Optimal Observation Mode Scheduling for Systems under Temporal Constraints

Eva Tesařová*, Mária Svoreňová*•, Jiří Barnat*, Ivana Černá*

***Abstract*— Modern control systems use various sensors to decrease the amount of uncertainty under which they operate. While providing observation of the current state of the system, sensors require resources such as energy, time and communication. We consider discrete models of such systems with non-deterministic control transitions and multiple observation modes that provide different information about the system's states. We consider two control problems. First, we aim to construct a control and observation mode scheduling strategy that guarantees satisfaction of a finite-time temporal property given as a formula of syntactically co-safe fragment of LTL (scLTL) and at the same time, minimizes the worst-case cost associated with observation modes until the point of satisfaction. Second, the bounded version of the problem is considered, where the temporal property must be satisfied within given finite time bound. We present correct and optimal solutions to both problems and demonstrate their usability on a case study motivated by robotic applications.**

I. Introduction

Control systems used in transportation, medical and other safety critical applications typically operate under uncertainty. There is the internal uncertainty of the system's control inputs such as noisy actuators in mobile robots, and the external uncertainty from the system's interaction with the environment such as other robots or people operating in the same space. To lower the uncertainty, sensors are deployed to provide information about the current state of the system. Individual sensors and their combinations may provide varying, partial observation and their deployment requires resources such as energy, time or communication. An example of a robotic system with multiple sensing capabilities and limited resources is a planetary rover [13].

The field of sensor scheduling studies the problem of the deployment of sensors in order to optimize estimation of a signal connected to the system's state. Results include techniques, *e.g.*, for linear systems [19], hybrid systems [9] and for applications in robot motion planning [13]. On the other hand, in the field of information gathering a fixed set of sensors is assumed and the aim is to control the system, rather than the use of sensors, with the same goal to optimize estimation of the signal. Recently, a problem combining optimization with temporal objectives has been considered in this context for discrete systems [10].

Partial observability has been extensively studied for discrete systems in artificial intelligence and game theory. The main focus is typically on partially observable Markov decision processes (POMDPs) that model both partial observation and probabilistic uncertainties. The problem of controlling a POMDP has been studied for various cost functions [3], [11], [14] as well as temporal objectives including reachability, safety, stability of response, expressed as formulas of Linear Temporal Logic (LTL) or Computation Tree Logic (CTL) [5]. However, these problems prove to be undecidable or expensive to solve. More importantly, all the above results consider only one fixed observation mode.

In this work, we focus on systems with multiple observation modes. We first present a discrete model for such a system called a *non-deterministic transition system (NTS) with observation modes*. The non-determinism can be used to model the internal and external uncertainty of the system, and observation modes capture the sensing capabilities. During executions of the NTS, one decides which control action and mode of partial observation to apply, and observation modes are associated with costs. Then, we focus on the following two problems. First, we aim to construct a control and observation mode scheduling strategy for an NTS with observation modes that (i) guarantees satisfaction of a finite-time temporal property given as a formula of syntactically co-safe fragment of LTL (scLTL) and (ii) minimizes the worst-case cost accumulated until the point of satisfaction. The second problem considers the bounded version of the above problem, where the temporal property is required to be satisfied in at most $k \geq 1$ steps. Leveraging techniques from automata-based model checking and graph theory, we present correct and optimal solutions to both problems. We refer the reader to the full version of this manuscript in [18] for proofs and more detailed explanation of the framework.

To the best of our knowledge, discrete systems with multiple observations were first considered only recently in [7], [6], with the focus on control with respect to properties in infinite time horizon. The most related work to ours is [2] that considers a variation of POMDPs, where at each step the user can either choose to use the partial or full information about the current state by paying a fixed cost. The authors discuss the problem of constructing a strategy to minimize the overall cost, while reaching a designated goal state with probability 1. In comparison, we introduce a model that extends the one in [2] in the sense that we allow multiple observation modes and we design optimal strategies that guarantee complex temporal objectives over finite or bounded time horizon while minimizing the corresponding cost.

*Faculty of Informatics, Masaryk University, Brno, Czech Republic, `xtesaro2@fi.muni.cz`, `svorenova@mail.muni.cz`, `barnat@fi.muni.cz`, `cerna@muni.cz`.

•Department of Computer Science, University of Oxford, Oxford, United Kingdom, `maria.svorenova@cs.ox.ac.uk`.

This work has been partially supported by the Czech Science Foundation grant No. 15-08772S, ERC Advanced Grant VERIWARE and EPSRC Programme Grant EP/M019918/1.

II. Preliminaries

We use X^* to denote the set of all finite sequences over a set X. A finite sequence $\sigma = x_0 \dots x_n \in X^*$ has length $|\sigma| = n+1$, $\sigma(i) = x_i$ is the i-th element and $\sigma^i = \sigma(i) \dots \sigma(n)$ is the suffix starting with the i-th element, for $0 \leq i \leq n$. For an infinite sequence $\rho = x_0 x_1 \dots \in X^\omega$, $\rho(i) = x_i$ for all $i \geq 0$. A prefix of a finite or infinite sequence ρ is any sequence $\rho(0) \dots \rho(k)$ for $0 \leq k \leq |\sigma|$ or $k \geq 0$, respectively.

A. System with observation modes

Definition 1 (NTS): A non-deterministic transition system (NTS) is a tuple $\mathcal{N} = (S, A, T, s_{\text{init}}, AP, L)$, where S is a non-empty finite set of states, A is a non-empty finite set of actions, $T\colon S \times A \to 2^S$ is a transition function, $s_{\text{init}} \in S$ is the initial state, AP is a set of atomic propositions, $L\colon S \to 2^{AP}$ is a labeling function.

