Constrained Control of Large Graph-based MDPs Under Measurement Uncertainty

Ravi N. Haksar, *Student Member, IEEE,* and Mac Schwager, *Member, IEEE*

***Abstract*—We consider controlling a graph-based Markov decision process (GMDP) with a control capacity constraint given only uncertain measurements of the underlying state. We also consider two special structural properties of GMDPs, called Anonymous Influence and Symmetry. Large-scale spatial processes such as forest wildfires, disease epidemics, opinion dynamics, and robot swarms are well-modeled by GMDPs with these properties. We adopt a certainty-equivalence approach and derive efficient and scalable algorithms for estimating the GMDP state given uncertain measurements, and for computing approximately optimal control policies given a maximum-likelihood state estimate. We also derive sub-optimality bounds for our estimation and control algorithms. Unlike prior work, our methods scale to GMDPs with large state-spaces and explicitly enforce a control constraint. We demonstrate the effectiveness of our estimation and control approach in simulations of controlling a forest wildfire using a model with 10^{1192} total states.**

***Index Terms*—Filtering, Markov processes, network analysis and control, stochastic optimal control, and variational methods.**

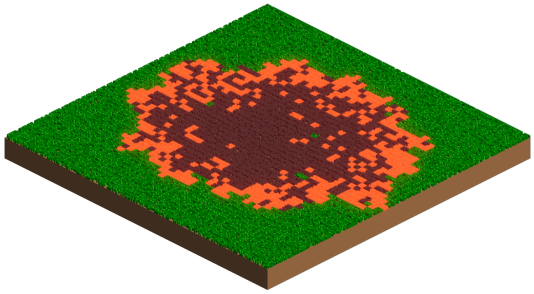


Figure 1: A forest wildfire modeled by a graph-based Markov decision process (GMDP), where green are healthy trees, red are trees on fire, and black are burnt trees. We propose a scalable framework to produce constrained control actions given noisy measurements to control large-scale spatial phenomena such as wildfires.

I. Introduction

In this work, we consider the problem of producing a control action, subject to a capacity constraint, given noisy measurements for a class of discrete space and discrete time graph-coupled Markov decision processes (GMDPs). Many large-scale, dynamic spatial processes of recent interest are described by these models, such as user interactions in a social network [1], the spread of a contagion in a population [2], and the spread of wildfire in a forest [3]. We consider controlling such large scale network models by applying localized control effort, for example supplying positive comments by a subset of users in a social network, supplying medical treatment to a subset of patients in a disease epidemic, or supplying fire retardant to a subset of trees to control a forest wildfire. Furthermore, controlling such natural phenomena is only meaningful if the total control effort applied at each instant is limited, which we call a control capacity constraint. Otherwise, the optimal unconstrained policy is straightforward, such as giving medicine to every individual at every time step to prevent a disease outbreak, or applying fire retardant to every tree at every time step to extinguish a forest wildfire.

In this work, we develop a certainty-equivalence approach to provide a single framework capable of addressing realistic spreading phenomena which naturally contain state uncertainty. To the best of our knowledge, our approach is the first framework to consider the control of GMDPs with a control constraint and measurement uncertainty.

For the graph-based models in this work, each vertex in the graph corresponds to an MDP and edges between vertices describe the coupling interactions between MDPs. In addition, a measurement model is associated with each MDP and describes the likelihood of observing different states of the MDP. While the partially observable MDPs (POMDPs) framework is appropriate for this type of model, it is difficult to develop approximately-optimal methods that are suitable for the model sizes we consider in this work using existing POMDP tools. In addition, any candidate method must also run in (near) real-time to be useful in the applications we consider here. Therefore, we adopt a certainty-equivalence approach and separate the problem into creating a filter to produce accurate state estimates, and a controller to produce effective constrained control actions given a state estimate.

We develop a fast online filtering method that is tractable for GMDPs with large state spaces by leveraging variational inference (VI) to derive a message-passing algorithm, which is similar in spirit to belief propagation (BP) methods. Prior work has proposed many variations of standard VI and BP methods for a variety of problem formulations and model assumptions. However, these methods do not scale to the model sizes we consider in this work without sacrificing real-time performance, and thus are not appropriate for our problem formulation. We prove that our filter approximately optimizes the evidence-based lower bound, and show that it achieves 5%

This research was supported by NSF grant IIS-1646921, DARPA YFA award D18AP00064, and ONR grant N00014-18-1-2830. We are grateful for this support.

R. N. Haksar is with the Department of Mechanical Engineering, Stanford University, Stanford, CA, 94305 USA (e-mail: rhaksar@stanford.edu).

M. Schwager is with the Department of Aeronautics & Astronautics, Stanford University, Stanford, CA, 94305 USA (e-mail: schwager@stanford.edu).

to 10% better accuracy in two orders of magnitude less time than comparable methods.

Our control approach, which produces actions that satisfy a global capacity constraint, is based on approximate linear programming (ALP) for MDPs. ALP methods can circumvent the explicit enumeration of the state space, which is required by standard value and policy iteration methods, by using a basis function approximation of the optimal value function for the underlying MDP problem. We first derive offline approximate dynamic programming approaches and prove that our method gives a value function with minimum deviation from the optimal value function. We then consider a class of capacity-constrained linear programs that has an efficient solution, so that constrained actions based on the value approximation can be produced online quickly.

Our framework is most appropriate for GMDPs with two properties common in large-scale spatial processes, called "Anonymous Influence" and "Symmetry." A GMDP has Anonymous Influence if the dynamics of a given MDP relies on the number of influencing MDPs in particular states, and not the identity of these influencing MDPs. Symmetry refers to the insight that value approximations for a given MDP can frequently be reused for other MDPs in the model, which greatly reduces the complexity of our methods.

Earlier versions of some of the material in this work appear previously in [3], [4]. In [3], we considered control of GMDPs with perfect state information and in [4], we considered state estimation of GMDPs without control. The current work brings these two methods together to form a closed-loop estimation and control framework for large-scale GMDPs.

The main contributions of this work are: (1) we propose a unified framework to address the constrained control of large GMDP models given only uncertain measurements; (2) we derive an approximately-optimal filtering method to produce accurate state estimates online; (3) we drive approximate dynamic programming approaches to produce a value function or a state-action function with sub-optimality bounds; (4) we propose a class of constrained linear programs with straightforward solutions to use with our value approximations; and (5) we show our approach is suitable for real-time online use, and that other methods are not scalable or appropriate for our problem formulation, on simulations of a forest wildfire model with 10^{1192} total states.

The remainder of this paper is organized as follows. Section II reviews prior work. Section III describes the GMDP framework with uncertain measurements, the Anonymous Influence and Symmetry properties, and a forest wildfire model. Sections IV and V present our certainty-equivalence framework and the derivations of our filtering and control methods. We present numerical results validating our approach in Section VI and provide concluding remarks in Section VII.

II. Prior Work

POMDPs. POMDPs are the most appropriate framework for our problem formulation and methods have been proposed for structured models, based on algebraic decision diagrams [5], [6], factored value functions [7], [8], and policy graphs [9]. Notably, the authors of [6] are able to solve models with approximately 10^6 states, but this is still much smaller than the models we wish to address. While prior work suggests some appealing approximation methods, it is not clear how to adopt them for online use. Furthermore, while exploration actions help improve the state belief, they may sacrifice some performance in controlling the process. We wish to maximize the effectiveness of the control since we aim to control large-scale natural disasters like forest wildfires and disease epidemics. Contrary to POMDP methods, which are dominated by the interleaving of filtering and control, we intentionally separate the two. This leads to our methods being able to scale up to problems with 10^{1192} possible discrete states, well beyond existing POMDP methods. We review individual control and filter methods next.

Control Methods. Traditional MDP methods are inappropriate for GMDPs as the state and action spaces of the aggregate MDP typically grow exponentially in the number of constituent MDPs. The factored MDP (FMDP) [10] as well as GMDP [11] frameworks have been formulated to compactly represent structured MDPs. Nevertheless, the compact representation does not translate to tractable exact solution methods that are analogous to traditional methods [12]. Other work [13], [14] has proposed structured policy iteration methods, but do not consider control constraints which are integral to our problem formulation.

Approximate linear programming (ALP) circumvents the explicit enumeration of the value function over the state space by using a basis function approximation. Efficient variable elimination methods have been proposed [15], [16], [17] so that all MDPs in the GMDP can be considered when computing the value function and policy. However, these approaches are applied to model sizes that are several orders of magnitude smaller than the ones we consider in this work.

Forsell et al. [11] use a specific basis function form to derive an approach that results in solving a linear program for each constituent MDP, which then results in a greedy policy that is a combination of the individual approximations. While our approach is similar in spirit, the authors do not consider control constraints or state-action functions (i.e., Q-functions), and their proposed basis approximation is not suitable for our problem formulation, as we show in our simulation experiments.

The assignment of limited control resources to a set of coupled or decoupled MDPs has also been considered in literature. Constrained MDP formulations [18] allow explicit control constraints but require traditional MDP descriptions. Approximate methods have been proposed including Lagrangian relaxation [19], [20], approximate dynamic programming [21], [22], Monte Carlo tree search [23], and receding horizon optimization [24]. However, these methods are intractable for the high-dimensional state and action spaces of GMDPs. We develop an approximate method for applying and satisfying a global capacity constraint that is tractable for large GMDPs, which is similar in spirit to [21].

Filter Methods. We consider graph-based models with state uncertainty in which the equivalent graphical model representation contains many cycles. Therefore, methods that

rely on a tree structure (e.g., belief propagation) or assume few cycles cannot be directly applied. Furthermore, we are interested in producing a full posterior distribution over states to quantify certainty in the state estimate, in contrast to a maximum-likelihood estimate (e.g., Viterbi algorithm).

Particle filters can address some issues of online inference for GMDP models [25]. However, the number of particles required for a given accuracy increases with the state dimension [26] which is generally intractable for GMDPs. Proposed approaches that have addressed this issue [27], [28], [29] are appropriate for continuous dynamical system models and not the discrete state space models we consider. Other methods [2], [30], [31] have been applied to relatively large models but do not scale to the model sizes we consider.

Variational inference (VI) methods have been applied to relatively large discrete models for inference [32], [33], [34], [35]. Notably, semi-implicit VI [35] optimizes bounds of the evidence lower bound (ELBO), but these bounds are not suitable for our approach and thus we develop our own approximation. While some methods [36] perform approximate inference for large datasets, it is unclear how to adapt them for online use as applications have been limited to relatively small models [37]. Stochastic gradient methods typically require a differentiable distribution whereas we estimate arbitrary discrete distributions. Other methods [38] are based on exploiting distribution structure which we do not require.

Belief propagation (BP) methods can be derived using variational inference with energy approximations (e.g., Bethe or Kikuchi). Loopy belief propagation (LBP) has been shown to be effective in some discrete loopy graphical models [39] and we use LBP as a benchmark method. Generalized belief propagation (GBP) improves upon LBP [40] but incurs additional (worst-case exponential) complexity and is non-trivial to apply generally. We emphasize that our approach does not use these energy approximations.

In contrast, our filtering approach uses a logarithm approximation to develop a message-passing scheme that approximates the Kullback–Leibler divergence typically used in VI methods. The result of this approach is a combination of the computational efficiency of message-passing methods with the theoretical insight and effectiveness of variational inference approaches.

In the next section, we review the GMDP framework, discuss our structural assumptions, and introduce a model to describe a forest wildfire.

