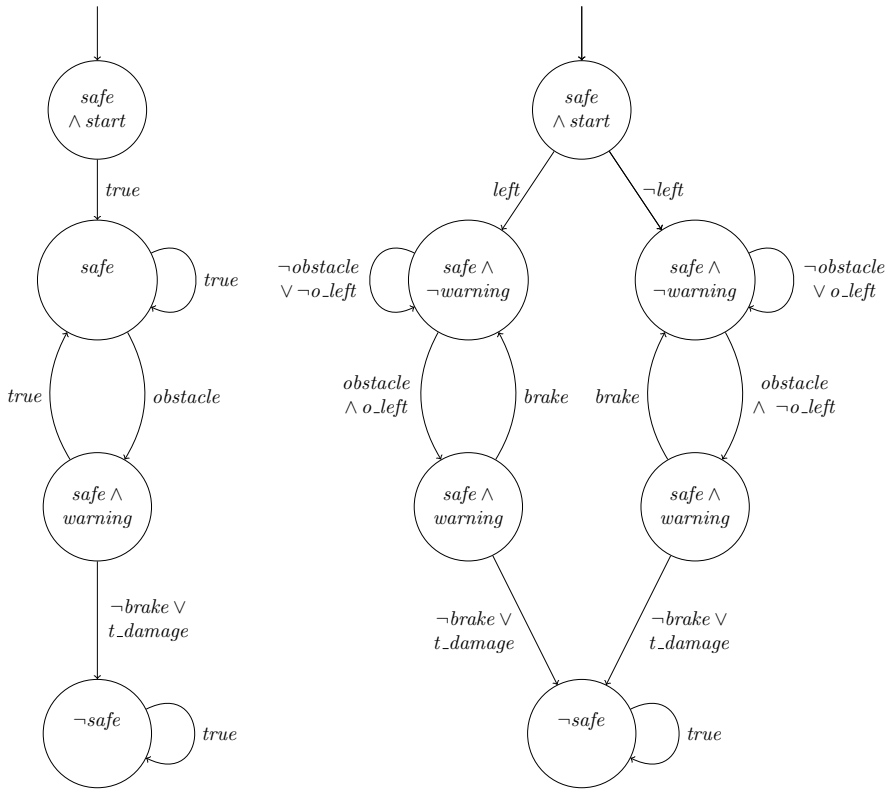
1 Introduction

What constitutes a good model? We revisit this fundamental question using concepts and algorithms from reactive synthesis, which allow us to construct and compare different strategies for achieving a given collection of objectives. A key challenge rests in the identification of the perimeter of the model: given a (physical) system S under development, what real-world aspects could potentially impact S in a way that endangers its proper functioning? Examples of systems of interest are aircrafts, cars, nuclear power plants, defibrillators, production plants, or in general what is often referred to as cyber-physical systems, i.e., systems where both software and hardware components play key roles in realizing the system's functionality.

1.1 A Motivating Example

Suppose we wish to develop a driver assistance system that maintains, whenever possible, a safe distance to objects ahead on the same lane, such as another car or some obstacle on the road. The starting point for the development is the world

* This research was partially supported by the German Science Foundation within the Transregional Collaborative Research Center TR14 AVACS.



(a) simple world model (b) refined world model

Fig. 1. Example world models for a driver assistance system. Sets of labels are specified using logical formulas: for example, the edges marked with *true* are labeled with all subsets of $\{obstacle, o\_left, t\_damage, left, brake\}$.

model shown in Figure 1(a), which specifies how environment disturbances like the appearance of obstacles (*obstacle*) or a sudden tire damage (*t_damage*), and controllable system actions, such as braking (*brake*), result in transitions on the world states.

In the initial state the world is *safe*. Once an obstacle appears somewhere ahead, we observe (through some distance sensor) a *warning* signal, as indicated in the middle state; at this point, the world may transition into the unsafe bottom state. Braking will prevent the unsafe state unless a rare event such as a sudden tire damage (*t_damage*) causes the transition to the unsafe state. Clearly, this world model is very simple; in particular, we have not modeled the lane in which our car is traveling and the lane where the obstacle appears. As a result, the effect of a decision not to brake is nondeterministic: after not braking in the *warning* state, the next state will either be *safe*, if the obstacle was indeed on

the same lane as the car and we thus avoided the obstacle by changing the lane, or the bottom unsafe state, if the obstacle was in fact on the other lane. The nondeterminism can be eliminated in a refined world model, such as the one shown in Figure 1(b): this world model extends the perimeter to additionally include the concept of lanes. In this example, there is a left and a right lane (for the same direction of travel). The states on the left correspond to the case that the car is on the left lane, the states on the right to the case that the car is on the right lane. In the initial state, the car chooses one of the two lanes, we assume that there is no further possibility to change lanes. The refined world model in Figure 1(b) is larger and more detailed than the simple world model in Figure 1(a). But does the extension pay?

Let us assume that the primary objective of our driver assistance system is to maintain safety, i.e., to ensure that *safe* is always true, and that an additional, secondary, objective is to avoid braking, i.e., to ensure that *brake* is always false. Consider the following three example control strategies: Strategy 1 never brakes. Strategy 2 brakes if and only if a *warning* has occurred. Strategy 3 brakes all the time. In both world models it is clear that none of the three strategies guarantees that the objectives are met in all situations: irrespectively of the strategy, the occurrence of an obstacle followed by tire damage will always cause a transition to the unsafe state, violating the primary objective. A more meaningful comparison of the strategies is therefore to see how they perform on particular sequences of disturbances.