A run of a NTS is an infinite sequence $s_0 s_1 \dots \in S^\omega$ such that for every $i \geq 0$ there exists $a \in A$ with $s_{i+1} \in T(s_i, a)$. A finite run is a finite prefix of a run of the NTS.

Definition 2 (NTS with observation modes): A NTS with observation modes is a tuple $(\mathcal{N}, O, M)$, where $\mathcal{N} = (S, A, T, s_{\text{init}}, AP, L)$ is a NTS, O is a non-empty finite set of observations and M is a non-empty finite set of observation modes. Every observation mode $m \in M$ is associated with an observation function $\gamma_m : S \to 2^O$ and a cost $g_m \in \mathbb{R}_0^+$.

A run of a NTS with observation modes is an infinite sequence $\rho = (s_0, m_0)(s_1, m_1) \dots \in (S \times M)^\omega$ such that $s_0 s_1 \dots$ is a run of the NTS. A finite run $\sigma = (s_0, m_0) \dots (s_n, m_n) \in (S \times M)^*$ of the NTS with observation modes is a finite prefix of a run. A pair $(s, m) \in S \times M$ of a state and an observation mode is called a configuration.

Given a finite run $\sigma = (s_0, m_0) \dots (s_n, m_n)$, we define the cost of σ as follows

$$g(\sigma) = \sum_{i=0}^{n} g_{m_i}. \qquad (1)$$

The observational trace of a run $\rho = (s_0, m_0)(s_1, m_1) \dots$ is the sequence $\gamma(\rho) = \gamma_{m_0}(s_0)\gamma_{m_1}(s_1) \dots \in (2^O)^\omega$ and the propositional trace of ρ is the sequence $L(\rho) = L(s_0)L(s_1) \dots \in (2^{AP})^\omega$. The observational and propositional traces of finite runs are defined analogously.

Definition 3 (Strategy): Given a NTS with observation modes, a (observation-based control and observation mode scheduling) strategy is a function $C : (2^O)^* \to A \times M$ that defines the action and the observation mode to be applied in the next step based only on the sequence of past observations.

We use σ_C and ρ_C to denote finite and infinite runs of the NTS $\mathcal{N}$ induced by a strategy C, respectively. Note that for every configuration (s, m), the strategy C induces a non-empty set of runs ρ_C with $\rho_C(0) = (s, m)$.

Example 1: Consider a NTS $\mathcal{N} = (S, A, T, s_{\text{init}}, AP, L)$, where $S = \{s_1, \dots, s_7\}$, $A = \{a, b\}$, $s_{\text{init}} = s_1$ and the transition function is as depicted in Fig. 1. We let $AP = \{\star\}$ and the labeling function is indicated in Fig. 1, *i.e.*, $L(s_6) = \{\star\}$ and $L(s_i) = \emptyset$ for every $i \neq 6$. Consider three observation modes $M = \{m_1, m_2, m_3\}$ for $\mathcal{N}$ such that their respective observation functions $\gamma_1, \gamma_2, \gamma_3$ report neither the shape nor the color, only the shape and both the shape and the color of the state as shown in Fig. 1. The set of observations is $O = \{\texttt{white}, \texttt{blue}, \texttt{red}, \texttt{circ}, \texttt{rect}, \texttt{diam}\}$. The costs of the observation modes are $g_1 = 0, g_2 = 1, g_3 = 2$.

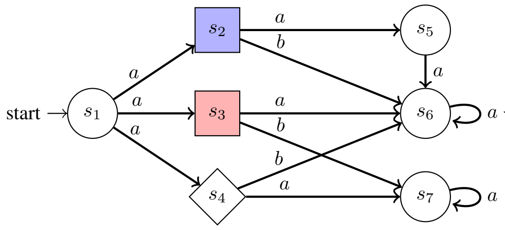

Fig. 1: Example of an NTS with observation modes. For full description see Ex. 1.

B. Specification

Linear temporal logic (LTL) is a logic with modalities referring to time [15]. Formulas of LTL are interpreted over infinite words such as the propositional traces generated by runs of a NTS. Co-safe fragment of LTL contains all LTL formulas such that every satisfying infinite word has a good finite prefix [12]. A good finite prefix is a finite word such that every its extension to an infinite word satisfies the formula. A class of co-safe LTL formulas that are easy to characterize are syntactically co-safe LTL (scLTL) formulas [12]. Intuitively, a scLTL formula φ is a LTL formula in positive normal form containing only $\mathbf{X}$ (*next*), $\mathbf{U}$ (*until*) and $\mathbf{F}$ (*future*) operators. For the detailed definition of syntax and satisfaction relation $w \models \varphi$ for words $w \in (2^{AP})^\omega$, see [12]. Formulas of scLTL can be represented with finite automata.

Definition 4 (DFA): A deterministic finite automaton (DFA) is a tuple $\mathcal{A} = (Q, 2^{AP}, \delta, q_0, F)$, where Q is a non-empty finite set of states, 2^{AP} is the alphabet, $\delta : Q \times 2^{AP} \to Q$ is a transition function, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is a non-empty set of accepting states.

A run of a DFA is a finite sequence $q_0 q_1 \dots q_n \in Q^*$ such that for every $i \geq 0$, there exists $X \in 2^{AP}$ such that $q_{i+1} = \delta(q_i, X)$. Every finite word $w \in (2^{AP})^*$ induces a run of the DFA. A run is called accepting if its last state is an accepting state. A word w is accepted by the DFA if it induces an accepting run.