III. Graph-based Markov Decision Processes

We describe the main aspects of the graph-based Markov decision process (GMDP) framework [2]. Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph with vertex set $\mathcal{V} = \{1, \dots, n\}$ containing n vertices and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each vertex $i \in \mathcal{V}$ corresponds to an MDP with latent state $x_i^t \in \mathcal{X}_i$, action $a_i^t \in \mathcal{A}_i$, and measurement $y_i^t \in \mathcal{Y}_i$ at time t. See Fig. 2 for a visualization of the GMDP structure. In a GMDP, the dynamics of MDP i are influenced by its neighbors, which are defined as follows.

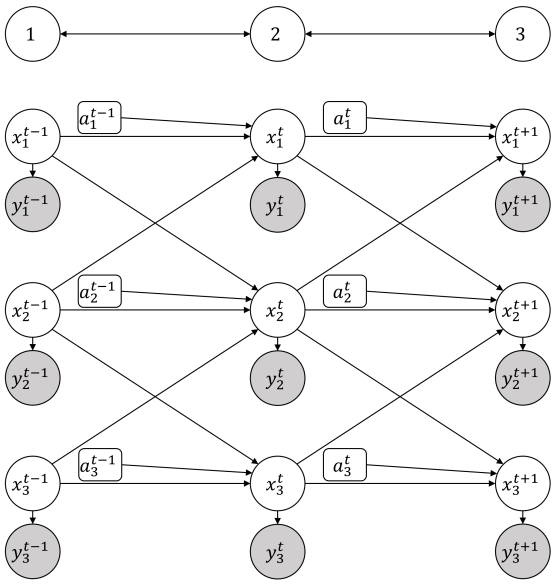


Figure 2: (top) An example GMDP consisting of three vertices, each of which represents an MDP, where arrows indicate the mutual influence between MDPs. (bottom) The underlying graphical model of the example GMDP, where arrows indicate influence between time steps.

Definition 1 (Neighbor Set)**.** The neighbor set $\mathcal{N}(\mathcal{S}) : \mathcal{S} \subseteq \mathcal{V} \to \mathcal{T} \subseteq \mathcal{V}$ is defined as,

$$\mathcal{N}(\mathcal{S}) = \bigcup_{i \in \mathcal{S}} \{j \mid (j, i) \in \mathcal{E}\}.$$

In the case that $\mathcal{S} = \{i\}$, this is the typical neighbor set of vertex i. However, our notion of a neighbor set also expresses the notion of neighbors of a set of vertices $\mathcal{S} \subseteq \mathcal{V}$. For the neighbor set of MDP i, we let $\mathcal{N}(\{i\}) = \mathcal{N}(i)$. Subscripts indicate the latent states or measurements of a subset of MDPs, for example x_i^t for MDP i and $x_{\mathcal{N}(i)}^t = \{x_j^t \mid j \in \mathcal{N}(i)\}$ for the neighbors of MDP i. We likewise use subscripts for the domain of variables, e.g., $x_{\mathcal{N}(i)}^t \in \mathcal{X}_{\mathcal{N}(i)} = \prod_{j \in \mathcal{N}(i)} \mathcal{X}_j$. We omit the subscript for the combination of all MDP states or measurements, $x^t = \{x_1^t, \dots, x_n^t\} \in \mathcal{X}$ and $y^t = \{y_1^t, \dots, y_n^t\} \in \mathcal{Y}$. Summing out (i.e., marginalizing) all MDP latent states from a distribution is denoted by $\sum_{x^t} = \sum_{x_1^t} \cdots \sum_{x_n^t}$. The marginalization of a subset of variables is specified in the summation, e.g., marginalizing out a neighbor set is $\sum_{x_{\mathcal{N}(i)}^t}$.

The probability of transitioning from a state x_i^t to x_i^{t+1} for an MDP in the GMDP model only depends on the current state of the MDP x_i^t, the state of its neighbors x_j^t in the graph, and the MDP action a_i^t. Hence the dynamics can be written compactly as,

$$p_i(x_i^{t+1} \mid x_i^t, x_{\mathcal{N}(i)}^t, a_i^t). \tag{1}$$

The dynamics for the aggregate state x^t describing the combination of all MDP states is then,

$$p(x^{t+1} \mid x^t, a^t) = \prod_{i=1}^{n} p_i(x_i^{t+1} \mid x_i^t, x_{\mathcal{N}(i)}^t, a_i^t). \tag{2}$$

Measurements for each MDP are conditionally independent given the state of the underlying MDP,

$$p_i(y_i^t \mid x_i^t),$$

and the measurement likelihood for the aggregate state is described by the distribution,

$$p(y^t \mid x^t) = \prod_{i=1}^{n} p_i(y_i^t \mid x_i^t). \tag{3}$$

Arbitrary measurement models, with $p_i(y_i^t \mid x_i^t, x_{\mathcal{M}(i)}^t)$ and $\mathcal{M}(i) \subseteq \mathcal{V}$, can be used in our framework. However, for clarity of exposition we consider each MDP measurement conditionally independent given the MDP state (i.e., $\mathcal{M}(i) = \emptyset$), and provide an additional case in the Appendix in which the MDP measurement is conditionally independent given the states of the MDP and its neighbors (i.e., $\mathcal{M}(i) = \mathcal{N}(i)$).

Finally, the reward function for the GMDP is additively composed of individual reward functions r_i which are associated with each MDP i,

$$R(x^t, a^t, x^{t+1}) = \sum_{i=1}^{n} r_i(x^t_{O(i)\cup\mathcal{N}(O(i))}, a^t_{O(i)}, x^{t+1}_{O(i)}). \tag{4}$$

Each MDP reward function is defined over a subset of variables, $O(i) \subseteq \mathcal{V}\ \forall i \in \mathcal{V}$, and typically $|O(i)| \ll |\mathcal{V}|$. Our goal in controlling the GMDP is to find a control policy to maximize the infinite-horizon discounted reward. Formally, we seek a policy $\pi(x^t)$ which maximizes,

$$J_\pi = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} R(x^t, \pi(x^t), x^{t+1}) \mid x^1, y^{1:t}\Big],$$

where the expectation is conditioned on the the history of measurements $y^{1:t}$ and the initial state x^1, and is with respect to the dynamics distribution (2) and the measurement distribution (3). A capacity constraint is enforced on the action $a^t = \pi(x^t)$, and γ is the discount factor. We emphasize that maximizing this objective requires solving a high dimensional POMDP which is intractable with typical methods, as we cannot directly observe the underlying state x^t. Therefore, we separate the problem into two stages. First, we approximate the Bayesian posterior $p(x^t \mid y^{1:t})$ and determine the maximum-likelihood state $\hat{x}^t$ of the posterior. Second, we solve the control problem assuming perfect state knowledge using the estimate $\hat{x}^t$ in place of the unknown state x^t to generate an action $a^t = \pi(\hat{x}^t)$.

A. Anonymous Influence

We consider the case where (1) is based on the number of neighbors in particular states rather than the identity of these neighbors. This property is called "Anonymous Influence" and we summarize the relevant ideas [1], [17].

We use $\mathbf{1}_j(x_i)$ to represent the indicator function which equals one when $x_i = j$ and zero otherwise. For a set of n discrete variables $x_i \in \{0, 1, \ldots, \mathcal{D} \in \mathbb{Z}_{\geq 0}\}$, the count aggregator (CA) is a vector $z \in \mathbb{Z}^{\mathcal{D}}_{\geq 0}$ where each element describes the number of variables taking on a particular value, $[z]_j = \sum_{i=1}^{n} \mathbf{1}_j(x_i)$ and $j \in \{0, 1, \ldots, \mathcal{D}\}$. A mixed-mode function (MMF) uses a CA, as well as other discrete arguments not part of a CA, and maps to the real numbers $\mathbb{R}$.

For a GMDP where all MDPs have the same discrete domain $x_i^t \in \{0, 1, \ldots, \mathcal{D}\}$, the dynamics (1) for each MDP requires specifying $(\mathcal{D}+1)^{|\mathcal{N}(i)|+1}$ values. If a CA z_i^{t-1} is used to represent the influence of other MDPs, then (1) can be represented by a MMF,

$$p_i(x_i^t \mid x_i^{t-1}, x^{t-1}_{\mathcal{N}(i)}, a_i^{t-1}) = p_i(x_i^t \mid x_i^{t-1}, z_i^{t-1}, a_i^{t-1}), \tag{5}$$

which requires specifying $(\mathcal{D}+1)\cdot\binom{|\mathcal{N}(i)|+\mathcal{D}}{|\mathcal{N}(i)|}$ values, where $\binom{n}{j}$ is the binomial coefficient, a potentially significant reduction. We illustrate this property in the discussion of our wildfire model in Section III-C.

B. Symmetry

If a GMDP consists of a comparatively small number of unique "classes" of MDPs, we say it has symmetry. Two MDPs i and j are in the same class k, denoted by $i, j \in \mathcal{C}_k$, if both MDPs have the same reward and dynamics functions. We describe how these quantities define a class in Section V.

The n MDPs in a GMDP are partitioned into $s \leq n$ unique classes, $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset\ \forall i \neq j$, and $\sum_{k=1}^{s} |\mathcal{C}_k| = n$. Many domains contain a graph topology with many MDPs but few classes. Methods that require enumerating the state space scale exponentially with the number of MDPs and quickly become intractable. In contrast, our control approach exploits Symmetry to instead scale with the number of classes.

C. Example GMDP: Forest Wildfires

We now introduce a model that describes the spread of a wildfire in a forest, to illustrate the GMDP modeling framework and our structural assumptions. The forest is modeled as a finite 2D lattice of dimensions $L \times W$ with LW total nodes; see Fig. 1. Each node $i \in \{1, \ldots, LW\}$ on the lattice represents a tree and the tree state x_i^t is one of three values, $\mathcal{X}_i = \{H, F, B\} = \{\text{healthy, on fire, burnt}\}$. An undirected graph is used to represent the trees and influence between trees, with the vertex set $\mathcal{V} = \{1, \ldots, LW\}$. Edges exist between trees if they are neighbors on the lattice. Table I summarizes the dynamics where $f_i^t = \sum_{j\in\mathcal{N}(i)} \mathbf{1}_F(x_j^t)$ is the number of neighboring trees on fire.

A tree that is healthy transitions to on fire only if at least one tree in its neighbor set is on fire where α describes the likelihood of fire spreading from a tree on fire to a healthy tree. A tree on fire will either remain on fire or transition to burnt in a single time step. The parameters β and $\Delta\beta$ describe the average number of time steps a fire will persist and the effectiveness of control actions, respectively. Control actions are binary and reflect the choice of whether or not to apply fire retardant on a tree, $a_i^t = \{0, 1\}$. Finally, a tree that is burnt will remain burnt for all time.

The measurement model for each tree is parameterized by the probability p_c, which is the probability of the ground truth tree state being observed. The other two tree states are

Table I: Tree transition probabilities for wildfire model. Blank entries are zero.

		x_i^{t+1}		
		H	F	B
x_i^t	H	$1-\alpha f_i^t$	αf_i^t	
	F		$\beta-\Delta a_i^t$	$1-\beta+\Delta\beta a_i^t$
	B			1

observed with probability $\frac{1}{2}(1-p_c)$. The measurement model is thus,

$$p(y_i^t \mid x_i^t) = \begin{cases} p_c & \text{if } y_i^t = x_i^t, \\ \frac{1}{2}(1-p_c) & \text{if } y_i^t \neq x_i^t. \end{cases} \tag{6}$$

Without mixed-mode functions (MMFs), the domain of the MDP dynamics p_i contains $|\mathcal{X}_i||\mathcal{X}_{\mathcal{N}(i)}||\mathcal{A}_i| = 6 \cdot 3^{|\mathcal{N}(i)|}$ elements and is exponential in the number of neighbors $|\mathcal{N}(i)|$. Since the probability for a healthy tree to transition to on fire depends on whether or not its neighboring trees are on fire and not the identity of these trees, a MMF reduces the representation size. With a MMF, the domain of p_i is reduced to $(|\mathcal{N}(i)|+1)|\mathcal{X}_i||\mathcal{A}_i| = 6(|\mathcal{N}(i)|+1)$ values, which is now linear in $|\mathcal{N}(i)|$.