Strategy 3 is clearly not dominant, because it brakes, and, hence, violates the secondary objective, even in the middle *safe* state, where there is no danger that the primary objective might be violated. Strategies 1 and 2 avoid this unnecessary braking. Because of the nondeterminism in the simple world model, it is, however, also impossible to identify one of the two strategies as remorsefreely dominant. If, starting from the *warning* state, the assistance system does not brake, following Strategy 1, then there is the possibility that the unsafe state is reached and, hence, the primary objective is violated. It would thus have been preferable to brake, as suggested by Strategy 2. On the other hand, if the assistance system does brake, following Strategy 2, then there is also the possibility that not braking would have led to a transition to the *safe* state, in which case the secondary objective was violated unnecessarily.

The simple world model thus does not allow us to choose a dominant strategy. Let us analyze the same situation in the refined world model of Figure 1(b). Strategy 2 still dominates Strategies 1 and 3, because Strategy 1 may violate the primary objective in the *warning* state, and Strategy 3 may violate the secondary objective in middle *safe* state. As before, Strategy 3 never performs better than Strategy 2. Unlike in the simple model, however, the same holds for Strategy 1: In the *warning* state it is now always better to brake, as suggested by Strategy 2, because the nondeterministic possibility to return to the middle *safe* state without braking has been eliminated. Hence, the payoff of the refinement of the world model is that we can identify Strategy 2 as dominant.

1.2 From Correctness to Dominance

As the example illustrates, cyber-physical systems should not be expected to always guarantee their functional requirements – the system is likely to exhibit failures when exposed to a physical environment outside the perimeter of the employed world model. Since the concept of an all encompassing complete model is of philosophical interest only, the notion of "correctness" has to be revisited and replaced by notions addressing the inherent incompleteness of verification approaches induced from modeling boundaries.

In this paper, we propose such a new notion of correctness based on a comparison of the available strategies. We call the strategic dominance described in the example (for each sequence of disturbances, the dominating strategy achieves at least the same priority as the dominated strategy) *remorsefree dominance*. Remorsefree dominance allows us to compare two strategies even if both strategies violate some objectives. Intuitively, remorsefree dominance means that one never feels "remorse" about a decision, because, in retrospect, after having seen a sequence of disturbances, the other strategy would appear preferable.

We compare the strategic capabilities when extending the perimeter of the world model: does it pay to extend the perimeter of a world model from from a set V of real-world phenomena to a set $V \cup \{e\}$ – i.e., can strategies which are aware about the real-world phenomenon e avoid violations of the system requirements more often than strategies which are only aware about real-world phenomena contained in V?

We must be careful to distinguish the unavailability of information about a phenomenon due to an incomplete observation of the world through a limited sensor system on the one hand from a complete absence of the phenomenon in the world model on the other. In systems engineering, this distinction corresponds to the following two different design questions. The first question is that of *sensor completeness:* Relative to a given perimeter of the world model, will adding certain sensors strengthen the strategic capabilities to achieve the given objectives? For example, in the driver assistance system, it is irrelevant whether we directly observe the occurrence of an *obstacle* as long as we observe the *warning* signal. The second question refers to the *perimeter of the model:* Will extending the perimeter of the model add strategic capabilities?

To answer the first question, we study strategy classes that are indexed by the available observables: The strategy class $\mathcal{S}_I$ consists of all strategies that determine the system actions based on observations from I. We are interested in finding strategies from one class $\mathcal{S}_I$ that remorsefreely dominate all strategies from another class $\mathcal{S}_J$ with more ($I \subset J$) or even full information.

To answer the second question, we search for strategies that remain dominant under all possible refinements of the world model. A simple (but correct) model hides the impact of certain phenomena with nondeterminism; whether or not this imprecision has an impact on the existence of a dominating strategy depends on how relevant the phenomenon is for the objectives under consideration. We thus have a formal characterization of what constitutes an optimal world model: A world model is *optimal* if the description of the relevant phenomena is suffi-

ciently precise to guarantee the existence of a dominating strategy that remains dominant in all possible refinements of the model.

1.3 Verification and Synthesis

An algorithmic treatment of these questions combines aspects of verification (which objectives are achieved?) with aspects of synthesis (is there a better strategy?). The algorithmic approach of the paper is therefore based on constructions on automata over infinite trees, similar to those used in the automata-theoretic synthesis of reactive systems [1–5]. We characterize remorsefree dominance as a language on infinite trees by providing a tree automaton that accepts all dominating strategies. This automaton can be used to verify that a given strategy is dominant (it is dominant iff it is accepted by the automaton) and to synthesize a dominant strategy iff such a strategy exists (it exists iff the language of the automaton is nonempty). We also give analogous constructions for the set of strategies that remain dominant over all refinements of the world model.

The paper thus provides a conceptual and algorithmic framework for what could be called a new relativized theory of correctness, which accepts that systems may fail and replaces absolute guarantees (all objectives are guaranteed to be achieved) with the relative notions of dominance and optimality.

2 Foundations

2.1 World Models

We study the interaction of a system with its environment. Let $\mathcal{V}_S$ be a finite set of system variables, modeling actions under the system's control, such as the setting of actuators, and $\mathcal{V}_E$ be an arbitrary set of *environment variables*, in the context of control theory corresponding to the variables of the plant model. We assume that the environment variables are partitioned into disjoint sets of *disturbances* $\mathcal{V}_D$, modeling uncontrollable environment observations, and *controllable environment variables* $\mathcal{V}_C$, modeling phenomena in the environment, i.e., actions and observations which can be influenced by the system through the system variables. For the purposes of this paper, we assume that all values can be finitely encoded, and thus, without loss of generality, assume all variables to be of Boolean type.

We formalize the possible interactions between the system and its environment using labeled graphs, which we call *world models*. Nodes represent the states of a plant. Transitions between nodes then represent the effect of a given setting of the system variables and a given disturbance on the current state of the plant. The actual choice of settings of system variables will be determined by (control-) strategies discussed below.