Given a scLTL formula φ, one can construct a minimal (in the number of states) DFA that accepts all and only good finite prefixes of φ using a translation algorithm from [12] and automata theory techniques [16].

A run ρ of a NTS with observation modes satisfies a scLTL formula φ if $L(\rho) \models \varphi$ or, equivalently, if there exists a finite prefix ρ^φ of ρ such that $L(\rho^\varphi)$ is a good finite prefix for the formula φ. We refer to prefixes ρ^φ as the good finite prefixes of the run ρ for the formula φ. We say that a strategy C satisfies φ starting from a configuration (s, m) if $\rho_C \models \varphi$ for every run ρ_C such that $\rho_C(0) = (s, m)$.

Example 2: Consider the set of atomic propositions $AP = \{\star\}$. An example of a scLTL formula over AP is $\varphi = \mathbf{F}\,\star$. A minimal DFA $\mathcal{A}$ for φ is shown in Fig. 2.

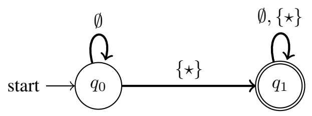
$$\begin{tikzcd}
\text{start} \arrow[r] & q_0 \arrow[loop above, "\emptyset"] \arrow[r, "\{\star\}"] & q_1 \arrow[loop above, "{\emptyset,\{\star\}}"]
\end{tikzcd}$$

Fig. 2: A minimal DFA for the scLTL formula from Ex. 2.

III. Problem formulation

The main motivation for the problem formulated in this section is a robotic system that involves uncertainty originating from the motion of the robot, *e.g.*, an autonomous car with noisy actuators as well as uncertainty originating from interaction with dynamic elements in the environment such as pedestrians on streets. Typically, the system is equipped with a set of sensors, where each sensor provides a partial information about the uncertainties. In such a case, a NTS can be used to model the motion capabilities of the robot in a partitioned environment and its interaction with the dynamic elements. The observation modes of the NTS then represent possible subsets of sensors and the cost of an observation mode corresponds to the amount of resources such as energy or communication needed to deploy the chosen set of sensors for a single step. During executions of the system, only the observations associated with the current state of the NTS are available. Hence, the current state of the system might not be uniquely recognized.

We assume that the system is given a temporal objective in the form of a scLTL formula φ over the set of atomic propositions AP. Given a starting configuration (s, m) and a strategy C that satisfies the formula φ, we define the following cost function:

$$V(C,(s,m),\varphi) = \max_{\rho_C, \rho_C(0)=(s,m)} \; \min_{\rho^\varphi \rho' = \rho_C} g(\rho^\varphi). \quad (2)$$

Intuitively, the cost $V(C,(s,m),\varphi)$ of a strategy C with respect to the formula φ and the configuration (s,m) is the worst-case cumulative cost of the (earliest) satisfaction of φ using C starting from configuration (s,m).

The problems that we consider in this work can be formulated as follows.

Problem 1 (Optimal scLTL control): Given a NTS with observation modes $(\mathcal{N}, O, M)$, where $\mathcal{N} = (S, A, T, s_{\text{init}}, AP, L)$, an initial observation mode $m_{\text{init}} \in M$, and a scLTL formula φ over AP, find a strategy C such that (1) C satisfies φ starting from the configuration $(s_{\text{init}}, m_{\text{init}})$, and (2) the cost $V(C,(s_{\text{init}}, m_{\text{init}}),\varphi)$ is minimized over all strategies satisfying φ starting from $(s_{\text{init}}, m_{\text{init}})$.

Problem 2 (Bounded optimal scLTL control): Given the same assumptions as in Prob. 1 and a finite bound $k \geq 0$, find a strategy C such that (1) C satisfies φ starting from the configuration $(s_{\text{init}}, m_{\text{init}})$ in at most k steps, and (2) the cost $V(C,(s_{\text{init}}, m_{\text{init}}),\varphi)$ is minimized over all strategies satisfying φ starting from the configuration $(s_{\text{init}}, m_{\text{init}})$ in at most k steps.

In Sec. IV, we propose an algorithm to solve the general Problem 1. The algorithm builds on techniques from automata-based model checking and graph theory. The bounded Problem 2 can be solved using an alternation of the above algorithm as proposed in Sec. V. Both algorithms are demonstrated in Sec. VI on an illustrative case study of a mobile robot in an indoor environment.

Example 3: Consider the NTS with observation modes introduced in Ex. 1 with initial observation mode m_1 and the scLTL formula from Ex. 2 that requires to reach the state labeled with $\star$, *i.e.*, state s_6. Note that only in states s_2, s_3, s_4 there is more than one action allowed and hence it suffices to discuss strategies based on their decision in these states. No strategy C with $C(\emptyset) = (a, m_1)$ can guarantee satisfaction of the formula. The reason is that the three states s_2, s_3, s_4 cannot be told apart using mode m_1 and both actions a, b always in at least one case lead to state s_7 from which s_6 cannot be reached. Consider strategy C_1 that recognizes the shape of the three states s_2, s_3, s_4, *i.e.*, $C_1(\emptyset) = (a, m_2)$, $C_1(\emptyset\{\texttt{rect}\}) = (a, m_1)$, $C_1(\emptyset\{\texttt{diam}\}) = (b, m_1)$. Strategy C_1 guarantees a visit to s_6 in at most 3 steps and its cost $V(C_1,(s_{\text{init}}, m_1),\varphi) = 1$. Alternatively, consider strategy C_2 that recognizes both the shape and the color of the three states s_2, s_3, s_4, *i.e.*, $C_1(\emptyset) = (a, m_3)$, $C_1(\emptyset\{\texttt{rect,blue}\}) = (b, m_1)$, $C_1(\emptyset\{\texttt{rect,red}\}) = (a, m_1)$, $C_1(\emptyset\{\texttt{diam,white}\}) = (b, m_1)$. Strategy C_2 guarantees a visit to s_6 in 2 steps and its cost $V(C_2,(s_{\text{init}}, m_1),\varphi) = 2$. Strategy C_1 is the solution to the optimal scLTL control Problem 1 as its cost is lower than the cost of C_2. However, if we consider the bounded optimal scLTL control Problem 2 with $k = 2$, then C_2 is the solution as C_1 may need more than 2 steps to reach s_6.