The aggregate state space of the wildfire model has 3^{LW} state configurations. A 100 tree forest has more states than there are grains of sand on Earth and a 250 tree forest has more states than atoms in the universe [41]. Therefore, using MMFs to represent tree dynamics and exploiting Symmetry is essential to building a tractable online framework, which we describe next.

IV. Efficient Online Filter for GMDPs

We adopt a certainty-equivalence approach and decompose the problem into two parts: a process filter to produce accurate maximum-likelihood estimates of the underlying state, and a process controller to produce effective constrained actions given a state estimate. Algorithm 1 provides a high-level overview of our framework. First, a value function approximation is produced offline by solving linear programs to determine the weights of the chosen basis functions. Second, a fast filter produces online estimates of the process state, which is used in a linear program to determine a control action that satisfies a capacity constraint. In the remainder of this section, we describe our filtering approach.

Given the previous control action a^{t-1}, the GMDP can be considered a graph-based hidden Markov model (GHMM), where each node in the graph refers to a HMM (previously, an MDP). A HMM can be considered an MDP that has no action input and only noisy state information is available. Therefore, for the following discussion, we use HMM terminology in our derivation of a scalable approximate filter. The objective of a filter at a single time step is to produce the posterior distribution $p(x^t \mid y^{1:t})$ where $y^{1:t}$ is the history of measurements up

Algorithm 1 Certainty-equivalent Framework

1: Offline
2: Solve linear program(s) for basis functions weights of approximate function $V_w(x^t)$ or $Q_w(x^t)$.
3:
4: Online
5: **for** each time step t **do**
6: Get measurement.
7: Update state estimate using filter with measurement.
8: Solve linear program to get constrained control action for the state estimate.
9: Apply control action.

to time t, $y^{1:t} = \{y^1, \ldots, y^t\}$. The exact filter is derived via Bayes' rule and is a recursive relationship,

$$p(x^t \mid y^{1:t}) \propto p(y^t \mid x^t) \sum_{x^{t-1}} p(x^t \mid x^{t-1}, a^{t-1}) p(x^{t-1} \mid y^{1:t-1}),$$

which is initialized by a prior at the initial time step, $p(x^1)$. Expanding with (2) and (3), the recursive Bayesian filter (RBF) [42] for the models considered in this work is,

$$\begin{aligned} p(x^t \mid y^{1:t}) &\propto \Big(\prod_{i=1}^n p_i(y_i^t \mid x_i^t) \Big) \\ &\Big(\sum_{x^{t-1}} p(x^{t-1} \mid y^{1:t-1}) \prod_{i=1}^n p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \Big). \end{aligned} \tag{7}$$

The above expression does not simplify to a tractable form as we allow for arbitrary graph structure. In particular, the graphical model representation of the graph G may contain many cycles (or loops), as is the case for the lattice-based graph in our wildfire model. See Fig. 2 for a graphical model representation of a simple GMDP as well as its equivalent Dynamic Bayesian Network representation.

Variational inference (VI) methods formulate an optimization problem to approximate an intractable posterior distribution [43]. The RBF (7) requires $(\prod_{i=1}^n |\mathcal{X}_i|) - 1$ values to specify $p(x^{t-1} \mid y^{t-1})$ and is intractable to compute for even a single time step despite the graph structure. For example, our wildfire model with 250 total trees has 10^{119} total states. Therefore, we use VI to introduce a family of distributions $q(x^t) \in \mathcal{Q}$ to approximate the posterior $p(x^t \mid y^{1:t})$ which ideally minimizes the Kullback-Leibler (KL) divergence between the two distributions. However, directly minimizing the KL divergence is infeasible due to requiring knowledge of the posterior. Instead, a tractable optimization is maximizing the evidence lower bound (ELBO) which indirectly minimizes the KL divergence. The ELBO is,

$$\text{ELBO} = \mathbb{E}_{q(x^t)} \left[\log p(x^t, y^t \mid y^{1:t-1}) - \log q(x^t) \right], \tag{8}$$

where $\mathbb{E}_{q(x^t)}$ indicates the expectation is taken with respect to the distribution $q(x^t)$. Choosing an appropriate form for the approximating distribution $q(x^t)$ results in an optimization problem with a tractable solution, as we explain next.

We leverage the mean-field approximation where the approximating distribution is factored, $q(x^t) = \prod_{i=1}^n q_i(x_i^t)$, and

a discrete distribution (or variational factor) is associated with each HMM in the GHMM. This approximation reduces the representation size of the posterior and leads to,

$$\text{ELBO} = \sum_{x^t} \Big(\prod_{i=1}^{n} q_i(x_i^t) \Big) \log p(x^t, y^t \mid y^{1:t-1}) - \sum_{i=1}^{n} \sum_{x_i^t} q_i(x_i^t) \log q_i(x_i^t),$$

after substitution of $q(x^t)$ and algebraic simplification. A common approach, known as coordinate ascent VI, finds a local optimum by iteratively optimizing each factor $q_i(x_i^t)$ while holding others fixed. The update expression for each factor is derived by collecting the terms in the ELBO which involve factor $q_i(x_i^t)$,

$$\text{ELBO} = \sum_{x_i^t} q_i(x_i^t) \mathbb{E}_{-i} \left[\log p(x^t, y^t \mid y^{1:t-1}) \right] - \sum_{x_i^t} q_i(x_i^t) \log q_i(x_i^t) + \text{other terms}, \tag{9}$$

where $\mathbb{E}_{-i}$ refers to the expectation taken with respect to the distribution $q(x^t)$ excluding factor $q_i(x_i^t)$, i.e., $\prod_{j=1, j\neq i}^{n} q_j(x_j^t)$. For the optimization of (9) over a single factor, the "other terms" are dropped as they are constant with respect to factor $q_i(x_i^t)$. As a result, the expression in (9) can be rewritten as the following objective function,

$$\mathcal{L}_i = -D_{KL}(q_i(x_i^t) \mid\mid \exp \mathbb{E}_{-i} \left[\log p(x^t, y^t \mid y^{1:t-1}) \right]), \tag{10}$$

where D_{KL} is the KL divergence. Since the KL divergence is non-negative and equals zero when the argument distributions are identical, maximizing $\mathcal{L}_i$ leads to the closed-form update expression,

$$q_i(x_i^t) \propto \exp \mathbb{E}_{-i} \left[\log p(x^t, y^t \mid y^{1:t-1}) \right]. \tag{11}$$

We emphasize that arbitrary graph structure prevents simplification of the previous expression through structure in $\log p(x^t, y^t \mid y^{1:t-1})$.

A. Approximating the ELBO

For GHMMs with the mean-field assumption, the coordinate ascent update (11) for a single time step requires computing the joint probability,

$$\begin{aligned} &p(x^t, y^t \mid y^{1:t-1}) \\ &\propto p(y^t \mid x^t) \sum_{x^{t-1}} p(x^t \mid x^{t-1}, a^{t-1}) p(x^{t-1} \mid y^{1:t-1}) \\ &\propto \prod_{i=1}^{n} p_i(y_i^t \mid x_i^t) \sum_{x^{t-1}} \prod_{i=1}^{n} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) u_i(x_i^{t-1}), \end{aligned} \tag{12}$$

where $r(x^{t-1}) = \prod_{i=1}^{n} u_i(x_i^{t-1}) \approx p(x^{t-1} \mid y^{t-1})$ is the approximate factored prior distribution. Computing this quantity is typically intractable due to the required marginalization of all n HMMs.

Instead, a tractable computation of $\mathbb{E}_{-i} \left[p(y^t, x^t \mid y^{1:t-1}) \right]$ is possible using a message-passing scheme. We first discuss necessary approximations to the ELBO (8) and then describe the scheme . We assume the joint probability satisfies a lower bound, $p(y^t, x^t \mid y^{1:t-1}) \geq \epsilon$ for some $0 < \epsilon < 1$. Given a bound ϵ, an under-approximation to the logarithm function over the interval $[\epsilon, 1]$ is the line,

$$g(\theta) = \frac{\log \epsilon}{1 - \epsilon} (1 - \theta), \tag{13}$$

and $g(\theta) \leq \log \theta$ for $\theta \in [\epsilon, 1]$. Using (13) to approximate $\log p(x^t, y^t \mid y^{1:t-1})$ in (8) results in a surrogate ELBO,

$$\overline{\text{ELBO}} = \mathbb{E}_{q(x^t)} \left[g \left(p(x^t, y^t \mid y^{1:t-1}) \right) - \log q(x^t) \right]. \tag{14}$$

Theorem 1. *Given $\epsilon > 0$ such that $\epsilon \leq p(x^t, y^t \mid y^{1:t-1}) \leq 1$, the surrogate ELBO (14) is a lower bound to the original ELBO (14).*

Proof. The difference between the surrogate ELBO (14) and the ELBO (8) is,

$$\begin{aligned} &\text{ELBO} - \overline{\text{ELBO}} \\ &= \mathbb{E}_{q(x^t)} \left[\log p(x^t, y^t \mid y^{1:t-1}) - \log q(x^t) \right] - \\ &\qquad \mathbb{E}_{q(x^t)} \left[g \left(p(x^t, y^t \mid y^{1:t-1}) \right) - \log q(x^t) \right] \\ &= \mathbb{E}_{q(x^t)} \left[\log p(x^t, y^t \mid y^{1:t-1}) \right] - \\ &\qquad \mathbb{E}_{q(x^t)} \left[g \left(p(x^t, y^t \mid y^{1:t-1}) \right) \right] \geq 0 \\ &\Rightarrow \text{ELBO} \geq \overline{\text{ELBO}} \ \forall x^t, y^t. \end{aligned}$$

The expectation operator is linear and thus preserves the lower bound relationship of the approximation (13) to the logarithm function. The lower bound is valid for any combination of states x^t and measurements y^t as the joint probability is bounded below by ϵ. □

Maximizing the surrogate ELBO (14) over the factors $q_i(x_i^t)$ therefore indirectly maximizes the ELBO (9). Following the same derivation for the ELBO, the factor objective (10) for the surrogate ELBO is,

$$\hat{\mathcal{L}}_i = -D_{KL}(q_i(x_i^t) \mid\mid \exp \mathbb{E}_{-i} \left[g \left(p(x^t, y^t \mid y^{1:t-1}) \right) \right]).$$

The coordinate update (11) changes to,

$$\begin{aligned} q_i(x_i^t) &\propto \exp \mathbb{E}_{-i} \left[g \left(p(x^t, y^t \mid y^{1:t-1}) \right) \right] \\ &\propto \exp g \left(\mathbb{E}_{-i} \left[p(x^t, y^t \mid y^{1:t-1}) \right] \right), \end{aligned} \tag{15}$$

and the factors are now a function of the expectation of the joint probability, as desired, due to the linear approximation.