We consider world models that restrict only a finite subset $V_E \subseteq \mathcal{V}_E$ of the environment variables. We call this subset the *perimeter* of the world model. As made formal in the definition below, models with restricted perimeter can be

understood as full models by assigning arbitrary valuations to the environment variables outside the perimeter of the model. We denote, for a valuation[3] $\sigma \subseteq V_E$ of environment variables V_E, the set of valuations of $\mathcal{V}_E$ obtained by assigning arbitrary valuations to environment variables outside the perimeter by $\mathcal{E}(\sigma) = \{\sigma \cup r \mid r \subseteq (\mathcal{V}_E \setminus V_E)\}$.

A *world model* $M = (V_E, N, n_0, E, L_N, L_E)$ is a labeled directed graph, where $V_E \subseteq \mathcal{V}_E$ is the subset of environment variables observed by the model, N is a possibly infinite set of nodes, $n_0 \in N$ is an initial node, $E \subseteq N \times N$ is a set of edges, and $L_N : N \to 2^{2^{\mathcal{V}_C}}, L_E : E \to 2^{2^{\mathcal{V}_S \cup \mathcal{V}_D}}$ is a pair of labeling functions that assign to each node a set of valuations of the controllable environment variables and to each edge a set of valuations of the system variables and the disturbances.

We denote by $V_C = V_E \cap \mathcal{V}_C$ and $V_D = V_E \cap \mathcal{V}_D$ the finite sets of controllable environment variables and disturbances, respectively, within the perimeter of the world model. We assume L_N and L_E to be induced from valuations of environment variables within the perimeter, e.g., L_N to be induced by a labeling function $L'_N : N \to 2^{2^{V_C}}$ such that $L_N(n) = \bigcup_{\sigma \in L'_N(n)} \mathcal{E}(\sigma)$, and, similarly, L_E to be induced by a function $L'_E : E \to 2^{2^{\mathcal{V}_S \cup V_D}}$ such that $L_E(e) = \bigcup_{\sigma \in L'_E(E)} \mathcal{E}(\sigma)$. We furthermore assume that the world models are *total*, i.e., for each node $n \in N$, each valuation σ_S of the system variables, and each valuation σ_D of the disturbances there exists a node $m \in N$ such that $\sigma_S \cup \sigma_E \in L'_E(n, m)$.

As an example, consider the world models of Figure 1. The models refer to the system variables $\mathcal{V}_S = \{brake, left\}$, the disturbances $\mathcal{V}_D = \{obstacle, o\_left, t\_damage\}$ and the controllable environment variables $\mathcal{V}_C = \{safe, start, warning\}$. The perimeter of the simple world model, $V_E = \{obstacle, t\_damage, safe, start, warning\}$, is a subset of the perimeter of the refined world model, $V'_E = V_E \cup \{left, o\_left\}$.

The purpose of the world model is to predict the consequences of system actions. Because each model only considers a finite number of phenomena, however, it typically abstracts from some relevant phenomena with nondeterminism: We call a world model *nondeterministic* if there exist two edges $(n, m_1), (n, m_2)$ such that the edge labels are not disjoint, $L'_E(n, m_1) \cap L'_E(n, m_2) \neq \emptyset$, and the union of the target node labels is not singleton $|L'_N(m_1) \cup L'_N(m_2)| \geq 1$. Otherwise, we call the world model *deterministic*[4].

It is often possible to reduce the nondeterminism by extending the perimeter of the world model. While the models may differ in their precision, they model the same reality. We therefore expect them to be consistent in the sense that the *more concrete* (or *refined*) model must be simulated by the *more abstract* model.

A concrete world model $M_1 = (V_{E_1}, N_1, n_{0_1}, E_1, L_{E_1}, L_{N_1})$ *is simulated by* an abstract world model $M_2 = (V_{E_2}, N_2, n_{0_2}, E_2, L_{E_2}, L_{N_2})$ with $V_{E_2} \subseteq V_{E_1}$ iff there

[3] All variables are of Boolean type. A variable is evaluated to true iff it is an element of σ.

[4] Note that a deterministic model may still have multiple runs due to different evaluations of the system and disturbance variables.

exists a *simulation relation* $R \subseteq \{(n_1, n_2) \in N_1 \times N_2 \mid L_{N_1}(n_1) \subseteq L_{N_2}(n_2)\}$ such that

1. $(n_{0_1}, n_{0_2}) \in R$ and
2. for all $(n_1, n_2) \in R, (n_1, m_1) \in E_1$, and $\sigma \in L_{E_1}(n_1, m_1)$ there exists a pair $(n_2, m_2) \in E_2$ such that $\sigma \in L_{E_2}$ and $(m_1, m_2) \in R$.

For example, the world model in Figure 1(a) simulates the world model in Figure 1(b). The simulation relation relates two nodes from the two wolds models whenever they are depicted horizontally next to each other, i.e., the initial nodes are related, the nodes labeled *safe* $\wedge \neg$*warning* are related to the node labeled *safe*, etc.

2.2 Strategic Objectives

We use linear-time temporal logic (LTL) [6] to specify strategic objectives. Let $\mathcal{V} = \mathcal{V}_S \cup \mathcal{V}_E$. We interpret LTL formulas over *computations*, which are infinite sequences $\sigma = \sigma_0\sigma_1\sigma_2 \ldots \in (2^{\mathcal{V}})^\omega$ of variable valuations. We denote restrictions of a computation σ or other sequences of variables to a subset $X \subseteq \mathcal{V}$ of the variables by $\sigma|_X = \sigma_0 \cap X\, \sigma_1 \cap X, \ldots$. The satisfaction of an LTL formula φ on a computation σ is denoted by $\sigma \models \varphi$.

An *objective specification* $\varphi = (\mathcal{O}, p)$ consists of a set $\mathcal{O}$ of LTL formulas, called *objectives*, and a *priority function* $p : \mathcal{O} \to \{1, \ldots, |\mathcal{O}|\}$, which identifies the priority of each objective as a positive number, where 1 is the most important priority.