IV. Optimal scLTL control

In this section, we describe the algorithm to solve Problem 1 in detail. To approach the problem, we leverage automata-based model checking techniques that analyze the state space using graph algorithms. We first construct a synchronous product of the NTS $\mathcal{N}$ and a DFA $\mathcal{A}$ for the scLTL formula φ, where the runs of the NTS satisfying the formula can be easily identified through accepting states of the DFA. Next, to account for the non-determinism and partial observation, we use a belief construction over the product that determines the set of possible current states of the product given any finite sequence of past observations. Using graph algorithms, we construct a strategy for the belief product that guarantees a visit of an accepting state and minimizes a function derived from the costs of the associated observation modes. Finally, we map the strategy from the belief product to the original NTS and prove that the resulting strategy is a solution to Problem 1.

A. Constructing the product

Definition 5 (Product): Let $\mathcal{N} = (S, A, T, s_{\text{init}}, AP, L)$ be a NTS and $\mathcal{A} = (Q, 2^{AP}, \delta, q_0, F)$ be a DFA. The synchronous product is a tuple $\mathcal{P} = \mathcal{N} \times \mathcal{A} = (S \times Q, A, T_\mathcal{P}, (s_{\text{init}}, q_0), AP, L_\mathcal{P}, F_\mathcal{P})$, where

- $S \times Q$ is the set of states,
- A is the alphabet,
- $T_\mathcal{P} \colon S \times Q \times A \to 2^{S \times Q}$ is a transition function such that $(s', q') \in T_\mathcal{P}((s,q), a)$ if and only if $s' \in T(s,a)$ and $\delta(q, L(s)) = q'$,

- (s_{init}, q_0) is the initial state,
- $L_{\mathcal{P}} : S \times Q \to 2^{AP}$ is the labeling function such that $L_{\mathcal{P}}((s,q)) = L(s)$,
- $F_{\mathcal{P}} = \{(s,q) \mid q \in F\}$ is the set of accepting states.

We abuse the notation by using γ_α to denote the observation function of a sensor $\alpha \in \Theta$ as well as its extension to $S \times Q$, *i.e.*, $\gamma_\alpha((s,q)) = \gamma_\alpha(s)$ for all $(s,q) \in S \times Q$.

B. Constructing the belief product

The belief construction over the product follows the standard principles used for partially observable systems. Besides keeping track of the states that the product can currently be in, we also keep track of the deployed observation mode.

Definition 6 (Weighted belief product): Given a product $\mathcal{P}$ and a set of observation modes M with the initial observation mode m_{init}, we define the weighted belief product

$$\mathcal{B} = (\mathbf{B}, \mathbf{A}, T_{\mathcal{B}}, \mathbf{b}_{\text{init}}, O, F_{\mathcal{B}}, w)$$

over $\mathcal{P}$, where

- $\mathbf{B} \subseteq 2^{S \times Q}$ is the set of all belief states, where a belief state $\mathbf{b} \in \mathbf{B}$ is a set of product states such that there exists an observation mode $m \in M$ such that all states in $\mathbf{b}$ have the same observations in mode m,
- $\mathbf{A} = A \times M$ is the set of belief actions of the form $\mathbf{a} = (a, m)$, where $a \in A$ is an action of $\mathcal{P}$ and $m \in M$ is an observation mode,
- $T_{\mathcal{B}} : \mathbf{B} \times \mathbf{A} \to 2^{\mathbf{B}}$ is the transition function such that a belief state $\mathbf{b}' \in T_{\mathcal{B}}(\mathbf{b}, (a, m'))$ if and only if $\mathbf{b}'$ is the set of all product states that can be reached in one step from a state in $\mathbf{b}$ using action a and have the same observations in mode m',
- $\mathbf{b}_{\text{init}} = \{(s_{\text{init}}, q_0)\}$ is the initial state,
- $F_{\mathcal{B}} = \{\mathbf{b} \mid \mathbf{b} \subseteq F_{\mathcal{P}}\}$ is the set of accepting belief states,
- $w : \mathbf{B} \times \mathbf{A} \to \mathbb{R}_0^+$ is the weight function such that $w(\mathbf{b}, (a, m)) = g_m$.

Weighted belief product can be seen as a NTS with a set of accepting states and weighted transitions. We adopt definitions of finite and infinite runs of the belief product.

Corollary 1: From the definition of the weighted belief product $\mathcal{B}$ it follows that every finite run of $\mathcal{B}$ corresponds to exactly one finite sequence of observations in $(2^O)^*$ and at the same time, every finite sequence of observations in $(2^O)^*$ corresponds to at most one finite run of $\mathcal{B}$.

