Controlling Large, Graph-based MDPs with Global Control Capacity Constraints: An Approximate LP Solution

Ravi N. Haksar[1] and Mac Schwager[2]

***Abstract*— We consider controlling graph-based MDPs (GMDPs) with two special properties: (i) Anonymous Influence and (ii) Symmetry. Large-scale spatial processes such as wildfires, disease epidemics, opinion dynamics, and robot swarms are well modeled by GMDPs with these properties. We derive two efficient and scalable algorithms for computing approximately optimal control policies for large GMDPs with Anonymous Influence and Symmetry and derive sub-optimality bounds for these policies. Unlike prior work, our algorithms explicitly enforce a global control capacity constraint. Our methods scale linearly in the number of equivalence classes in the GMDP rather than the total number of MDPs in the graph. We demonstrate our methods in simulations of controlling a wildfire with a global fire retardant constraint and controlling an Ebola outbreak with a global medicine constraint. Our Ebola model is derived from data from the 2014 West Africa outbreak.**

I. INTRODUCTION

We consider controlling a class of discrete time, discrete space, stochastic systems that are described by a graph model. Each node in the graph represents a Markov Decision Process (MDP) whose transition distribution is influenced by the states of its one-hop neighbors. Such systems are known as Graph-based MDPs (GMDPs) and are well-suited to modeling large-scale spatial dynamic processes. Examples include wildfires (each tree is an MDP influenced by neighboring trees), disease epidemics (each community is an MDP influenced by neighboring communities), opinion dynamics (each person's opinion is an MDP influenced by other people in a social network), and more [1], [2]. In this paper we propose two algorithms for generating approximately optimal control policies for GMDPs. These algorithms are most appropriate for GMDPs with two special properties commonly found in large-scale spatial processes: (i) "Anonymous Influence" and (ii) "Symmetry." Simply stated, a GMPD has Anonymous Influence if each MDP in the graph depends on the number of its neighbors in a particular state and not the identities of these neighbors. Similarly, a GMDP has Symmetry if each MDP in the graph belongs to one of a small number of "equivalence classes." Equivalence classes are constructed to contain the properties that uniquely define a value approximation of a graph node.

*This research was supported in part by NSF grant IIS-1646921. We are grateful for this support.

[1]R. N. Haksar is with the Department of Mechanical Engineering, Stanford University, Stanford, CA 94305 `rhaksar@stanford.edu`

[2]M. Schwager is with the Department of Aeronautics & Astronautics, Stanford University, Stanford, CA 94305 `schwager@stanford.edu`

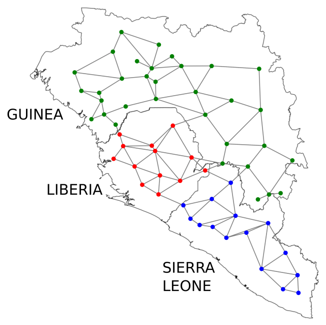

Fig. 1: Graph-based MDP encompassing three countries affected by the 2014 West Africa Ebola outbreak. Nodes in the graph are MDPs representing the infection state of the local community while edges represent their mutual infection influence. We demonstrate our control policy with a GMDP model derived from data on the 2014 Ebola outbreak.

We specifically consider controlling GMDPs with a global control capacity constraint as capacity constraints are crucial in modeling the control of spatial processes, such as wildfires and disease epidemics. The optimal unconstrained policy for fighting a wildfire is to apply fire retardant to every tree at every time. Similarly, the optimal unconstrained policy for a disease epidemic is to treat every person with medicine at every time. Hence, these problems are only rendered meaningful if a constraint is applied to the total control effort available, as we consider in this work.

Our algorithms are formulated as linear programs (LPs) based on the LP framework for approximate dynamic programming. We provide one algorithm for value functions and one algorithm for Q-functions. Both algorithms generate approximations by solving for the weights of a factored basis function representation and we provide sub-optimality bounds for each algorithm. We also provide an approximate method for satisfying a global capacity constraint. Lastly, we present results of a simulation study of controlling a wildfire with limited fire retardant and controlling an Ebola outbreak with limited medicine. The Ebola model is derived from data on the 2014 West Africa Ebola outbreak [3].