Remark. Imposing a lower bound on the joint probability (12) precludes combinations of states and observations that have zero probability of occurring. In practice, we round estimates of (12) lower than ϵ up to ϵ, which has the effect of introducing noise into the joint probability. For large GHMM models, probabilities naturally tend to zero, e.g., the aggregate state distribution (2) and observation distribution (3), since the product of probabilities less than one will approach zero. This approximation can therefore be seen as preventing the expectation in (15) from being zero for all states x_i^t prior to updating the posterior factor $q_i(x_i^t)$. In addition, after updating a posterior factor with (15), state probabilities lower than ϵ are rounded to zero before normalizing the distribution. This is used to preserve the idea that some state transitions must

be considered impossible, e.g., a healthy tree transitioning to on fire without any neighboring trees being on fire. We show through numerical simulations in Section VI that this approach is effective. Finally, ϵ is a tuning parameter and is chosen to be a small positive value to avoid excessively influencing the posterior factors.

B. Message-passing Scheme

We now build a tractable message-passing scheme to estimate the quantity $\mathbb{E}_{-i}\left[p(x^t, y^t \mid y^{1:t-1})\right]$ required in (15) for each coordinate update. Substituting (12) leads to,

$$\begin{aligned}\mathbb{E}_{-i}\left[p(x^t, y^t \mid y^{1:t-1})\right] \propto \\ \sum_{\{x_j^t | j\in\mathcal{V}, j\neq i\}} \Big(\prod_{\substack{j=1\\ j\neq i}}^{n} q_j(x_j^t)\Big)\Big(\prod_{i=1}^{n} p_i(y_i^t \mid x_i^t)\Big) \\ \Big(\sum_{x^{t-1}} \prod_{i=1}^{n} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) u_i(x_i^{t-1})\Big).\end{aligned} \tag{16}$$

The message-passing scheme works as follows. Each HMM in the GHMM maintains an estimate of its posterior factor, $q_i^k(x_i^t)$, and an estimate of (16), $E_i^k(x_i^t)$; the superscript k on these quantities, and other quantities below, refers to the k^{th} estimate. For each iteration k of the scheme, each HMM i receives messages from the neighbor HMMs $j \in \mathcal{N}(i)$ and generates estimate $E_i^k(x_i^t)$. This estimate then updates the posterior factor using (15). Lastly, an updated message is calculated for the next iteration $k+1$.

The development of a message-passing scheme requires recognizing a recursive structure in the information that must be shared for each HMM to compute (12). We illustrate this structure and derive the required messages by describing the first two iterations of the scheme for a given HMM i.

The first iteration E_i^1 considers information from the neighbor HMMs $j \in \mathcal{N}(i)$,

$$\begin{aligned}E_i^1(x_i^t) \propto \Big[p_i(y_i^t \mid x_i^t)\Big] \sum_{x_i^{t-1}} u_i(x_i^{t-1}) \\ \Big[\sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} u_j(x_j^{t-1})\Big].\end{aligned} \tag{17}$$

The second iteration E_i^2 considers information from the neighbors $\mathcal{N}(i)$ and the neighbors of neighbors $\bigcup_{j\in\mathcal{N}(i)}\mathcal{N}(j)$,

$$\begin{aligned}E_i^2(x_i^t) \propto\ p_i(y_i^t \mid x_i^t) \sum_{x_i^{t-1}} u_i(x_i^{t-1}) \\ \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} u_j(x_j^{t-1}) \sum_{x_j^t} q_j^1(x_j^t) \\ \Big[p_j(y_j^t \mid x_j^t) \sum_{x_{\mathcal{N}(j)}^{t-1}} p_j(x_j^t \mid x_j^{t-1}, x_{\mathcal{N}(j)}^{t-1}, a_j^{t-1}) \prod_{l\in\mathcal{N}(j)} u_l(x_l^{t-1})\Big].\end{aligned} \tag{18}$$

Note that the above expression assumes that HMM i is not a part of the neighbors of HMMs $j \in \mathcal{N}(i)$, i.e., that $i \notin \mathcal{N}(j)\ \forall j \in \mathcal{N}(i)$. This is a simplifying assumption to approximately include information from the neighbor set, as we do not assume any simplifying structure; see the Remark at the end of Section IV-C. There is a common structure in the first (17) and second (18) iterations, as indicated by the large brackets in both expressions. Define d_i^1 as the following quantity,

$$\begin{aligned}d_i^1(x_i^{t-1}, x_i^t) = p_i(y_i^t \mid x_i^t) \\ \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} u_j(x_j^{t-1}).\end{aligned} \tag{19}$$

The second iteration (17) then simplifies to,

$$E_i^1(x_i^t) \propto \sum_{x_i^{t-1}} u_i(x_i^{t-1}) d_i^1(x_i^{t-1}, x_i^t). \tag{20}$$

In addition, the second iteration (18) simplifies to,

$$\begin{aligned}E_i^2(x_i^t) \propto p_i(y_i^t \mid x_i^t) \sum_{x_i^{t-1}} u_i(x_i^{t-1}) \\ \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \\ \prod_{j\in\mathcal{N}(i)} \Big[u_j(x_j^{t-1}) \sum_{x_j^t} q_j^1(x_j^t) d_j^1(x_j^{t-1}, x_j^t)\Big],\end{aligned} \tag{21}$$

by using information computed by the neighbor HMMs $j \in \mathcal{N}(i)$ during the first iteration of the scheme, d_j^1. Notably, the simplified second iteration (21) now only requires information from the neighbors $\mathcal{N}(i)$, as indicated by brackets. If the neighbors produce a message to share,

$$m_j^1(x_j^{t-1}) \propto u_j(x_j^{t-1}) \sum_{x_j^t} q_j^1(x_j^t) d_j^1(x_j^{t-1}, x_j^t),$$

then the second iteration (21) further simplifies,

$$\begin{aligned}E_i^2(x_i^t) \propto \Big[p_i(y_i^t \mid x_i^t)\Big] \sum_{x_i^{t-1}} u_i(x_i^{t-1}) \\ \Big[\sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} m_j^1(x_j^{t-1})\Big].\end{aligned} \tag{22}$$

Finally, (22) shares a common structure with the first iteration (17), as indicated by brackets. If d_i^2 is defined as the following quantity for the second iteration,

$$\begin{aligned}d_i^2(x_i^{t-1}, x_i^t) = p_i(y_i^t \mid x_i^t) \\ \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} m_j^1(x_j^{t-1}),\end{aligned} \tag{23}$$

then the second iteration E_i^2 simplifies again to,

$$E_i^2(x_i^t) \propto \sum_{x_i^{t-1}} u_i(x_i^{t-1}) d_i^2(x_i^{t-1}, x_i^t),$$

which mirrors the form of the simplified first iteration (20). By initializing the messages as the prior for all HMMs,

$m_i^0(x_i^{t-1}) = u_i(x_i^{t-1})$, the quantity d_i^k can be written generally as,

$$d_i^k(x_i^{t-1}, x_i^t) = p_i(y_i^t \mid x_i^t) \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j \in \mathcal{N}(i)} m_j^{k-1}(x_j^{t-1}). \quad (24)$$

Subsequent iterations E_i^k, $k \geq 3$ continue to incrementally add influence from additional HMMs in the GHMM to improve the estimate of (16).

Using the general form of d_i^k, the estimate of the joint probability computed by HMM i at iteration k is,

$$\begin{aligned} E_i^k(x_i^t) &\propto \sum_{x_i^{t-1}} u_i(x_i^{t-1}) d_i^k(x_i^{t-1}, x_i^t) \\ &\approx \mathbb{E}_{-i}\left[p(x^t, y^t \mid y^{1:t-1})\right], \end{aligned} \quad (25)$$

and $q_i^k(x_i^t)$ is updated by using the above estimate with (15). Lastly, the updated message that is shared by each HMM at the next iteration is,

$$m_i^k(x_i^{t-1}) \propto u_i(x_i^{t-1}) \sum_{x_i^t} q_i^k(x_i^t) d_i^k(x_i^{t-1}, x_i^t). \quad (26)$$

Using (13) moves the expectation into the argument of (15) which allows the posterior factors to be used for marginalization of (16). This property is key for creating a tractable message-passing method. Otherwise, the quantity $\mathbb{E}_{-i}\left[\log p(x^t, y^t \mid y^{1:t-1})\right]$ is not tractable to compute as we do not allow for simplification based on the graph structure of the GHMM, on the properties of distributions for the posterior factors, or other properties. In addition, the estimates (25) are normalized to estimate the normalization constant of the joint probability (12), since the approximation (13) does not allow this constant to be factored out.

The previous derivation is based on a model where each MDP measurement is conditionally independent of other MDPs given its own state, i.e., measurement models of the form $p(y_i^t \mid x_i^t)$. In general, MDP measurements may be influenced by other MDPs as well, and we provide a derivation for the model $p(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t)$ in the Appendix to consider a broader class of GMDP models with state uncertainty.

C. Simplifying with Anonymous Influence

Computing d_i^k (24) may be intractable as marginalizing out $x_{\mathcal{N}(i)}^{t-1}$ requires considering $\prod_{j \in \mathcal{N}(i)} |\mathcal{X}_j|$ values. If a HMM has many neighbors (large $|\mathcal{N}(i)|$) or if the neighbors have large state spaces $|\mathcal{X}_j|$ then the computational cost may be significant. Therefore, we now exploit Anonymous Influence to address this potential issue. If the dynamics (1) of a HMM rely on a count aggregator (CA) (as shown in (5)), then it is useful to create a mixed-mode function (MMF) $\tilde{m}_i^{k-1}(z_i^{t-1})$ to represent the received neighbor messages. Using this MMF leads to the modified form of d_i^k,

$$d_i^k(x_i^{t-1}, x_i^t) = p_i(y_i^t \mid x_i^t) \sum_{z_i^{t-1}} p_i(x_i^t \mid x_i^{t-1}, z_i^{t-1}, a_i^{t-1}) \tilde{m}_i^{k-1}(z_i^{t-1}). \quad (27)$$

Algorithm 2 Relaxed Anonymous Variational Inference (RAVI) for time step t

1: **Input:** prior factors $u_i(x_i^{t-1})$, graph G, actions a_i^{t-1}, dynamics $p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1})$, measurements y_i^t and models $p_i(y_i^t \mid x_i^t)$
2: **Output:** posterior factors $q_i(x_i^t)$
3: **Parameters:** iteration limit $K_{\max}$, lower bound ϵ, convergence criteria
4: **for** each vertex i **do**
5: initialize message $m_i^0(x_i^{t-1}) = u_i(x_i^{t-1})$
6: initialize factor $q_i^0(x_i^t)$
7: **for** iteration $k = 1, \ldots, K_{\max}$ **do**
8: **for** each vertex $i \in \mathcal{V}$ **do**
9: Receive messages $\{m_j^{k-1}(x_j^{t-1}) \mid j \in \mathcal{N}(i)\}$
10: Compute $d_i^k(x_i^{t-1}, x_i^t)$ with (24) or (27)
11: Estimate $E_i^k(x_i^t)$ using (25)
12: Update $q_i^k(x_i^t)$ by (15)
13: Compute $m_i^k(x_i^{t-1})$ with (26)
14: **if** factors $q_i^k(x_i^t)$ converge **then** terminate early
15: **return** posterior factors $q_i(x_i^t) = q_i^k(x_i^t)$

The marginalization for (27) is now with respect to z_i^{t-1} which has lower computational cost.

Algorithm 2, Relaxed Anonymous Variational Inference (RAVI), summarizes the approximate filter for a single time step. The factors $q_i(x_i^t)$ are then used as the priors $u_i(x_i^t)$ for the next time step. The main component is the message-passing scheme, which is relatively straightforward to implement. The posterior factors are initialized to any valid discrete distribution (line 6) and the algorithms runs for a fixed number of iterations $K_{\max}$ unless the factors converge (line 14).