Definition 7 (Strategy): Given a weighted belief product $\mathcal{B} = (\mathbf{B}, \mathbf{A}, T_{\mathcal{B}}, \mathbf{b}_{\text{init}}, O, F_{\mathcal{B}}, w)$, a strategy for $\mathcal{B}$ is a function $C : \mathbf{B}^* \to \mathbf{A}$.

We call a strategy C memoryless if it can be defined as a function $C : \mathbf{B} \to \mathbf{A}$.

C. Constructing a strategy for the belief product

In this section, we propose an algorithm that constructs a memoryless strategy for the weighted belief product that guarantees a visit to an accepting state (if such a strategy exists) and minimizes the worst-case cumulative weight. Such a strategy then maps to a strategy for the original NTS with observation modes that solves Problem 1. The algorithm can be seen as a combination of the standard algorithm for computing winning states in non-deterministic systems [1] and Dijkstra's algorithm for computing shortest paths in a weighted graph [8].

Algorithm 1 Constructing a strategy for the weighted belief product that maps to a solution of Problem 1.

```
Input: $\mathcal{B} = (\mathbf{B}, \mathbf{A}, T_{\mathcal{B}}, \mathbf{b}_{\text{init}}, O, F_{\mathcal{B}}, w)$
Output: memoryless strategy $C$ for the belief product $\mathcal{B}$
 1: $W_0 := F_{\mathcal{B}}$
 2: $\forall \mathbf{b} \in W_0 : \mathrm{wtg}(\mathbf{b}) := 0$
    $\forall \mathbf{b} \in (\mathbf{B} \backslash W_0) : \mathrm{wtg}(\mathbf{b}) := \infty$
 3: $i := 1$
 4: while $\mathbf{b}_{\text{init}} \not\in W_i$ and exist $\mathbf{b} \in \mathbf{B} \backslash W_{i-1}$ and $\mathbf{a} \in \mathbf{A}$ such that
    $\emptyset \neq T_{\mathcal{B}}(\mathbf{b}, \mathbf{a}) \subseteq W_{i-1}$ do
 5:     $\mathbf{b}_{\min} := \bot$     $\mathbf{a}_{\min} := \bot$     $\Delta_{\min} := \infty$
 6:     for every $\mathbf{b} \in \mathbf{B} \backslash W_{i-1}$ and $\mathbf{a} \in \mathbf{A}$ such that $\emptyset \neq T_{\mathcal{B}}(\mathbf{b}, \mathbf{a}) \subseteq$
        $W_{i-1}$ do
 7:         $\Delta := \max_{\mathbf{b}' \in T_{\mathcal{B}}(\mathbf{b}, \mathbf{a})} \{w(\mathbf{b}, \mathbf{a}) + \mathrm{wtg}(\mathbf{b}')\}$
 8:         if $\Delta < \Delta_{\min}$ then
 9:             $\mathbf{b}_{\min} := \mathbf{b}$     $\mathbf{a}_{\min} := \mathbf{a}$     $\Delta_{\min} := \Delta$
10:         end if
11:     end for
12:     $W_i := W_{i-1} \cup \{\mathbf{b}_{\min}\}$
13:     $C(\mathbf{b}_{\min}) := \mathbf{a}_{\min}$
14:     $\mathrm{wtg}(\mathbf{b}_{\min}) := \Delta_{\min}$
15:     $i := i + 1$
16: end while
17: if $\mathbf{b}_{\text{init}} \in W_i$ then
18:     return $C$
19: else
20:     return no suitable strategy exists
21: end if
```

In the algorithm, we incrementally compute a value $\mathrm{wtg}(\mathbf{b})$ (weight-to-go) for every belief state $\mathbf{b}$ that is the minimum worst case weight of reaching an accepting state starting from $\mathbf{b}$. Initially, the value is 0 for accepting belief states and ∞ otherwise. We use W_i to denote the set of belief states for which the value $\mathrm{wtg}(\mathbf{b}) \neq \infty$ after i-th iteration. In i-th iteration, we consider the belief state $\mathbf{b}_{\min} \in \mathbf{B} \setminus W_{i-1}$ and its action $\mathbf{a}_{\mathbf{b}_{\min}}$ that leads to the set W_{i-1} and minimizes the worst-case sum of the weight of the action and the value wtg of a successor state. The algorithm terminates when the initial belief state $\mathbf{b}_{\text{init}}$ is added to the set W_i or when there exists no state $\mathbf{b} \in \mathbf{B} \setminus W_i$ with an action leading to W_i. If the resulting set W_i contains the initial belief state, the strategy consisting of the above actions for each belief state in W_i is returned. The algorithm is summarized in Alg. 1.

Proposition 1 (Correctness): Alg. 1 results in a strategy C for the weighted belief product such that every run under C that starts in $\mathbf{b}_{\text{init}}$ eventually visits an accepting belief state, if such a strategy exists.

Proposition 2 (Optimality): Let C be the strategy resulting from Alg. 1. Then among all strategies that guarantee a visit to an accepting belief state, C minimizes the value

$$V_{\mathcal{B}}(C, \mathbf{b}_{\text{init}}) = \max_{\rho_C, \rho_C(0) = \mathbf{b}_{\text{init}}} \quad \min_{\rho^{\text{acc}} \rho' = \rho_C} w(\rho^{\text{acc}}, C) \qquad (3)$$

where ρ^{acc} is a finite run ending in an accepting state and

$$w(\rho^{\text{acc}}, C) = \sum_{i=0}^{|\rho^{\text{acc}}|-2} w\big(\rho^{\text{acc}}(i), C(\rho^{\text{acc}}(0) \ldots \rho^{\text{acc}}(i))\big).$$

Intuitively, the value $V_{\mathcal{B}}(C, \mathbf{b})$ of a strategy C with respect to a belief state $\mathbf{b}$ is the worst-case cumulative weight of the (earliest) visit to an accepting state using C starting from $\mathbf{b}$.