A. Related Work

Traditional MDP descriptions are inappropriate for structured model descriptions as the state and action spaces of the aggregate MDP description typically grow exponentially in the number of constituent MDPs. The Factored MDP (FMDP) [4] and Graph-based MDP (GMDP) [5] frameworks have been formulated to compactly represent structured

MDPs. Nevertheless, the compact representation does not generally translate to tractable exact solution methods that are analogous to traditional MDP methods [6]. As a result, approximate solution techniques have been proposed.

We leverage approximate LPs (ALPs) in this work. ALP methods circumvent the explicit enumeration of the state space by using a basis function approximation of the value function or the $\mathcal{Q}$-function. Guestrin et al. [7] used variable elimination (VE) to efficiently generate constraints which results in an exponential dependence on the tree-width instead of the full state space. The method proposed by Forsell et al. [5] uses a specific basis function form and requires solving a linear program for each constituent MDP instead of for the aggregate MDP linear program. Chen et al. [1] proposed a VE approach that solves linear programs for the constituent MDPs and then enforces consistency constraints. Robbel et al. [2] developed a VE approach for systems where only the number of variables in a state is important, not the identity of the variables. This property, called Anonymous Influence, is also exploited in this paper.

With the exception of [5], prior work considers all MDPs in the graph to compute a policy. In contrast, we consider GMDPs where each MDP belongs to one of a small number of equivalence classes, a property which we call Symmetry. Our solution technique only requires solving an LP once for each class with the resulting policy then applied to each MDP belonging to that class, resulting in a large computational benefit when there are a small number of classes. We note that [5] is the most relevant prior work but the authors do not consider control constraints or $\mathcal{Q}$-functions.

The assignment of limited resources to a set of coupled or decoupled MDPs has also been considered in literature. Constrained MDP formulations [8] allow explicit control constraints but require traditional MDP descriptions. Approximate methods are proposed due to process stochasticity and large state and action spaces, including Lagrangian relaxation [9], [10], approximate dynamic programming [11], [12], Monte Carlo tree search [13], and receding horizon optimization [14]. 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. Our approximation is similar in spirit to [11].

The rest of this work is organized as follows. In Section II we introduce graph-based models, describe the properties of Symmetry and Anonymous Influence, and define GMDPs. We also summarize the main results of prior ALP approaches. In Section III we derive our novel ALP methods without capacity constraints. We impose a capacity constraint and give the resulting solution in Section IV. Numerical examples validating our methods are provided in Section V.

II. Background

A. Graph-based Models

Graph-based models contain a (directed or undirected) graph $G = (V, E)$ with a vertex set $V = \{1, \dots, m\}$ containing m nodes and an edge set $E \subseteq V \times V$. Each node i corresponds to a Markov process and has an associated discrete state space $\mathcal{X}_i$ and discrete action space $\mathcal{A}_i$. We consider discrete-time problems and use the notation $x^t_{O(i)}$ to denote the states of the nodes in the set $O(i) \subseteq V$ at time t, $x^t_{O(i)} = \{x^t_j \mid j \in O(i)\}$. Similarly, $a^t_{O(i)}$ describes the action for the set $O(i)$ at time t. An aggregate state and action space is defined by the Cartesian products $\mathcal{X} = \prod_{i=1}^m \mathcal{X}_i$ and $\mathcal{A} = \prod_{i=1}^m \mathcal{A}_i$. We omit the subscript for these quantities, $x^t \in \mathcal{X}$ and $a^t \in \mathcal{A}$. Subsets of the aggregate spaces are denoted by subscripts, e.g. $\mathcal{X}_{O(i)} = \prod_{j \in O(i)} \mathcal{X}_j \subseteq \mathcal{X}$.