In the driver assistance system, the objectives consist of two invariants, the invariant $\Box safe$ with priority two, and the invariant $\Box\neg$*brake* with priority two.

We say that a computation σ *satisfies priorities up to* n if n is the greatest nonnegative number $n \in \mathbb{N}_{\geq 0}$ such that $\sigma \models \phi$ for all $\phi \in \mathcal{O}$ and $p(\phi) \leq n$. A set of computations $\mathcal{C}$ *satisfies priorities up to* n if n is the greatest nonnegative number $n \in \mathbb{N}_{\geq 0}$ such that $\sigma \models \phi$ for all $\varphi \in \mathcal{O}$, $p(\phi) \leq n$, and $\sigma \in \mathcal{C}$. If a strategy s or a set $\mathcal{S}$ of strategies satisfies priorities up to 0, i.e., none of the priorities are met, we say that s or $\mathcal{S}$, respectively, *completely violates* $(\mathcal{O}, p)$.

2.3 Strategy Classes

Let $M = (V_E, N, n_0, d_0, E, L_N, L_E)$ be a world model over V_E, modeling the influence of a valuation of the system variables $\mathcal{V}_S$ and the disturbances V_D on the controllable environment variables. Let $V = V_E \cup \mathcal{V}_S$.

A *control strategy* (or short: *strategy*) for M selects the valuation of the system variables dependent on the sequence of valuations of the environment variables V_E as observed through a finite set of *observables* $I \subseteq V_E$. Formally, a control strategy over observables I is a function $s : (2^I)^* \to 2^{\mathcal{V}_S}$.

A control strategy s and a sequence of disturbances $\sigma_D \in (2^{V_D})^\omega$ determine jointly the following set of computations $C_M(\sigma_D, s)$ in the world model M: $C_M(\sigma_D, s) = \{\sigma_0\sigma_1 \ldots \in (2^V)^\omega \mid \exists n_0 n_1 \ldots \in N^\omega \,.\, \sigma_0 = L'_N(n_0) \cup d_0 \cup s(\epsilon) \wedge \forall j >$

0 . $(n_{j-1}, n_j) \in E$ $L'_E(n_{j-1}, n_j) = (\sigma_S, \sigma_D) \wedge \sigma_j = \sigma_S \cup \sigma_D \cup L'_N(n_j)\}$, where $\sigma_S = s(\sigma_0 \cap I \ \ldots \ \sigma_{j-1} \cap I)$.

We call the class of such strategies s_I the *I-observation strategy class*, denoted by $\mathcal{S}_I$. The special case $I = \mathcal{V}_E$ is called the *full-observation class*. We use the term *partial observation strategies* for strategies in some I-observation strategy class with $I \subsetneq \mathcal{V}_E$.

2.4 Winning strategies

We call the set $\mathcal{C}_M(s) = \bigcup_{\sigma_D \in (2^{V_D})^\omega} C_M(\sigma_D, s)$ of computations that result from combining a strategy s with sequences of disturbances the *computations of s*. We say that strategy s is *winning up to priority n* if every computation in $\mathcal{C}_M(s)$ satisfies the objective specification up to priority n. This induces a partial order $\sqsubseteq$ on strategies: a strategy s is *dominated by* a strategy t, denoted by $s \sqsubseteq_M t$, if s is winning up to priority n, t is winning up to priority m, and $n \leq m$.

3 Remorsefree Dominance

As discussed in the introduction, it is often unrealistic to expect a strategy to enforce an objective for *every* possible environment behavior. For example, in the driver assistance system, a tire damage can always cause a transition to the unsafe state. If even the full-observation strategies violate the objective specification completely, then the $\sqsubseteq$-hierarchy collapses: all strategies violate the objective specification completely.

We now introduce a finer dominance order that allows us to distinguish strategies even if none of the objectives can be guaranteed. Remorsefree dominance refers to individual computations rather than full strategies. We say a computation σ *is dominated by* a computation η, denoted by $\sigma \sqsubseteq \eta$, if σ satisfies the objective specification up to priority m, η up to priority n, and $m \leq n$.

A strategy s *is remorsefreely dominated* (in world model M) by a strategy t, denoted by $s \preceq_M t$, iff for every sequence of disturbances $\sigma_D \in (2^{V_D})^\omega$ and every computation $c \in C_M(\sigma_D, t)$ there is a computation $c' \in C_M(\sigma_D, s)$ such that c dominates c'.

To motivate the "for every ... there is" quantification in the above definition, consider the following two variations of the same scenario: An aircraft is approaching the airport under strong shear winds. In variation 1, the aircraft is not equipped with a shear wind sensor, and the autopilot (unaware of the shear wind) initiates an approach. In this variation, an uncontrollable nondeterminism will decide over life and death, i.e., whether by chance the aircraft is still able to land, or whether the shear wind will cause the aircraft to crash on the ground. In variation 2, the aircraft is equipped with a shear wind sensor, and the autopilot tests for the velocity and changes of the shear wind and automatically initiates an abort of the approach. In variation 1, there is thus an uncontrollable chance that both the primary objective (safety) and the secondary objective (landing) is achieved. However, at the uncontrollable risk of violating the primary objective.

In variation 2, by contrast, safety is guaranteed at the price of excluding the possibility of achieving the secondary objective. In such a situation, we argue that the more informed strategy should be considered remorsefreely dominating.