Remark. Our filtering approach is based on two key approximations, an approximate lower bound on the joint probability (12) and a message-passing scheme to approximate the expectation of the joint probability (16). Variational Inference techniques commonly rely on approximations and simplifications, such as other bounds of the ELBO [35], the mean-field approximation [43], conjugate distributions [44], or the presence of tractable substructures [45]. These approximations are necessary to reduce the optimization of the ELBO to a tractable optimization. Furthermore, Loopy Belief Propagation (LBP) approximates computing marginals on a cyclic graph with a message-passing scheme, and has been shown to be accurate in a variety of applications [39]. Our focus in this work is to develop a probabilistic approach with a focus on scalability and performance, which we demonstrate with our simulation results in Section VI.

V. Scalable Constrained Control of GMDPs

We now derive approximately optimal controllers that use a maximum-likelihood estimate from our approximate filter to produce a constrained control action. We present two approaches, one based on approximate value functions, and another based on approximate state-action functions. The use of approximate value functions is more common in prior

work, but its more difficult to enforce a capacity constraint on the control action. In contrast, state-action functions are less studied in relevant prior work, but are easier to use with our capacity-constrained formulations, as we discuss in Section V-C. We present both approaches to provide a broader set of tools for controlling large-scale natural phenomena.

We assume binary actions, $a_i^t \in \{0,1\}\ \forall i \in \mathcal{V}$, and enforce a capacity constraint where the feasible action set at each time step is,

$$\mathcal{A}_c = \{a^t \in \mathcal{A} \mid \sum_{i=1}^{n} a_i^t \leq C\}, \tag{28}$$

and $C \in \mathbb{Z}_{\geq 0}$ is the maximum allowed capacity. We first derive an approach based on approximate value functions.

A. Approximate Value Functions

We consider approximate value functions which are a sum of local basis functions,

$$V_w(x^t) = \sum_{i=1}^{n} w_i^T h_i(x_{O(i)}^t), \tag{29}$$

which mirrors the structure of the reward function (4), where $w_i \in \mathbb{R}^{k_i}$ and $h_i : \mathcal{X}_{O(i)} \mapsto \mathbb{R}^{k_i}$. Each basis function h_i typically relies on state information from a few MDPs, $O(i) \subseteq \mathcal{V}$ and $|O(i)| \ll |\mathcal{V}|$. The purpose of this basis representation is to leverage the additive structure of the reward function and to greatly reduce the complexity of solving a linear program to determine the approximate value function. We use the following bound from [46] to derive a tractable method for solving for the weights of the value function, and for determining the approximation error relative to the optimal value function.

Proposition 1 (Value function approximation error [46])**.** *The maximum difference between an approximate value function $V_w(x^t)$ and the optimal value function $V^\star(x^t)$ is $\delta = \max_{x^t \in \mathcal{X}} |V_w(x^t) - V^\star(x^t)|$ and is bounded,*

$$\delta \leq \frac{2\gamma}{1-\gamma} \max_{x^t \in \mathcal{X}} |V_w(x^t) - (\mathcal{B}V_w)(x^t)|,$$

where $(\mathcal{B}V)(x^t)$ is the Bellman operator for value functions,

$$(\mathcal{B}V)(x^t) = \max_{a^t \in \mathcal{A}} \mathbb{E}_p \left[R(x^t, a^t, x^{t+1}) + \gamma V(x^{t+1}) \right],$$

and the expectation is taken with respect to the dynamics model $p(x^{t+1} \mid x^t, a^t)$.

This bound is useful as the right hand side (R.H.S.) involves quantities that can be approximated and minimizing the R.H.S. explicitly minimizes the approximation error relative to the optimal value function. In the following discussion, we occasionally omit the argument of functions for clarity. Minimizing $\phi = \max_{x^t \in \mathcal{X}} |V_w(x^t) - (\mathcal{B}V_w)(x^t)|$ leads to the non-linear program,

$$\begin{aligned} \min_{\substack{w_i \in \mathbb{R}^{k_i} \\ \phi \in \mathbb{R}}} \quad & \phi \\ \text{s. t.} \quad & \phi \geq V_w(x^t) - (\mathcal{B}V_w)(x^t), \\ & \phi \geq (\mathcal{B}V_w)(x^t) - V_w(x^t), \forall x^t \in \mathcal{X}, \end{aligned} \tag{30}$$

where the Bellman operator for the models considered in this work is,

$$\begin{aligned} (\mathcal{B}V_w)(x^t) &= \max_{a^t \in \mathcal{A}_c} \mathbb{E}_p \left[\sum_{i=1}^{n} r_i + \gamma w_i^T h_i(x_{O(i)}^{t+1}) \right] \\ &= \max_{a^t \in \mathcal{A}_c} \sum_{i=1}^{n} \mathbb{E}_p \left[r_i + \gamma w_i^T h_i(x_{O(i)}^{t+1}) \right], \end{aligned}$$

with the expectation taken with respect to the aggregate dynamics model (2). Computing operations over the full state space and the feasible action set is intractable so we develop upper and lower bounds, $(\underline{\mathcal{B}V_w})(x^t) \leq (\mathcal{B}V_w)(x^t) \leq (\overline{\mathcal{B}V_w})(x^t)$. With these bounds, the following constraints are imposed,

$$\begin{aligned} &\phi \geq V_w(x^t) - (\underline{\mathcal{B}V_w})(x^t) \geq V_w(x^t) - (\mathcal{B}V_w)(x^t), \\ &\phi \geq (\overline{\mathcal{B}V_w})(x^t) - V_w(x^t) \geq (\mathcal{B}V_w)(x^t) - V_w(x^t), \forall x^t \in \mathcal{X}, \end{aligned}$$

and the original non-linear program constraints are still satisfied. Let the expected immediate reward and future value for every MDP be,

$$g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, a_{O(i)}^t) = \mathbb{E}_p \left[r_i + \gamma w_i^T h_i \right].$$

A lower bound is any action that satisfies the constraint. We use $a_i^t = 0\ \forall i \in \mathcal{V}$ and denote this action $\overline{a}^t$,

$$(\underline{\mathcal{B}V_w}) = \sum_{i=1}^{n} g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, \overline{a}_{O(i)}^t).$$

An upper bound is an over-approximation of the constrained Bellman operator by removing the capacity constraint and instead maximizing over the set of actions for each summand,

$$(\overline{\mathcal{B}V_w}) = \sum_{i=1}^{n} \max_{a_{O(i)}^t} g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, a_{O(i)}^t).$$

Removing the constraint over-approximates the value of each MDP but is critical in dividing the original intractable non-linear program into n tractable programs. The constraints in (30) simplify to,

$$\begin{aligned} &\phi \geq \sum_{i=1}^{n} -w_i^T h_i + \max_{a_{O(i)}^t} g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, a_{O(i)}^t), \\ &\phi \geq \sum_{i=1}^{n} w_i^T h_i - g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, \overline{a}_{O(i)}^t), \forall x^t \in \mathcal{X}. \end{aligned} \tag{31}$$

We now decompose the approximation error, $\phi = \sum_{i=1}^{n} \phi_i$, to impose the following constraints instead,

$$\begin{aligned} &\phi_i \geq -w_i^T h_i + \max_{a_{O(i)}^t} g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, a_{O(i)}^t), \\ &\phi_i \geq w_i^T h_i - g_i(x_{O(i)\cup\mathcal{N}(O(i))}^t, \overline{a}_{O(i)}^t), \forall x_{O(i)\cup\mathcal{N}(O(i))}^t. \end{aligned}$$

This over-approximates ϕ by adding structure to the error contribution of each MDP but reduces the coupled non-linear program into n separate non-linear programs. In addition, the constraints in (31) are still satisfied after adding this structure.

The maximum operator is then replaced by adding a constraint for each action which results in the linear program,

$$\begin{aligned} \min_{\substack{w_i\in\mathbb{R}^{k_i}\\ \phi_i\in\mathbb{R}}} \quad & \phi_i \\ \text{s. t.} \quad & \phi_i \geq -w_i^T h_i(x^t_{O(i)}) + g_i(x^t_{\mathcal{N}(O(i))}, a^t_{O(i)}), \\ & \quad \forall x^t_{O(i)\cup\mathcal{N}(O(i))}, a^t_{O(i)}. \\ & \phi_i \geq w_i^T h_i(x^t_{O(i)}) - g_i(x^t_{O(i)\cup\mathcal{N}(O(i))}, \overline{a}^t_{O(i)}), \\ & \quad \forall x_{O(i)\cup\mathcal{N}(O(i))}. \end{aligned} \tag{32}$$

Each program contains $k_i + 1$ variables and $|\mathcal{X}_{O(i)\cup\mathcal{N}(O(i))}|\left(|\mathcal{A}_{O(i)}|+1\right)$ constraints, and there is one linear program associated with each MDP in the GMDP. The quantity $\phi = \sum_{i=1}^n \phi_i$ is a sub-optimality estimate of V_w compared to the optimal value function $V^\star$. Furthermore, we can exploit Symmetry to reduce the number of programs that must be solved, as typically multiple MDPs will have identical solutions ϕ_i, w_i. In particular, we use the same basis approximation for all MDPs in the same equivalence class.

Theorem 2. *For a GMDP containing $s \leq n$ equivalence classes, the value function ALP method requires solving s linear programs and $\sum_{k=1}^s |\mathcal{C}_k|\phi_k$ is the sub-optimality error.*

Proof. The constraints in Program (32) are uniquely defined by the reward function r_i, dynamics p_i, and basis approximation $w_i^T h_i$. By definition, all MDPs in the same equivalence class have identical reward functions r_i and dynamics p_i. Therefore, using the same basis approximation $w_i^T h_i$ for all MDPs in the same equivalence class results in identical Programs (32). As a result, for n equivalence classes, the solution to each program is unique as no two MDPs share the same class. When there are $s < n$ classes, there are at most s unique solutions for all n linear programs. In this case, only one linear program per equivalence class must be solved as the solution for the per-MDP program (32) is identical for all MDPs within a class. □

B. Approximate State-Action Functions

We now derive a novel ALP approach to produce a state-action function (i.e., a Q-function) to approximate a constrained value function. Our approximate state-action function approach is based on the following bound from [46].