The state evolution of node i generally requires information from its neighbor set $N(i)$ which is defined as $N(i) = \{j \in V \mid (j,i) \in E\}$ for a (directed or undirected) graph $G = (V, E)$. We define the *parent function* to describe the dependency of a node's transition distribution on other nodes in the graph.

Definition 1 (Parent Function)**.** The *parent function* $\Gamma(\mathcal{S}) : \mathcal{S} \subseteq V \mapsto \mathcal{T} \subseteq V$ is defined relative to a graph $G = (V, E)$ as $\Gamma(\mathcal{S}) = \bigcup_{j \in \mathcal{S}} j \, \cup \, N(j)$ where $N(j)$ is the neighbor set of node j. For a singleton set, $\Gamma(j) = j \, \cup N(j)$.

Using the parent function, the node dynamics are,

$$p_i(x_i^{t+1} \mid x^t_{\Gamma(i)}, a^t_{\Gamma(i)}). \tag{1}$$

The transition distribution for the aggregate state is defined by the joint probability distribution,

$$p(x^{t+1} \mid x^t, a^t) = \prod_{i=1}^m p_i(x_i^{t+1} \mid x^t_{\Gamma(i)}, a^t_{\Gamma(i)}). \tag{2}$$

The representations of (1) and (2) are reduced by adding structure to the graph model.

B. Symmetry in Graph-based Models

If a graph-based model consists of a comparatively small number of "equivalence classes" of MDPs, we say it has "Symmetry." Two MDPs i and j are in the same equivalence class k, denoted by $i, j \in \mathcal{C}_k$, if both MDPs use the same reward, transition, and basis approximation functions.

$$\text{class } k : \begin{cases} \text{describes nodes } i \in \mathcal{C}_k \subseteq V \text{ and } |\mathcal{C}_k| = n_k \\ \text{transition distribution } p_i = p_j \ \forall i,j \in \mathcal{C}_k \\ \text{reward functions } r_i = r_j \ \forall i,j \in \mathcal{C}_k \\ \text{basis functions } w_i^T h_i = w_j^T h_j \ \forall i,j \in \mathcal{C}_k \end{cases}$$

The m MDPs in a graph-based MDP are partitioned into $s \leq m$ unique classes and $\sum_{k=1}^s n_k = m$. Many domains contain a graph topology with many nodes but few classes. Methods that require enumerating the state space scale exponentially with the number of nodes and quickly become intractable. In contrast, our methods exploit Symmetry to instead scale with the number of equivalence classes.

C. Anonymous Influence in Graph-based Models

We consider graph-models which describe the node dynamics as dependent on the number of neighbors in a particular state rather than the identity of the neighbors. Robbel et al. [2] describe this property as "Anonymous Influence" and leverage it to improve the efficiency of variable elimination. Similarly, we leverage it to improve the efficiency of implementing our approximate linear programming methods. We summarize the relevant definitions here and assume binary variables in the following discussion.

A *count aggregator* (CA) describes the number of "enabled" variables in a set of binary variables $\mathcal{Z}$, $\#\{\mathcal{Z}\}(z) = \sum_{i=1}^{|\mathcal{Z}|} z_i$, and maps an instantiation of the variables $z \in \mathcal{Z}$ to $\{0, \ldots, |\mathcal{Z}|\}$. A *count aggregator function* (CAF) uses a CA to map a set of binary numbers to the set of real numbers, $f : \mathcal{Z} \mapsto \mathbb{R}$. CAFs have a compact representation as they are represented by $|\mathcal{Z}| + 1$ numbers compared to $2^{|\mathcal{Z}|}$ for a function that operates on instantiations of the binary variables. We also adopt the notation $f(\#(z))$ to emphasize that f uses a CA. A *Mixed-mode function* (MMF) $f(w, \#(z))$ is a function that uses a CA and a discrete variable w not part of a CA as arguments and maps to the real numbers, $f : \mathcal{Z} \times \mathcal{W} \mapsto \mathbb{R}$. A MMF also has a compact representation, requiring at most $|\mathcal{W}|\left(|\mathcal{Z}| + 1\right)$ values.