D. *Constructing a strategy for the NTS*

Let $C_{\mathcal{B}}$ be the strategy for the weighted belief product $\mathcal{B}$ resulting from Alg. 1. Consider the following strategy C for the NTS $\mathcal{N}$ with observation modes. For a finite sequence of observations $\sigma_O \in (2^O)^*$, we define

$$C(\sigma_O) = C_{\mathcal{B}}(\mathbf{b}), \tag{4}$$

where $\mathbf{b}$ is the last state of the finite run $\sigma_{\mathcal{B}}$ of the belief product that corresponds to σ_O as described in Cor. 1, if such a run exists.

Theorem 1: Let $C_{\mathcal{B}}$ be the strategy resulting from Alg. 1. Then the strategy C for the NTS with observation modes constructed according to Eq. 4 is a solution to Problem 1.

Complexity. Given an scLTL formula φ the number of states of a corresponding minimal DFA $\mathcal{A}$ is in general doubly exponential in the size of the formula. However, compared to the size of the NTS, the size of the automaton typically does not play a crucial role in the overall complexity. The product $\mathcal{P}$ of the NTS $\mathcal{N}$ and $\mathcal{A}$ is then of size $\mathcal{O}(|S| \times |Q|)$. The belief product $\mathcal{B}$ involves a subset construction over the product, hence its size is in $\mathcal{O}(2^{|S| \times |Q|})$. Typically only the reachable states of both the product and the belief product are constructed in practice. With a proper choice of a data structure storing the belief product $\mathcal{B}$, Alg. 1 runs in time $\mathcal{O}(|\mathbf{B}| \cdot \log |\mathbf{B}| + |\mathbf{A}| \cdot \mathrm{dn})$, where dn is the degree of non-determinism of the NTS $\mathcal{N}$, *i.e.*, the maximum number of possible successors given a state and an action. Note that while the algorithms are polynomial with respect to their input, *i.e.*, the belief product, they are exponential in the size of the product.

V. Bounded optimal scLTL control

In order to solve Problem 2, we first construct the product $\mathcal{P}$ of the NTS $\mathcal{N}$ with a DFA $\mathcal{A}$ for the scLTL formula φ and the corresponding belief product $\mathcal{B}$ as proposed in Sec. IV-A and IV-B, respectively. To compute a strategy for the belief product from Sec. IV-C, we use an alternation of Alg. 1 presented below and summarized as Alg. 2. Intuitively, as Alg. 1 builds on the principles of Dijkstra's algorithm, Alg. 2 follows the idea behind Bellman-Ford algorithm for solving the bounded shortest path problem in weighted graphs [8]. By mapping the resulting strategy $C_{\mathcal{B}}$ for the belief product to the original system as described in Sec. IV-D, we obtain a correct and optimal solution to Problem 2.

A. *Constructing a strategy for the belief product*

Instead of computing the weight-to-go value $\mathrm{wtg}(\mathbf{b})$ for a single well-chosen belief state $\mathbf{b}$ at a time as in Alg. 1, in Alg. 2 we update the value in parallel for all states in every iteration. We show that the set W_i which is the set of all belief states for which $\mathrm{wtg}(\mathbf{b}) \neq \infty$ after i-th iteration, consists of all belief states $\mathbf{b}$ that can reach an accepting belief state in at most i steps and with the worst-case cumulative weight $\mathrm{wtg}(\mathbf{b})$. The algorithm terminates after k, but at most $|\mathbf{B}| - 1$, iterations. If the resulting set W_i contains the initial belief state, the strategy consisting of the chosen actions for each belief state in W_i is returned.

Algorithm 2 Constructing a strategy for the weighted belief product and the given bound that maps to a solution of Problem 2.

```
Input: $\mathcal{B} = (\mathbf{B}, \mathbf{A}, T_{\mathcal{B}}, \mathbf{b}_{\mathrm{init}}, O, F_{\mathcal{B}}, w)$, bound $k \geq 1$
Output: strategy $C$ for the belief product $\mathcal{B}$
 1: $W_0 := F_{\mathcal{B}}$
 2: $\forall\, \mathbf{b} \in W_0 : \mathrm{wtg}(\mathbf{b}) := 0$
    $\forall\, \mathbf{b} \in (\mathbf{B} \backslash W_0) : \mathrm{wtg}(\mathbf{b}) := \infty$
 3: $i := 1$
 4: while $i \leq k$ do
 5:     for every $\mathbf{b} \in \mathbf{B}$ do
 6:         $\mathbf{a}^{\mathbf{b}}_{\min} := \bot$     $\Delta^{\mathbf{b}}_{\min} := \mathrm{wtg}(\mathbf{b})$
 7:         for every $\mathbf{a} \in \mathbf{A}$ such that $\emptyset \neq T_{\mathcal{B}}(\mathbf{b}, \mathbf{a}) \subseteq W_{i-1}$ do
 8:             $\Delta := \max_{\mathbf{b}' \in T_{\mathcal{B}}(\mathbf{b},\mathbf{a})} \{w(\mathbf{b}, \mathbf{a}) + \mathrm{wtg}(\mathbf{b}')\}$
 9:             if $\Delta < \Delta^{\mathbf{b}}_{\min}$ then
10:                 $\mathbf{a}^{\mathbf{b}}_{\min} = \mathbf{a}$     $\Delta^{\mathbf{b}}_{\min} := \Delta$
11:             end if
12:         end for
13:     end for
14:     $W_i := W_{i-1}$
15:     for every state $\mathbf{b} \in \mathbf{B}$ do
16:         $C(\mathbf{b}) := \mathbf{a}^{\mathbf{b}}_{\min}$
17:         $\mathrm{wtg}(\mathbf{b}) := \Delta^{\mathbf{b}}_{\min}$
18:         if $\mathrm{wtg}(\mathbf{b}) < \infty$ then
19:             $W_i := W_i \cup \{\mathbf{b}\}$
20:         end if
21:     end for
22:     $i := i + 1$
23: end while
24: if $\mathbf{b}_{\mathrm{init}} \in W_i$ then
25:     return $C$
26: else
27:     return no suitable strategy exists for given bound
28: end if
```