Proposition 2 (State-action function approximation error [46])**.** *The difference $\delta = \max_{x^t\in\mathcal{X}} |V_w(x^t) - V^\star(x^t)|$ is bounded by the maximum difference between the approximate state-action function $Q_w(x^t, a^t)$ and the Bellman operator on $Q_w(x^t, a^t)$,*

$$\delta \leq \frac{2}{1-\gamma} \max_{x^t\in\mathcal{X}, a^t\in\mathcal{A}} |Q_w(x^t, a^t) - (\mathcal{B}Q_w)(x^t, a^t)|,$$

where $(\mathcal{B}Q)(x^t, a^t)$ is the Bellman operator for state-action functions,

$$(\mathcal{B}Q)(x^t, a^t) = \mathbb{E}_p\left[R(x^t, a^t, x^{t+1}) + \gamma \max_{a^{t+1}\in\mathcal{A}} Q(x^{t+1}, a^{t+1})\right].$$

We assume the approximate state-action function Q_w form,

$$Q_w(x^t, a^t) = \sum_{i=1}^n w_i^T b_i(x^t_{O(i)}) + a_i^t w_i^T c_i(x^t_{O(i)}), \tag{33}$$

with $w_i \in \mathbb{R}^{k_i}$ and $b_i, c_i : \mathcal{X}_{O(i)} \subseteq \mathcal{X} \mapsto \mathbb{R}^{k_i}$, specifically for our capacity constrained formulations which we describe in the next section. Minimizing $\phi = \max_{x^t\in\mathcal{X}, a^t\in\mathcal{A}_c} |Q_w(x^t, a^t) - (\mathcal{B}Q_w)(x^t, a^t)|$ results in the non-linear program,

$$\begin{aligned} \min_{\substack{w_i\in\mathbb{R}^{k_i}\\ \phi\in\mathbb{R}}} \quad & \phi \\ \text{s. t.} \quad & \phi \geq Q_w(x^t, a^t) - (\mathcal{B}Q_w)(x^t, a^t) \\ & \phi \geq (\mathcal{B}Q_w)(x^t, a^t) - Q_w(x^t, a^t), \\ & \quad \forall x^t \in \mathcal{X}, a^t \in \mathcal{A}_c, \end{aligned}$$

where the non-linearity is due to the maximization in the Bellman operator. We follow a similar procedure as before to develop a tractable and scalable method. The constrained Bellman operator is,

$$(\mathcal{B}Q)(x^t, a^t) = \mathbb{E}_p\Bigg[\sum_{i=1}^n r_i(x^t_{O(i)\cup\mathcal{N}(O(i))}, a^t_{O(i)}, x^{t+1}_{O(i)}) + \dots \\ \gamma \max_{a^{t+1}\in\mathcal{A}_c} \sum_{i=1}^n w_i^T b_i(x^{t+1}_{O(i)}) + a_i^{t+1} w_i^T c_i(x^{t+1}_{O(i)})\Bigg].$$

We construct upper and lower bounds for the constrained Bellman operator to instead impose the constraints,

$$\phi \geq Q_w(x^t, a^t) - (\underline{\mathcal{B}Q_w})(x^t, a^t),$$
$$\phi \geq (\overline{\mathcal{B}Q_w})(x^t, a^t) - Q_w(x^t, a^t), \forall x^t \in \mathcal{X}, a^t \in \mathcal{A}_c.$$

Function arguments are omitted at times for clarity in the following discussion. A lower bound is any action a^{t+1} that satisfies the control constraint. A convenient choice is $a_i^{t+1} = 0\ \forall i \in \mathcal{V}$ thus,

$$\left(\underline{\mathcal{B}Q_w}\right) = \sum_{i=1}^n \mathbb{E}_p\left[r_i + \gamma w_i^T b_i\right].$$

An upper bound is found by removing the capacity constraint and choosing actions to improve the total value of Q_w,

$$\left(\overline{\mathcal{B}Q_w}\right) = \sum_{i=1}^n \mathbb{E}_p\Big[r_i + \gamma w_i^T b_i + \gamma \max\{0, w_i^T c_i\}\Big].$$

The maximization in the upper bound is replaced by two linear constraints and the error ϕ is decomposed as a sum $\sum_{i=1}^n \phi_i$. The result is the following per-MDP linear program,

$$\begin{aligned} \min_{\substack{w_i\in\mathbb{R}^{k_i}\\ \phi_i\in\mathbb{R}}} \quad & \phi_i \\ \text{s. t.} \quad & \phi_i \geq w_i^T b_i + a_i^t w_i^T c_i - \mathbb{E}_p\left[r_i + \gamma w_i^T b_i\right], \\ & \phi_i \geq \mathbb{E}_p\left[r_i + \gamma w_i^T b_i\right] - w_i^T b_i - a_i^t w_i^T c_i, \\ & \phi_i \geq \mathbb{E}_p\left[r_i + \gamma w_i^T b_i + \gamma w_i^T c_i\right] - w_i^T b_i - a_i^t w_i^T c_i, \\ & \quad \forall x^t_{O(i)\cup\mathcal{N}(O(i))}, a^t_{O(i)}. \end{aligned} \tag{34}$$

Each program contains $k_i + 1$ variables and $3|\mathcal{X}_{O(i)\cup\mathcal{N}(O(i))}||\mathcal{A}_{O(i)}|$ constraints. Solving Program (34) for each MDP does not enforce that the total control effort will

satisfy the capacity constraint. However, allowing infeasible actions results in a more conservative approximation of the true constrained value function since adding constraints to Program (34) cannot lower the error ϕ_i. Similar to Theorem 2, we again exploit Symmetry to reduce the number of linear programs that must be solved, by using the same basis approximation for all MDPs in the same equivalence class.

Theorem 3. *For a GMDP containing $s \leq n$ unique equivalence classes, the approximate state-action function Q_w ALP method requires solving s linear programs and $\sum_{k=1}^{s} |\mathcal{C}_k| \phi_k$ is the approximation error.*

Proof. The approximate state-action function Q_w is determined after solving for the weights w_i. The Program (34) for two MDPs have identical solutions w_i, ϕ_i if both are in the same equivalence class and the same basis functions are used. Therefore, for n classes, n programs are solved to determine Q_w. Only s programs are solved for $s < n$ classes as the solution is identical for all MDPs in the same class. □

C. Capacity Constrained Programs

In the previous two sections, we derived methods for constructing an approximate value or state-action function. However, it is still necessary to extract a policy, which may be non-trivial given the $\binom{n}{C}$ possible feasible constrained actions. Therefore, we now introduce a class of linear programs that have a capacity constraint and an explicit solution, and then discuss building a policy for an approximate value or state-action function.

Proposition 3 (Capacity Constrained Linear Program)**.** *The integer linear program,*

$$\max_{a^t \in \mathcal{A}_c} \ \lambda + \sum_{i=1}^{n} \mu_i a_i^t, \tag{35}$$

where $\lambda, \mu_i \in \mathbb{R}$, has an explicit solution. Assume that $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$. An optimal solution is,

$$a_i^t = \begin{cases} 1 & \text{if } i \leq C \text{ and } \mu_i \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

Proof. The values μ_i can always be sorted a priori. The solution is optimal as changing it cannot improve the objective. Consider an optimal solution described as $a_i^t = 1$ for $i \in \{1, \ldots, j\}$ with $j \leq C$ and zero otherwise. If $j = C$, then choosing $a_k^t = 1$ for any $k > j$ will violate the constraint. If $j < C$, then choosing $a_k^t = 1$ for $j < k \leq C$ lowers the objective as μ_k must be negative. Finally, switching $a_k^t = 1$ to $a_k^t = 0$ for $k \leq j$ does not improve the objective as μ_k must be non-negative. □

We now discuss the conditions under which our approximate functions result in the policy described by (36).

Theorem 4. *For approximate value functions V_w, if the form of the dynamics (1), reward functions (4), and basis functions (29) result in,*

$$\mathbb{E}_p \left[R(x^t, a^t, x^{t+1}) + \gamma V_w(x^{t+1}) \right] = \lambda + \sum_{i=1}^{n} \mu_i a_i^t, \tag{37}$$

then the constrained policy is determined by (36). Furthermore, for approximate state-action functions of the form in (33), the constrained policy is determined by (36).

Proof. The approximate value function is determined after solving for the weights of the basis function representation. If the relationship in (37) holds, the constrained policy is,

$$\pi(x^t) = \arg\max_{a^t \in \mathcal{A}_c} \ \lambda + \sum_{i=1}^{n} \mu_i a_i^t.$$

Therefore, the policy $\pi(x^t)$ is determined by (36). For approximate state-action functions, if the assumed form (33) is used then the constrained policy is,

$$\pi(x^t) = \arg\max_{a^t \in \mathcal{A}_c} \ \sum_{i=1}^{n} w_i^T b_i(x_{O(i)}^t) + a_i^t w_i^T c_i(x_{O(i)}^t). \tag{38}$$

Let $\lambda = \sum_{i=1}^{n} w_i^T b_i(x_{O(i)}^t)$ and $\mu_i = w_i^T c_i(x_{O(i)}^t)$. The constrained maximization (38) is equivalent to Program (35) and so the policy is determined by (36). □

Although the policy (36) exactly solves Program (35), it is necessary in our derivations to remove the capacity constraint when determining an approximate value function V_w or state-action function Q_w to develop tractable methods. As a result, the approximate functions are analogous to unconstrained solutions found via dynamic programming. The use of (36) with an unconstrained approximate solution is therefore an approximation to the true constrained policy determined by a method that explicitly includes the control constraint.

D. Simplifying with Anonymous Influence

Theorems 2 and 3 describe how we leverage Symmetry to reduce the total number of linear programs that must be solved to fully determine an approximate value or state-action function. Similarly, Anonymous Influence can be exploited to simplify the implementation of the Programs (32) and (34). For example, consider the wildfire model (Section III) and let $O(i) = i \cup \mathcal{N}(i)$ and $|\mathcal{N}(i)| = 4$ for all trees. Without mixed-mode functions (MMFs), Program (32) requires enumerating on the order of 10^7 state combinations. By using a MMF, for the basis functions we present in Section VI, we only need to consider on the order of 10^3 state combinations. This reduction significantly simplifies the implementation of our framework which is still tractable for graphs where MDPs may have many neighbors or large state spaces. In the next section, we present several simulation experiments to validate the performance of our filter and the combined filter and controller.

VI. Simulation Experiments

For all simulation experiments, values of $\alpha = 0.2$ and $\beta = 0.9$ were used for the tree dynamics in the forest wildfire model (Section III). We use a forest size of 50×50 with 10^{1192} total states and at the initial time, all trees are healthy except for a 4×4 grid of fires in the center of the forest. Simulations terminate when there are no more trees on fire.

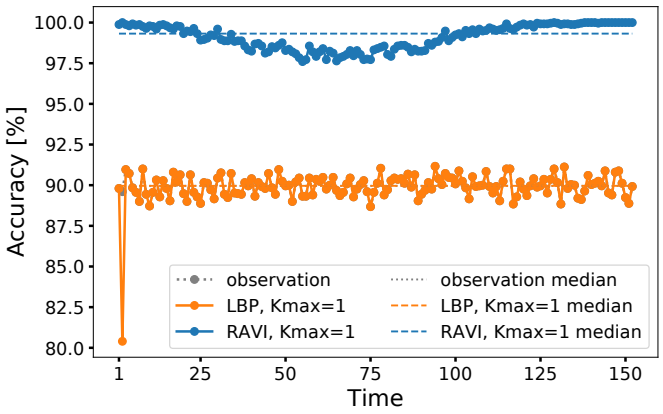


Figure 3: Example filter results for a single model simulation. The simulation accuracy for a filter is the median accuracy over the entire time series. Here, LBP is the same as taking as the measurement as the estimate, as it overlays the measurement accuracy in the plot. In contrast, RAVI is 9% better.

A. Filter Performance

We compare our filter RAVI against loopy belief propagation (LBP) which we adapt for online sequential estimation by limiting the time history to a maximum of length $H = 3$. If adding a new time step will exceed this limit, the model is reinitialized with only the latest time step and the prior belief is from the first time step of the previous model. For RAVI, the value of ϵ was chosen to be 10^{-10}. Both RAVI and LBP were terminated early if less than 1% of the posterior factors stop changing their maximum likelihood belief. We also compare against taking each measurement as the estimate for each time step, which may produce inconsistent estimates, e.g., a tree transitioning from healthy to burnt in one time step. All filters were initialized with the ground truth.