Comparing remorsefree dominance $\preceq_M$ to the notion of dominance defined in Section 3, it is easy to see that $s \preceq_M t$ implies $s \sqsubseteq t$:

Theorem 1. *For two strategies s and t, $s \preceq_M t$ implies $s \sqsubseteq_M t$.*

Proof. Assume, by way of contradiction, that $s \preceq_M t$ and $s \not\sqsubseteq_M t$. Since $s \not\sqsubseteq_M t$, there exists a priority n such that all computations in $\mathcal{C}_M(s)$ satisfy the objective specification up to priority n but there exists a computation $c \in \mathcal{C}(t)$ that violates priority n. Let σ_D be the sequence of disturbances in c. No computation in $\mathcal{C}_M(\sigma_D, s)$ is dominated by c, which is in $\mathcal{C}_M(\sigma_D, t)$. This contradicts the assumption that $s \preceq_M t$. □

The converse does not hold: suppose, for example, that strategies s_1 and s_2 both completely violate the objective specification, but s_1 achieves some objective for some specific sequence of disturbances, while s_2 achieves no objective, no matter what the environment does; then $s_1 \sqsubseteq_M s_2$, but not $s_1 \preceq_M s_2$.

In addition to comparing strategies with strategies, we are also interested in comparing strategy classes with strategies: A strategy class $\mathcal{S}$ is *remorsefully dominated* by a specific strategy $t \in \mathcal{T}$ from some strategy class $\mathcal{T}$, denoted by $\mathcal{S} \preceq_M t$, iff for every strategy $s \in \mathcal{S}$, it holds that $s \preceq_M t$.

4 An Automata-Theoretic Characterization of Remorsefree Dominance

The remainder of the paper is devoted to the algorithmic analysis of remorsefree dominance. We start, in this section, by constructing an automaton over infinite trees that characterizes the set of remorsefreely dominating strategies in a particular strategy class. This automaton can be used to verify that a strategy is dominant and to synthesize dominant strategies.

4.1 Preliminaries: Automata over Infinite Words and Trees

We assume familiarity with automata over infinite words and trees. In the following we only give a quick summary of the standard terminology, the reader is referred to [7] for a full exposition.

A (full) *tree* is given as the set Υ^* of all finite words over a given set of directions Υ. For given finite sets Σ and Υ, a *Σ-labeled Υ-tree* is a pair $\langle \Upsilon^*, l \rangle$ with a labeling function $l : \Upsilon^* \to \Sigma$ that maps every node of Υ^* to a letter of Σ.

An *alternating tree automaton* $\mathcal{A} = (\Sigma, \Upsilon, Q, q_0, \delta, \alpha)$ runs on Σ-labeled Υ-trees. Q is a finite set of states, $q_0 \in Q$ a designated initial state, δ a transition function $\delta : Q \times \Sigma \to \mathbb{B}^+(Q \times \Upsilon)$, where $\mathbb{B}^+(Q \times \Upsilon)$ denotes the positive Boolean combinations of $Q \times \Upsilon$, and α is an acceptance condition. Intuitively, disjunctions in the transition function represent nondeterministic choice; conjunctions start

an additional branch in the run tree of the automaton, corresponding to an additional check that must be passed by the input tree. A run tree on a given Σ-labeled Υ-tree $\langle \Upsilon^*, l \rangle$ is a $Q \times \Upsilon^*$-labeled tree where the root is labeled with $(q_0, l(\varepsilon))$ and where for a node n with a label (q, x) and a set of children $\mathit{child}(n)$, the labels of these children have the following properties:

- for all $m \in \mathit{child}(n)$: the label of m is $(q_m, x \cdot \upsilon_m)$, $q_m \in Q, \upsilon_m \in \Upsilon$ such that (q_m, υ_m) is an atom of $\delta(q, l(x))$, and
- the set of atoms defined by the children of n satisfies $\delta(q, l(x))$.

A run tree is *accepting* if all its paths fulfill the acceptance condition. A *parity condition* is a function α from Q to a finite set of colors $C \subset \mathbb{N}$. A path is accepted if the highest color appearing infinitely often is even. The *safety condition* is the special case of the parity condition where all states are colored with 0. The *Büchi condition* is the special case of the parity condition where all states are colored with either 1 or 2. A Σ-labeled Υ-tree is *accepted* if it has an accepting run tree. The set of trees accepted by an alternating automaton $\mathcal{A}$ is called its *language* $\mathcal{L}(\mathcal{A})$. An automaton is empty iff its language is empty.

A *nondeterministic* automaton is a special alternating automaton where the image of δ consists only of such formulas that, when rewritten in disjunctive normal form, contain exactly one element of $Q \times \{\upsilon\}$ in every disjunct. A *deterministic* automaton is a special nondeterministic automaton where the image of δ contains no disjunctions.

A *word automaton* is the special case of a tree automaton where the set Υ of directions is singleton. For word automata, we omit the direction in the transition function.

4.2 Dominant Computations

A strategy dominates another if, for every sequence of disturbances, each computation that is produced by the first strategy dominates some computation produced by the second. We begin our construction with a word automaton that checks for dominance between computations. In order to relate two computations that result from the same sequence of disturbances, we duplicate the variables in $\mathcal{V}_S$ and V_C, i.e., we consider sequences over the alphabet $2^{V_D \cup \mathcal{V}_S \cup V_C \cup \mathcal{V}'_S \cup V'_C}$, where $\mathcal{V}'_S$ and V'_C are sets of fresh "primed" variables duplicating the variables in $\mathcal{V}_S$ and V_C, respectively. We write $\mathit{primed}(X)$ for the set of primed variables corresponding to the variables in a set X, and, for two sequences $\sigma \in (2^{V_D \cup V_S \cup V_C})^\omega$ and $\eta \in (2^{V_D \cup V'_S \cup V'_C})^\omega$, we write $[\sigma, \eta]$ for the sequence $((\sigma_0[V \mapsto V']) \cup \eta_0)\,((\sigma_1[V \mapsto V']) \cup \eta_1)\ldots$, where $[V \mapsto V']$ indicates that each variable from V is replaced by the corresponding primed variable in V'.