Proposition 3 (Correctness): Alg. 2 results in a strategy C for the weighted belief product such that every run under C that starts in $\mathbf{b}_{\mathrm{init}}$ visits an accepting belief state in at most k steps, if such a strategy exists.

Proposition 4 (Optimality): Let C be the strategy resulting from Alg. 2. Then among all strategies that guarantee a visit to an accepting belief state in at most k steps, C minimizes the value in Eq. 3.

Theorem 2: Let $C_{\mathcal{B}}$ be the strategy resulting from Alg. 2. Then the strategy C for the NTS with observation modes constructed according to Eq. 4 is a solution to Problem 2.

Complexity. The size of a minimal DFA for φ, the product and the belief product are discussed in Sec. IV-D. Similarly as for Alg. 1, with a proper choice of a data structure storing the belief product $\mathcal{B}$, Alg. 2 runs in time $\mathcal{O}(k \cdot |\mathbf{B}| \cdot |\mathbf{A}| \cdot \mathrm{dn})$, where dn is the degree of non-determinism of $\mathcal{N}$, *i.e.*, the maximum number of possible successors given a state and an action. Here, the value $|\mathbf{B}| \cdot |\mathbf{A}| \cdot \mathrm{dn}$ serves as the upper bound on the number of all edges in the belief product.

VI. Case Study

We implemented the algorithms from Sec. IV and V in C++. In this section, we demonstrate their use on a case study motivated by examples in [4], [17]. All executions were performed on Mac OS X 10.10.3 with 2.6 GHz Intel Core i5 processor and 8 GB 1600 MHz DDR3 memory.

Consider a mobile robot moving in an environment partitioned into a grid of 5×5 equally sized regions. The grid contains a starting, a target and possibly multiple dangerous regions, where the robot is detected and captured. The robot

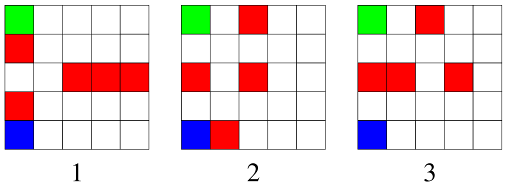

Fig. 3: The environment of a mobile robot partitioned into a grid of 5×5 regions. The three grids correspond to three possible placements of dangerous regions, shown in red. The starting and target regions are shown in green and blue, respectively.

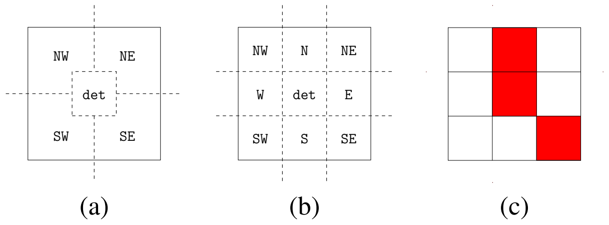

Fig. 4: Two sensors that provide information about the presence of dangerous regions in robot's immediate surroundings. We also show the names of the corresponding observations learned by the robot. For the first sensor in (a), the surrounding area is divided into quadrants and the sensor reports all quadrants containing a dangerous region. For the second sensor in (b), the exact set of dangerous regions is reported. For example, let (c) show the immediate surroundings of the robot with it's current position in the middle and dangerous regions in red. The first sensor reports the set of observations $\{\mathtt{NW}, \mathtt{NE}, \mathtt{SE}, \mathrm{det}\}$ and the second sensor reports $\{\mathtt{N}, \mathtt{SE}, \mathrm{det}\}$.

knows the locations of the starting and the target regions but it does not know the exact locations of dangerous regions. Nevertheless, the robot knows that the grid takes one of the three forms depicted in Fig. 3. The robot moves deterministically in (up to) four compass directions. To learn the presence of dangerous regions in it's immediate surroundings, the robot can deploy one of the two sensors described in Fig. 4. In every step, the robot can decide which sensor to activate, if any. The costs of deployment of the two sensors is 1 and 2, respectively. The cost can be interpreted as the amount of resources needed for the use of each sensor or as the amount of information received by the enemy. The goal of the robot is to reach the target region from the starting region without being detected, while minimizing the cost.