At each time step of the simulation, the belief produced by the filter is converted to a maximum likelihood estimate and compared to the true state. We compute the percentage of trees whose states are correctly estimated as the "accuracy" of the estimator at that time step. The result is a time history of accuracies for a single simulation, as shown in Fig. 3. We compute the median accuracy for the time history which we call the simulation accuracy. We run 10 total simulations and report the first quartile, the median, and the third quartile simulation accuracy. Table II shows the results for different message passing iteration limits and measurement model accuracies p_c in (6). While $p_c = 0.9$ may seem like a very accurate measurement model, there is a $0.9^{2500} \approx 10^{-115}$ probability that the true ground state is observed. For the lowest limit $K_{\max} = 1$, LBP is slow and is the same accuracy as simply taking the measurement as the estimate. While we showed in prior work that LBP can be effective when given enough iterations [4], it does not scale to the model size in this work and cannot be used online. For example, the 2018 Camp wildfire in Northern California at one point was spreading at a rate of 80 acres per minute [47]. At this rate, a wildfire burns 184 acres in 138 seconds, compared to 5.33 acres in 4 seconds. Therefore, it is critical to use a fast, accurate online filter to enable an effective response to natural disasters.

Table II: Filter results for two different measurement accuracies p_c in (6). Data are the median simulation accuracy for 10 simulations, and the subscript and superscript indicate the first and third quartiles, respectively. LBP improves with more iterations but is slow while RAVI is accurate and fast enough to be used online.

Method		Measurement Accuracy 80%	Measurement Accuracy 90%	Estimate Time (seconds)
Measurement		$80.0^{+0.1}_{-0.1}\%$	$90.0^{+0.2}_{-0.1}\%$	—
LBP	$K_{\max} = 1$	$80.0^{+0.0}_{-0.0}\%$	$90.0^{+0.0}_{-0.0}\%$	138.73
	$K_{\max} = 2$	$85.0^{+0.8}_{-1.1}\%$	$94.5^{+0.9}_{-0.8}\%$	251.43
RAVI	$K_{\max} = 1$	$98.0^{+0.4}_{-0.3}\%$	$99.4^{+0.1}_{-0.2}\%$	1.41
	$K_{\max} = 5$	$98.6^{+0.2}_{-0.2}\%$	$99.5^{+0.0}_{-0.1}\%$	4.25
	$K_{\max} = 10$	$98.6^{+0.2}_{-0.2}\%$	$99.5^{+0.0}_{-0.1}\%$	4.23

In contrast, RAVI is effective even for the low iteration limit, and improves slightly given more iterations. Coordinate ascent methods are known to find local optima and RAVI quickly finds a solution which does not significantly change with more iterations. In particular, the time to produce an estimate drops for $K_{\max} = 10$ which is likely due to slightly more accurate posterior factors at earlier time steps in each simulation.

B. Closed-loop Filter and Controller Performance

We call our control approach ACSAR (Approximate Constrained Scalable Allocation of Resources) and present simulation results to demonstrate the performance of our closed-loop filter and control approach, RAVI ACSAR.

Given the filtering results in Table II, we use the measurement and RAVI (with $K_{\max} = 5$) and $p_c = 0.9$ as filtering methods. We use ACSAR to generate both an approximate value function and an approximation state-action function, and we also compare with a method from prior work for approximate value functions. The control effectiveness parameter used was $\Delta\beta = 0.45$, the discount factor was $\gamma = 0.95$, and the control capacity was $C = 5$.

We use the following reward and basis functions in conjunction with our approximate value function approach,

$$r_i(x^t_{i\cup\mathcal{N}(i)}) = \mathbf{1}_H(x^t_i) - \mathbf{1}_F(x^t_i)e^t_i,$$
$$w_i^T h_i(x^t_{i\cup\mathcal{N}(i)}) = w_{i,0} + w_{i,1}\mathbf{1}_H(x^t_i) + w_{i,2}\mathbf{1}_F(x^t_i)e^t_i,$$

where $e^t_i = \sum_{j\in\mathcal{N}(i)} \mathbf{1}_H(x^t_j)$ is the number of healthy trees that are neighbors of tree i. Furthermore, we assume that every tree has four neighbors, $|\mathcal{N}(i)| = 4\ \forall i \in \mathcal{V}$, so that there is one equivalence class. The derived policy is then applied to the original graph model. The Program (32) requires computing an expectation of the basis functions, which is conditioned on a given configuration of the states $x^t_{i\cup\mathcal{N}(i)\cup\mathcal{N}(\mathcal{N}(i))}$. The result is the following expression after applying the dynamics

in Table I,

$$\begin{aligned}
&\mathbb{E}_p\left[w_i^T h_i(x^{t+1}_{i\cup\mathcal{N}(i)})\right] \\
&= w_{i,0} + w_{i,1}\mathbf{1}_H(x_i^t)p(x_i^{t+1}=H \mid x_i^t, x^t_{\mathcal{N}(i)}) \\
&\quad + w_{i,2}\Big(\mathbf{1}_H(x_i^t)p(x_i^{t+1}=F \mid x_i^t, x^t_{\mathcal{N}(i)}) \\
&\qquad + \mathbf{1}_F(x_i^t)p(x_i^{t+1}=F \mid x_i^t, a_i^t)\Big) \\
&\qquad \sum_{j\in\mathcal{N}(i)} \mathbf{1}_H(x_j^t)p(x_j^{t+1}=H \mid x_j^t, x^t_{\mathcal{N}(j)}).
\end{aligned} \tag{39}$$

The resulting approximate linear program is then,

$$\begin{aligned}
\min_{\substack{w_i\in\mathbb{R}^3\\ \phi_i\in\mathbb{R}}} \quad & \phi_i \\
\text{s. t.} \quad & \phi_i \geq w_{i,0} + w_{i,1}\mathbf{1}_H(x_i^t) + w_{i,2}\mathbf{1}_F(x_i^t)e_i^t \\
& \quad - \mathbf{1}_H(x_i^t) - \mathbf{1}_F(x_i^t)e_i^t \\
& \quad - \gamma\,(\text{Eq. (39)})\,, a_i^t = 0, \forall\ x^t_{i\cup\mathcal{N}(i)\cup\mathcal{N}(\mathcal{N}(i))}. \\
& \phi_i \geq -w_{i,0} - w_{i,1}\mathbf{1}_H(x_i^t) - w_{i,2}\mathbf{1}_F(x_i^t)e_i^t \\
& \quad + \mathbf{1}_H(x_i^t) + \mathbf{1}_F(x_i^t)e_i^t \\
& \quad + \gamma\,(\text{Eq. (39)})\,\forall\ x^t_{i\cup\mathcal{N}(i)\cup\mathcal{N}(\mathcal{N}(i))}, a_i^t.
\end{aligned}$$

To implement this program, we exploit Anonymous Influence to drastically reduce the number of constraints that must be specified. Computing the expectation (37) to determine the control policy then yields the following action weights,

$$\mu_i = -\gamma w_2 \mathbf{1}_F(x_i^t)\Delta\beta \sum_{j\in\mathcal{N}(i)} \mathbf{1}_H(x_j^t)(1-\alpha f_j^t).$$

Solving the program yields $w_2 = -1.43$, and so this policy treats fires in priority of the number of neighboring healthy trees and their likelihood of remaining healthy at the next time step. We compare our approach with the basis approximation proposed in [11], which can also be used with our capacity constrained formulation 35. The basis functions are,

$$w_i^T h_i^{\text{prior}}(x_i^t) = w_0\mathbf{1}_H(x_i^t) + w_1\mathbf{1}_F(x_i^t) + w_2\mathbf{1}_B(x_i^t),$$

The action weight for the policy (36) when using this basis with the same reward function as before is,

$$\mu_i = \gamma\Delta\beta\mathbf{1}_F(x_i^t)(w_3 - w_2).$$

Therefore, the resulting policy is to randomly treat trees on fire at each time step.

Table III summarizes the results for two different filtering methods in combination with the prior work basis functions and our basis functions. We present the median percent of surviving healthy trees, along with the first and third quartile, to summarize the performance of the overall framework. Without control, nearly the entire forest typically burns down. Although the measurement appears to be accurate enough for control, there are many cases where a tree is believed to be on fire but is actually healthy or burnt. As a result, control actions are wasted as treating a healthy or burnt tree has no effect. Finally, only the combination of our filter and our value function basis approximation is successful in extinguishing the wildfire. We also note that the approximation error for the prior work basis is $\phi_i = 2.29$ whereas for our approach it is $\phi_i = 1.97$.

Table III: Results for two filter methods and two choices of value function basis approximations. Data are the median remaining percent of remaining healthy trees over 100 simulations, with the subscript and superscript denoting the first and third quartile, respectively. Without control, the majority of the forest burns down. An accurate filter is required, as otherwise control effort is wasted on trees that are believed to be on fire but are actually healthy or burnt. Only our filtering method RAVI and our control approach ACSAR is successful in preserving the majority of trees in the forest.

Filter Method	Control Method	Remaining Healthy Trees
—	No Control	$1.0^{+0.0}_{-0.0}\%$
Measurement	Prior Work [11] $V_w(x^t)$	$1.3^{+0.3}_{-0.3}\%$
	ACSAR $V_w(x^t)$	$2.2^{+0.5}_{-0.3}\%$
RAVI $K_{\max} = 5$	Prior Work [11] $V_w(x^t)$	$1.9^{+1.7}_{-0.4}\%$
	ACSAR $V_w(x^t)$	$97.8^{+0.6}_{-1.0}\%$

Lastly, we also construct an approximate Q_w function using ACSAR to illustrate a complementary approach. We use the following reward and basis functions,

$$\begin{aligned}
&r_i(x_i^t, x_i^{t+1}) = \mathbf{1}_H(x_i^t) - (1-a_i^t)\mathbf{1}_F(x_i^{t+1}), \\
&w_i^T b_i(x_i^t) = w_{i,0} + w_{i,1}\mathbf{1}_H(x_i^t) + w_{i,2}\mathbf{1}_F(x_i^t), \\
&a_i^t w_i^T c_i(x^t_{i\cup\mathcal{N}(i)}) = a_i^t w_{i,3}\mathbf{1}_F(x_i^t)e_i^t.
\end{aligned}$$

After solving Program (34), the approximation error was $\phi_i = 0.84$ and over 100 simulations, the percent of remaining healthy trees was $97.8^{+0.7}_{-1.0}\%$.

VII. Conclusions

In this work, we proposed a certainty-equivalence approach to build a framework capable of addressing large-scale graph-based models with control constraints and measurement uncertainty. After splitting the problem into two parts, we derived approximately optimal filtering and control methods. Future work will focus on relaxing the structural assumptions that were required to scale our approach to models with very large state spaces. In addition, the parameters α and β in our wildfire model may be time or spatially varying, to model wind and terrain effects. It would be useful to parameterize either our linear programs or the resulting policy (or both) by these quantities, so that it is possible to react to changing model parameters online. Lastly, prior work on factored POMDP formulations may provide ideas for a scalable framework construction with improved guarantees.