Lemma 1. *Let $\varphi = (\mathcal{O}, p)$ be an objective specification, let $V = V_D \cup \mathcal{V}_S \cup V_C$, $\mathcal{V}'_S = \mathit{primed}(\mathcal{V}_S), V'_C = \mathit{primed}(V_C)$, and $V' = V_D \cup \mathcal{V}'_S \cup V'_C$. There exists a deterministic parity word automaton $\mathcal{A}_\varphi$ over $V \cup V'$ such that $[\sigma, \eta]$ is accepted by $\mathcal{A}_\varphi$ iff $\sigma \sqsubseteq \eta$.*

Proof. We define an LTL formula ψ for the desired property as follows:

$$\psi = \bigwedge_{n \in \{1,\ldots,|\mathcal{O}|\}} \left(\bigwedge_{\phi \in \mathcal{O}, p(\phi) \leq n} \phi[V \mapsto V'] \;\rightarrow \bigwedge_{\phi \in \mathcal{O}, p(\phi) \leq n} \phi \right).$$

We translate ψ to a deterministic parity word automaton using a standard construction (cf. [8]). □

The definition of dominant strategies compares different computations of a particular world model M. In preparation for the construction of the automaton for dominant strategies in the next subsection, we condition $\mathcal{A}_\varphi$ with respect to M.

Lemma 2. *Let M be a model, $\varphi = (\mathcal{O}, p)$ an objective specification, and let $V = V_D \cup \mathcal{V}_S \cup V_C$, $\mathcal{V}'_S = primed(\mathcal{V}_S), V'_C = primed(V_C)$, and $V' = V_D \cup \mathcal{V}'_S \cup V'_C$. There exists a parity word automaton $\mathcal{B}_{M,\varphi}$ over $V \cup V'$ such that the sequence $[\sigma, \eta]$ is accepted by $\mathcal{B}_{M,\varphi}$ iff (1) $\sigma \sqsubseteq \eta$ and σ is a computation of M or (2) η is not a computation of M.*

Proof. We start by translating the world model M into a safety word automaton $\mathcal{A}_M$ over the alphabet $2^{V_D \cup V_C \cup \mathcal{V}_S}$, which accepts the sequences of variable evaluations allowed by M. Let $\mathcal{A}'_M$ be the same automaton as $\mathcal{A}_M$, but over the alphabet $2^{V_D \cup V'_C \cup \mathcal{V}'_S}$ with the variables in V_C and $\mathcal{V}_S$ replaced by their counterparts in V'_C and $\mathcal{V}'_S$, respectively. We represent condition (1) by building the product of $\mathcal{A}'_M$ with $\mathcal{A}_\varphi$. We represent condition (2) by complementing $\mathcal{A}_M$. $\mathcal{B}_{M,\varphi}$ is an automaton that accepts the language union of the two automata. □

4.3 Dominant Strategies

We now construct a tree automaton that recognizes the subset of strategies in $\mathcal{S}_{I_1}$ that dominate a strategy class $\mathcal{S}_{I_2}$ in a given world model M. We assume that $I_1 \subseteq I_2$, i.e., the reference strategy class $\mathcal{S}_{I_2}$ has the larger set of observable variables. A first observation is that we can, without loss of generality, assume that $V_D \subseteq I_2$. As explained in the following lemma, this is due to the fact that the definition of remorsefree dominance refers to individual sequences of disturbances. If some strategy in $\mathcal{S}_{I_2 \cup V_D}$ is not dominated, then this is due to some individual computation; however, there exists a strategy in $\mathcal{S}_{I_2}$ that behaves exactly as the strategy in $\mathcal{S}_{I_2 \cup V_D}$ on that particular computation.

Lemma 3. *Let M be a world model and let $\mathcal{S}_{I_1}$ and $\mathcal{S}_{I_2}$ be two strategy classes with $I_1 \subseteq I_2$. For every $t \in \mathcal{S}_{I_1}$ it holds that $\mathcal{S}_{I_2} \preceq_M t$ iff $\mathcal{S}_{I_2 \cup V_D} \preceq_M t$.*

Proof. The "if" direction is obvious, because $\mathcal{S}_{I_2} \subseteq \mathcal{S}_{I_2 \cup V_D}$. For the "only if" direction, suppose, by way of contradiction, that $\mathcal{S}_{I_2} \preceq_M t$ but $\mathcal{S}_{I_2 \cup V_D} \not\preceq_M t$. Since $\mathcal{S}_{I_2 \cup V_D} \not\preceq_M t$, there exists a sequence of disturbances $\sigma_D \in (2^{V_D})^\omega$ and a computation $c \in C_M(\sigma_D, t)$ such there exists an alternative strategy $t' \in S_{I_2 \cup V_D}$

where for all $c' \in C_M(\sigma_D, t')$ it holds that $c' \not\preceq c$. There exists, however, the strategy $t'' \in \mathcal{S}_{I_2}$ with $t''(\sigma_0\sigma_1 \dots \sigma_k) = t'((\sigma_0 \cup \sigma_{D,0})(\sigma_1 \cup \sigma_{D,1}) \dots (\sigma_k \cup \sigma_{D,k}))$, which has the same set $C_M(\sigma_D, t'') = C_M(\sigma_D, t')$ of strictly better computations. This contradicts the assumption that $\mathcal{S}_{I_2} \preceq_M t$. □

The construction of the tree automaton in the following theorem uses the conjunctive branching available in alternating automata to compare against all possible strategies in $\mathcal{S}_{I_2}$.