The NTS with observation modes that models the above system has 76 states $S = \{s_{\text{init}}, s_{ijk} \mid 1 \leq i \leq 3, 1 \leq j,k \leq 5\}$ and 5 actions $A = \{a, \mathsf{N}, \mathsf{S}, \mathsf{E}, \mathsf{W}\}$. States s_{ijk} correspond to the regions in the three grids, where $1 \leq i \leq 3$ is the grid identifier and $1 \leq j,k \leq 5$ determine the row and column coordinate, respectively. For example, s_{111} is the top left corner of the first grid. The initial state s_{init} has only one transition $T(s_{\text{init}}, a) = \{s_{111}, s_{211}, s_{311}\}$ that corresponds to the enemy choosing one of the three grids in Fig. 3. The transitions of all s_{ijk} are deterministic and correspond to moving in compass directions $\mathsf{N}, \mathsf{S}, \mathsf{E}, \mathsf{W}$. The set $AP = \{\mathtt{dang}, \mathtt{target}\}$ and the labeling function is such that $L(s_{\text{init}}) = \emptyset$ and $L(s_{ijk})$ indicates the target and dangerous regions as in Fig. 3. The set of observations is $O = \{\mathtt{N}, \mathtt{S}, \mathtt{W}, \mathtt{E}, \mathtt{NW}, \mathtt{NE}, \mathtt{SW}, \mathtt{SE}, \mathrm{det}\}$. The NTS has 3 observation modes corresponding to not activating any sensor, activating the first sensor and activating the second sensor. The respective observation functions $\gamma_1, \gamma_2, \gamma_3$ are defined in Fig. 4 and $g_1 = 0, g_2 = 1, g_3 = 2$.

The scLTL formula specifying the robot's mission is $(\neg\mathtt{dang})\,\mathbf{U}\,\mathtt{target}$ and the corresponding minimal DFA $\mathcal{A}$ has 3 states. The product $\mathcal{P}$ of $\mathcal{N}$ and $\mathcal{A}$ has 208 states and 667 (possibly non-deterministic) transitions, and was constructed in less than 0.1 seconds. The weighted belief product $\mathcal{B}$ has 375 states and 2634 transitions, and was constructed in 1.5 seconds.

The strategy solving Problem 1 was computed in 7 seconds, it reaches the target state in at most 15 steps with the worst-case cost 1. Strategies solving Problem 2 for bounds $k < 15$ were computed in less than 3 seconds each. For $k \leq 8$, there does not exist a suitable strategy. For $9 \leq k \leq 12$ and $k = 13, 14$, the optimal strategies reach the target region after at most 9 and 13 steps with the worst-case cost 1 and 2, respectively. For full description of the strategies, see [18].

References

[1] C. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[2] N. Bertrand and B. Genest. Minimal Disclosure in Partially Observable Markov Decision Processes . In *Proc. of FSTTCS*, volume 13, pages 411–422, 2011.

[3] K. Chatterjee, M. Chmelik, R. Gupta, and A. Kanodia. Optimal Cost Almost-Sure Reachability in POMDPs. In *Proc. of AAAI*, pages 3496–3502, 2015.

[4] K. Chatterjee, M. Chmelik, R. Gupta, and A. Kanodia. Qualitative analysis of POMDPs with temporal logic specifications for robotics applications. In *Proc. of ICRA*, pages 325–330, 2015.

[5] K. Chatterjee, L. Doyen, and T. A. Henzinger. A survey of partial-observation stochastic parity games. *Formal Methods in System Design*, 43(2):268–284, 2013.

[6] K. Chatterjee and R. Majumdar. Minimum Attention Controller Synthesis for Omega-Regular Objectives. In *Proc. of FORMATS*, pages 145–159, 2011.

[7] K. Chatterjee, R. Majumdar, and T. Henzinger. Controller Synthesis with Budget Constraints. In *Proc. of HSCC*, pages 72–86, 2008.

[8] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[9] Z. Feng, K. Teo, and V. Rehbock. Hybrid method for a general optimal sensor scheduling problem in discrete time. *Automatica*, 44(5):1295 – 1303, 2008.

[10] A. Jones, M. Schwager, and C. Belta. A receding horizon algorithm for informative path planning with temporal logic constraints. In *Proc. of ICRA*, pages 5019–5024, 2013.

[11] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(12):99 – 134, 1998.

[12] O. Kupferman and M. Y. Vardi. Model Checking of Safety Properties. *Formal Methods in System Design*, 19(3):291–314, 2001.

[13] P. Ondruska, C. Gurau, L. Marchegiani, C. H. Tong, and I. Posner. Scheduled Perception for Energy-Efficient Path Following. In *Proc. of ICRA*, pages 4799–4806, 2015.

[14] J. Pineau, G. Gordon, and S. Thrun. Point-based Value Iteration: An Anytime Algorithm for POMDPs. In *Proc. of IJCAI*, pages 1025–1030, 2003.

[15] A. Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science*, pages 46–57, Oct 1977.

[16] M. Sipser. *Introduction to the theory of computation*. PWS Publishing Company, 1997.

[17] M. Svorenova, M. Chmelik, K. Leahy, H. F. Eniser, K. Chatterjee, I. Cerna, and C. Belta. Temporal logic motion planning using POMDPs with parity objectives: case study paper. In *Proc. of HSCC*, pages 233–238, 2015.

[18] E. Tesařová, M. Svoreňová, J. Barnat, and I. Černá. Optimal observation mode scheduling for systems under temporal constraints. *CoRR*, abs/1602.08260, 2016.

[19] M. Vitus, W. Zhang, A. Abate, J. Hu, and C. Tomlin. On efficient sensor scheduling for linear dynamical systems. In *Proc. of ACC*, pages 4833–4838, 2010.