APPENDIX

A. Derivations for Other Measurement Models

We now consider measurement models of the form $p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t)$. The form of (16) is then,

$$\mathbb{E}_{-i}\left[p(x^t, y^t \mid y^{1:t-1})\right] \propto \sum_{\{x_j^t \mid j\in\mathcal{V}, j\neq i\}} \Big(\prod_{\substack{j=1\\ j\neq i}}^{n} q_j(x_j^t)\Big)\Big(\prod_{i=1}^{n} p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t)\Big)\times \Big(\sum_{x^{t-1}}\prod_{i=1}^{n} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) u_i(x_i^{t-1})\Big). \tag{40}$$

The first iteration is,

$$E_i^1(x_i^t) \propto \left[\sum_{x_{\mathcal{N}(i)}^t} p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t) \prod_{j\in\mathcal{N}(i)} q_j^0(x_j^t)\right] \sum_{x_i^{t-1}} u_i(x_i^{t-1})\left[p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} u_j(x_j^{t-1})\right],$$

and the second iteration is,

$$\begin{aligned} E_i^2(x_i^t) \propto{}& \sum_{x_{\mathcal{N}(i)}^t} p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t) \prod_{j\in\mathcal{N}(i)} q_j^1(x_j^t) \\ & \sum_{x_i^{t-1}} u_i(x_i^{t-1}) p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} u_j(x_j^{t-1}) \\ & \left[\sum_{x_{\mathcal{N}(j)}^t} p_j(y_j^t \mid x_j^t, x_{\mathcal{N}(j)}^t) \prod_{l\in\mathcal{N}(j)} q_l^1(x_l^t) \right. \\ & \left. p_j(x_j^t \mid x_j^{t-1}, x_{\mathcal{N}(j)}^{t-1}, a_j^{t-1}) \prod_{l\in\mathcal{N}(j)} u_l(x_l^{t-1})\right]. \end{aligned}$$

Following the same derivation as before and using the common structure in the previous two expressions indicated by brackets, the quantity d_i^k is,

$$\begin{aligned} d_i^k(x_i^{t-1}, x_i^t) = {}& \sum_{x_{\mathcal{N}(i)}^t} p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t) \\ & \sum_{x_{\mathcal{N}(i)}^{t-1}} p_i(x_i^t \mid x_i^{t-1}, x_{\mathcal{N}(i)}^{t-1}, a_i^{t-1}) \prod_{j\in\mathcal{N}(i)} m_j^{k-1}(x_j^{t-1}, x_j^t). \end{aligned}$$

Messages are now initialized as,

$$m_i^0(x_i^{t-1}, x_i^t) = u_i(x_i^{t-1}) q_i^0(x_i^t).$$

The k^{th} estimate update (15) and message update (26) remain the same for this case. Similar to the derivation in Section IV, influence from additional HMMs is approximately added to estimates of (16), as we do not consider special graph structure. In addition, the Anonymous Influence property can be exploited if either the dynamics (1) or the measurement distribution $p_i(y_i^t \mid x_i^t, x_{\mathcal{N}(i)}^t)$ (or both) can be represented using mixed-mode functions (MMFs).

ACKNOWLEDGMENTS

The authors would like to thank Adam Caccavale, Eric Cristofalo, Preston Culbertson, and Kunal Shah for their insightful discussions and suggestions.

REFERENCES

[1] V. Raghavan, G. ver Steeg, A. Galstyan, and A. G. Tartakovsky, "Coupled hidden Markov models for user activity in social networks," in *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2013, pp. 1–6.

[2] W. Dong, A. S. Pentland, and K. A. Heller, "Graph-coupled HMMs for modeling the spread of infection," in *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 2012, pp. 227–236.

[3] R. N. Haksar and M. Schwager, "Controlling large, graph-based MDPs with global control capacity constraints: An approximate LP solution," in *57th IEEE Conference on Decision and Control (CDC)*, Dec 2018, pp. 35–42.

[4] R. N. Haksar, J. Lorenzetti, and M. Schwager, "Scalable filtering of large graph-coupled hidden Markov models," in *2019 IEEE Conference on Decision and Control (CDC)*, Dec 2019, in press.

[5] Z. Feng and E. A. Hansen, "Approximate planning for factored POMDPs," in *Proceedings of the 6th European Conference on Planning*, 2001.

[6] P. Poupart and C. Boutilier, "VDCBPI: an approximate scalable algorithm for large POMDPs," in *Advances in Neural Information Processing Systems 17*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. MIT Press, 2005, pp. 1081–1088.

[7] C. Guestrin, D. Koller, and R. Parr, "Solving factored POMDPs with linear value functions," in *Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01) workshop on Planning under Uncertainty and Incomplete Information*, 2001, pp. 67–75.

[8] T. Veiga, M. Spaan, and P. Lima, "Point-based POMDP solving with factored value function approximation," 2014.

[9] J. Pajarinen, J. Peltonen, A. Hottinen, and M. A. Uusitalo, "Efficient planning in large POMDPs through policy graph based factorized approximations," in *Machine Learning and Knowledge Discovery in Databases*, J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–16.

[10] C. Boutilier, R. Dearden, and M. Goldszmidt, "Stochastic dynamic programming with factored representations," *Artificial Intelligence*, vol. 121, no. 1-2, pp. 49–107, 2000.

[11] N. Forsell and R. Sabbadin, "Approximate linear-programming algorithms for graph-based Markov decision processes," in *Proceedings of the European Conference on Artificial Intelligence*, 2006, pp. 590–594.

[12] D. Koller and R. Parr, "Computing factored value functions for policies in structured MDPs," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 1999, pp. 1332–1339.

[13] R. Sabbadin, N. Peyrard, and N. Forsell, "A framework and a mean-field algorithm for the local control of spatial processes," *International Journal of Approximate Reasoning*, vol. 53, no. 1, pp. 66–86, 2012.

[14] Q. Cheng, Q. Liu, F. Chen, and A. T. Ihler, "Variational planning for graph-based MDPs," in *Advances in Neural Information Processing Systems*, 2013, pp. 2976–2984.

[15] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman, "Efficient solution algorithms for factored MDPs," *Journal of Artificial Intelligence Research*, vol. 19, pp. 399–468, 2003.

[16] F. Chen, Q. Cheng, J. Dong, Z. Yu, G. Wang, and W. Xu, "Efficient approximate linear programming for factored MDPs," *International Journal of Approximate Reasoning*, vol. 63, pp. 101–121, 2015.

[17] P. Robbel, F. A. Oliehoek, and M. J. Kochenderfer, "Exploiting anonymity in approximate linear programming: scaling to large multiagent MDPs," in *AAAI Conference on Artificial Intelligence*, 2016, pp. 2537–2573.

[18] E. Altman, *Constrained Markov decision processes*. CRC Press, 1999.

[19] D. Adelman and A. J. Mersereau, "Relaxations of weakly coupled stochastic dynamic programs," *Operations Research*, vol. 56, no. 3, pp. 712–727, 2008.

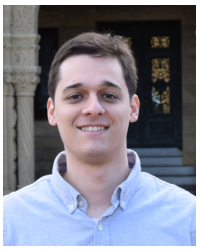
[20] F. Ye, H. Zhu, and E. Zhou, "Weakly coupled dynamic program: Information and lagrangian relaxations," *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 698–713, 2018.

[21] N. Meuleau, M. Hauskrecht, K.-E. Kim, L. Peshkin, L. P. Kaelbling, T. Dean, and C. Boutilier, "Solving very large weakly coupled Markov decision processes," in *AAAI Conference on Artificial Intelligence*, 1998, pp. 165–172.

[22] H.-J. Schütz and R. Kolisch, "Approximate dynamic programming for capacity allocation in the service industry," *European Journal of Operational Research*, vol. 218, no. 1, pp. 239 – 250, 2012.

[23] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, 2012.

[24] D. F. Ciocan and V. Farias, "Model predictive control for dynamic resource allocation," *Mathematics of Operations Research*, vol. 37, no. 3, pp. 501–525, 2012.
[25] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, July 2000.
[26] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings F - Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
[27] N. Vaswani, A. Yezzi, Y. Rathi, and A. Tannenbaum, "Particle filters for infinite (or large) dimensional state spaces- part 1," in *IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 3, May 2006, pp. III–III.
[28] T. Seo, T. T. Tchrakian, S. Zhuk, and A. M. Bayen, "Filter comparison for estimation on discretized PDEs modeling traffic: Ensemble Kalman filter and minimax filter," in *55th IEEE Conference on Decision and Control (CDC)*, Dec 2016, pp. 3979–3984.
[29] A. Beskos, D. Crisan, A. Jasra, K. Kamatani, and Y. Zhou, "A stable particle filter for a class of high-dimensional state-space models," *Advances in Applied Probability*, vol. 49, no. 1, p. 24–48, 2017.
[30] A. Jasra, S. S. Singh, J. S. Martin, and E. McCoy, "Filtering via approximate Bayesian computation," *Statistics and Computing*, vol. 22, no. 6, pp. 1223–1237, Nov 2012.
[31] K. Fan, C. Li, and K. Heller, "A unifying variational inference framework for hierarchical graph-coupled HMM with an application to influenza infection," in *AAAI Conference on Artificial Intelligence*, 2016, pp. 3828–3834.
[32] D. M. Blei, M. I. Jordan, and J. W. Paisley, "Variational Bayesian inference with stochastic search," in *Proceedings of the 29th International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research. PMLR, 2012, pp. 1367–1374.
[33] M. D. Hoffman and D. M. Blei, "Structured stochastic variational inference," in *Artificial Intelligence and Statistics*, 2015.
[34] A. Saeedi, T. D. Kulkarni, V. K. Mansinghka, and S. J. Gershman, "Variational particle approximations," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2328–2356, Jan. 2017.
[35] M. Yin and M. Zhou, "Semi-implicit variational inference," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018, pp. 5660–5669.
[36] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.
[37] V. Smidl and A. Quinn, "Variational Bayesian filtering," *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 5020–5030, 2008.
[38] J. Winn and C. M. Bishop, "Variational message passing," *Journal of Machine Learning Research*, vol. 6, pp. 661–694, Dec. 2005.
[39] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999, pp. 467–475.
[40] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Generalized belief propagation," in *Advances in Neural Information Processing Systems 13*. MIT Press, 2001, pp. 689–695.
[41] D. Blatner, *Spectrums: Our Mind-boggling Universe from Infinitesimal to Infinity*. A&C Black, 2013.
[42] A. H. Jazwinski, *Stochastic processes and filtering theory*. Courier Corporation, 2007.
[43] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.
[44] J. Hensman, M. Rattray, and N. D. Lawrence, "Fast variational inference in the conjugate exponential family," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2888–2896.
[45] L. K. Saul and M. I. Jordan, "Exploiting tractable substructures in intractable networks," in *Advances in Neural Information Processing Systems 8*. MIT Press, 1996, pp. 486–492.
[46] R. Williams and L. C. Baird, "Tight performance bounds on greedy policies based on imperfect value functions," Tech. Rep., 1993.
[47] M. Simon. (2018) The terrifying science behind California's massive Camp fire. [Online]. Available: https://www.wired.com/story/the-terrifying-science-behind-californias-massive-camp-fire/



Ravi N. Haksar Ravi N. Haksar is a Ph.D. candidate with the Department of Mechanical Engineering at Stanford University in the Multi-Robot Systems Lab. He obtained his BS degree from the Georgia Institute of Technology in 2014 and his MS degree from Stanford University in 2017. His research interests include control theory, probabilistic frameworks, decentralized optimization, and cooperative multi-robot teams.

Mac Schwager Mac Schwager is an assistant professor with the Aeronautics and Astronautics Department at Stanford University. He obtained his BS degree in 2000 from Stanford University, his MS degree from MIT in 2005, and his PhD degree from MIT in 2009. He was a postdoctoral researcher working jointly in the GRASP lab at the University of Pennsylvania and CSAIL at MIT from 2010 to 2012, and was an assistant professor at Boston University from 2012 to 2015. He received the NSF CAREER award in 2014, the DARPA YFA in 2018, and a Google faculty research award in 2018. His research interests are in distributed algorithms for control, perception, and learning in groups of robots and animals.