Theorem 2. *Let M be a world model, φ an objective specification, and $\mathcal{S}_{I_1}$ and $\mathcal{S}_{I_2}$ two strategy classes with $I_1 \subseteq I_2$. Then there exists an alternating parity tree automaton $\mathcal{C}_{M,\varphi,I_1,I_2}$ that recognizes the subset of strategies in $\mathcal{S}_{I_1}$ that dominate the strategy class $\mathcal{S}_{I_2}$ in M.*

Proof. We construct a tree automaton that reads a strategy of $\mathcal{S}_{I_1}$ as its input tree and simulates all possible reference strategies in $\mathcal{S}_{I_2}$ in its run tree. Since $I_1 \subseteq I_2$, the reference strategy may have more branches than the input tree. These additional branches are simulated by the transition function as follows: The part due to V_D is simulated by conjunction, since the computation of the input strategy must dominate the computation of the reference strategy for all possible sequences of disturbances. Applying Lemma 3, we assume that $V_D \subseteq I_2$. The reference strategy therefore branches according to (at least) V_D.

The part due to $I_2 \setminus V_D$ is simultated by disjunction, because we may choose the computation of the reference strategy. The part due to $V_C \setminus I_2$ is chosen by projection: we choose a computation of the reference strategy for every computation of the input strategy; unlike the part due to I_2, however, the reference strategy does not branch according to these choices. We therefore project and determinize the word automaton $\mathcal{B}_{M,\varphi}$ accordingly.

Let $\mathcal{B}_{M,\varphi}$ be as defined in Lemma 2. Let $\mathcal{B}' = (2^{V_D \cup \mathcal{I}_1 \cup \mathcal{V}_S \cup I_2' \cup \mathcal{V}_S'}, \{\emptyset\}, Q, q_0, \delta, \alpha)$ be the determinization of the universal projection with respect to $V_C \setminus I_1$ of the existential projection with respect to $V_C' \setminus I_2'$ of $\mathcal{B}_{M,\varphi}$. We construct the alternating parity tree automaton $\mathcal{C}_{M,\varphi,I_1,I_2} = (2^{\mathcal{V}_S}, 2^{I_1}, Q, q_0, \delta', \alpha)$ with transition function

$$\delta'(q, W_S) = \bigwedge_{W_D \subseteq V_D} \bigwedge_{W_1 \subseteq I_1 \setminus V_D} \bigvee_{W_2' \subseteq I_2' \setminus V_D'} \bigwedge_{W_S' \subseteq \mathcal{V}_S'} (\delta(q, W_S \cup W_D \cup W_1 \cup W_2' \cup W_S'), (W_D \cup W_1) \cap I_1).$$

□

5 Verifying and Synthesizing Dominant Strategies

With the tree automaton $\mathcal{C}_{M,\varphi,I_1,I_2}$ constructed in the previous section, we can check if a given strategy is dominant (iff it is accepted by $\mathcal{C}_{M,\varphi,I_1,I_2}$) and if the strategy class $\mathcal{S}_{I_1}$ contains a strategy that dominates $\mathcal{S}_{I_2}$ (iff $\mathcal{C}_{M,\varphi,I_1,I_2}$ is nonempty).

Theorem 3. *Let M be a world model, φ an objective specification and $\mathcal{S}_{I_1}$ and $\mathcal{S}_{I_2}$ two strategy classes with $I_1 \subseteq I_2$. We can automatically check whether a given strategy in $\mathcal{S}_{I_1}$ (given as a safety word automaton) dominates $\mathcal{S}_{I_2}$, and whether $\mathcal{S}_{I_1}$ contains a strategy that remorsefreely dominates $\mathcal{S}_{I_2}$.*

Proof. To check whether a given strategy (represented as a safety word automaton T) is accepted by $\mathcal{C}_{M,\varphi,I_1,I_2}$, we solve the parity game that results from combining T and $\mathcal{C}_{M,\varphi,I_1,I_2}$ (cf. [9]). To check whether there exists a strategy that is accepted by $\mathcal{C}_{M,\varphi,I_1,I_2}$, we translate the alternating automaton into an equivalent nondeterministic automaton. Language emptiness of the nondeterministic automaton can be checked by solving a parity game on its state graph. If the language is non-empty, an accepted tree can be extracted. □

6 Towards Optimal World Models

We now address the fundamental question whether a given world model is optimal: does it pay to refine the model?

For a given objective specification and strategy class, we say that a world model is *optimal* if the current strategy class already contains a strategy that dominates all strategies in the same class for any refined model. For example, the simple world model from Figure 1(a) is not optimal because there is no dominant strategy; the refined world model from Figure 1(b) is optimal due to the dominating Strategy 2 (brake if and only if a *warning* has occurred).

In the automata-theoretic analysis, we can safely restrict our attention to refined world models that are deterministic. In a deterministic model, a sequence of disturbances and a strategy determine a unique computation. Remorsefree dominance in the deterministic models thus implies remorsefree dominance in nondeterministic models, where it suffices to dominate some computation from a set of possible computations.

In the following, we describe the necessary adaptations of our constructions. When comparing the computations of two strategies in an unknown but deterministic refinement of a world model M, we must ensure that variations in V_C may only occur if there was a previous variation in $\mathcal{V}_S$. The following lemma adapts Lemma 2 accordingly:

Lemma 4. *Let M be a deterministic world model, let $\varphi = (\mathcal{O}, p)$ be an objective specification, and let $V = V_D \cup \mathcal{V}_S \cup V_C$, $\mathcal{V}'_S = primed(\mathcal{V}_S), V'_C = primed(V_C)$, and $V' = V_D \cup \mathcal{V}'_S \cup V'_C$. There exists a parity word automaton $\mathcal{E}_{M,\varphi}$ over $V \cup V'$ such that $[\sigma, \eta]$ is accepted by $\mathcal{E}_{M,\varphi}$ iff (1) $\sigma \sqsubseteq \eta$ and σ is a computation of M and the valuation of the variables in V_C differs in σ and η only if there was a previous difference in the valuation of the variables in $\mathcal{V}_S$ or (2) η is not a computation of M.*

Proof. The construction of $\mathcal{E}_{M,\varphi}$ is analogous to the construction of $\mathcal{B}_{M,\varphi,I_2}$ in Lemma 2. For the modified condition (1), we add a memory bit to the state of the automaton that records whether a deviation in the valuation of the variables

in $\mathcal{V}_S$ has occurred. If a deviation in the valuation of the variables in V_C occurs before the bit has been set, the automaton enforces condition (2). □

The following theorem is an adaptation of Theorem 2 that checks for dominance in all deterministic refinements of the world model.

Theorem 4. *Let M be a world model with environment variables V_E, φ an objective specification, and $\mathcal{S}_I$ a strategy class. There exists an alternating parity tree automaton $\mathcal{F}_{M,\varphi,I}$ that recognizes the subset of strategies in $\mathcal{S}_I$ that dominate a strategy class $\mathcal{S}_I$ in every deterministic model M' such that M simulates M'.*

Proof. The construction of $\mathcal{F}_{M,\varphi,I}$ is analogous to the construction of $\mathcal{C}_{M,\varphi,I_1,I_2}$ in Theorem 2. Let $\mathcal{E}_{M,\varphi}$ be the parity word automaton from Lemma 4 and let $\mathcal{E}' = (2^{V_D \cup \mathcal{V}_S \cup \mathcal{V}'_S \cup V_C \cup V'_C}, \{\emptyset\}, Q, q_0, \delta, \alpha)$ be the determinization of the universal projection with respect to $(V_C \setminus I) \cup (V'_C \setminus I')$ of $\mathcal{B}_{M,\varphi}$. We construct the alternating parity tree automaton $\mathcal{F}_{M,\varphi,I} = (2^{\mathcal{V}_S}, 2^I, Q, q_0, \delta', \alpha)$ with transition function

$$\delta'(q, W_S) = \bigwedge_{W_D \subseteq V_D} \bigwedge_{W_1 \subseteq I \setminus V_D} \bigwedge_{W'_2 \subseteq I' \setminus V'_D} \bigwedge_{W'_S \subseteq \mathcal{V}'_S} (\delta(q, W_D \cup W_1 \cup W'_2 \cup W'_S), (W_D \cup W_1) \cap I).$$

The difference to the construction in Theorem 2 is that the valuation of the variables in $V'_C \setminus V'_D$ is chosen conjunctively, rather than disjunctively: if the world model is deterministic, then the set of computations corresponding to the reference strategy is singleton. □

Using the tree automaton from Theorem 4, we can check if our world model is optimal: M is optimal iff there exists a strategy that dominates the strategies of the same class in all deterministic refinements of M, which is the case iff $\mathcal{F}_{M,\varphi,I}$ is nonempty.

Theorem 5. *Let M be a world model, φ an objective specification and $\mathcal{S}_I$ a strategy class. We can automatically check whether M is optimal for φ and $\mathcal{S}_I$ and, in case of a positive answer, synthesize a remorsefreely dominating strategy.*

Proof. To check whether there exists a strategy that is accepted by $\mathcal{F}_{M,\varphi,I}$ and to obtain such a strategy, we proceed as in the proof of Theorem 3: we translate the alternating automaton into an equivalent nondeterministic automaton and solve the resulting emptiness game. □

7 Conclusions

One of the paradoxical challenges in bringing formal methods to practice is that detecting errors is most useful when done early, at a point in the design process when it is still inexpensive to make a change; but how can one apply formal verification if there is no system yet to analyze?

Arguably, the approach presented in this paper brings formal methods to the earliest possible point in the design process, when the developer has not

even started with the design, but rather tries to understand the environment in which the planned system is to achieve its objectives. Our constructions allow the designer to optimize the world model to ensure that no objective will be missed because too few real-world phenomena have been considered.

Since the verification of remorsefree dominance must not only analyze the given strategy, but also search through an entire class of strategies for an alternative, our algorithms are more complex than typical verification algorithms and are, in fact, closer to synthesis [1–5] than to standard verification. Recently, there has been a lot of interest in efficient implementation techniques for synthesis algorithms (such as bounded synthesis [5] and GR(1) synthesis [4]), for which the analysis of world models should provide an interesting new application domain.

A fundamental difference to both the classical verification and synthesis problems is, however, that we do not necessarily expect our strategies to meet all objectives and instead only demand optimality with respect to remorsefree dominance. While such a relative notion of correctness is very unusual in formal verification and synthesis, it is an established concept in general decision theory: Regret minimization [10], for example, is a model of choice under uncertainty, where the player minimizes the difference between the payoff of the chosen strategy and the payoff that would have been obtained with a different course of action.

References

1. Pnueli, A., Rosner, R.: On the synthesis of a reactive module. In: Proc. of POPL. (1989) 179–190
2. Kupferman, O., Vardi, M.Y.: Synthesis with incomplete informatio. In: Proc. of ICTL. (1997)
3. Finkbeiner, B., Schewe, S.: Uniform distributed synthesis. In: Proc. of LICS. (2005) 321–330
4. Piterman, N., Pnueli, A., Sa'ar, Y.: Synthesis of reactive(1) designs. In: Proc. of VMCAI. (2006) 364–380
5. Finkbeiner, B., Schewe, S.: SMT-based synthesis of distributed systems. In: Proc. of AFM. (2007)
6. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer-Verlag, New York (1991)
7. Grädel, E., Thomas, W., Wilke, Th., eds.: Automata, Logics, and Infinite Games. Volume 2500 of LNCS. Springer-Verlag (2002)
8. Vardi, M.Y., Wilke, T.: Automata: from logics to algorithms. In Flum, J., Grädel, E., Wilke, T., eds.: Logic and Automata: History and Perspectives. (2007) 629–736
9. Jurdziński, M.: Small progress measures for solving parity games. In: Proc. STACS. (2000) 290–301
10. Loomes, G., Sugden, R.: Regret theory: An alternative theory of rational choice under uncertainty. Economic Journal **92** (1982) 805–824