The node dynamics (1) are alternatively described by an MMF, $p(x_i^{t+1} \mid x_i^t, \#(z_i^t), a_{\Gamma(i)}^t)$ where z_i^t is a CA representing the influence from the neighbor set, for a graph-based MDP with Anonymous Influence. In the next section, we describe a wildfire model inspired by [15] to elucidate the previous discussion on Symmetry and Anonymous Influence.

D. Example: Wildfire Model

We model the forest as a set of trees represented by nodes on a two dimensional plain square lattice $\mathbb{Z}^2$. The tree state x_i^t is one of three values, $\mathcal{X}_i = \{H, F, B\} = \{\text{healthy, on fire, burnt}\}$. The position of a tree is $q_i \in \mathbb{Z}^2$ and the inter-tree distance on the lattice is assumed to be one unit. An undirected graph is used to represent the trees and influence between trees. There are m total trees and the vertex set is $V = \{1, \ldots, m\}$. Edges exist between nodes if the Manhattan distance between trees is one and the neighbor set for a tree is $N(i) = \{j \mid \|q_j - q_i\|_1 = 1 \ \forall j \in V\}$.

TABLE I: Single tree dynamics for wildfire model

x_i^t \ x_i^{t+1}	H	F	B
H	$1 - \alpha u_i^t$	αu_i^t	0
F	0	$\beta - \Delta\beta a_i^t$	$1 - \beta + \Delta\beta a_i^t$
B	0	0	1

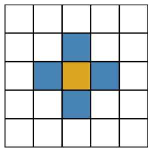

Fig. 2: The lattice for the wildfire model is visualized as a grid of cells. For the gold cell, the neighbor set contains the blue cells and the parent function return the gold and blue cells. White cells are part of the forest but are neither in the neighbor set nor are a parent cell of the gold cell.

Table I summarizes the transition distribution where $u_i^t = \sum_{j \in N(i)} \mathbf{1}_F(x_j^t)$ is the number of neighboring trees on fire. We use $\mathbf{1}_A(x)$ to represent the indicator function that equals one when $x = A$ and zero otherwise. Figure 2 shows an example of the neighbor set and parent function.

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.

One equivalence class describes all trees if the same basis approximation is used and each tree has the same number of neighbors. Without MMFs, the domain of each node transition distribution p_i contains $|\mathcal{X}_{\Gamma(i)}||\mathcal{A}_i| = 6 \cdot 3^{|N(i)|}$ elements and is exponential in the number of neighbors $|N(i)|$. 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 function representation. The model contains ternary variables and $\mathbf{1}_F(x_j^t)$ describes trees on fire as enabled variables. With a MMF, the domain of p_i is reduced to $(|N(i)|+1)|\mathcal{X}_i||\mathcal{A}_i| = 6(|N(i)| + 1)$ values and is now linear in $|N(i)|$.

The aggregate state space of the wildfire model has 3^m forest 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 [16]. For comparison, there are approximately 26,000 trees in New York's Central Park and is a relatively small forest. Thus using MMFs to represent tree dynamics and exploiting Symmetry is essential to implementing our ALP methods (Section III).

E. Graph-based Markov Decision Process (GMDP)

A graph-based MDP (GMDP) is a graph-based model with structured transition and reward functions. The transition distribution is represented by (2) and the reward function is additively composed,

$$R(x^t, a^t, x^{t+1}) = \sum_{i=1}^{m} r_i(x_{\Gamma(O(i))}^t, a_{\Gamma(O(i))}^t, x_{O(i)}^{t+1}), \qquad (3)$$

where r_i is a per-node immediate reward function that depends on a subset of nodes, $O(i) \subseteq V \ \forall i$. Prior work frequently assumes that the per-node rewards are deterministic and only include the action a_i^t. In contrast, our methods allow for a more general formulation of the per-node reward function. The tuple $\langle \{\mathcal{X}_i \ \forall i\}, \{\mathcal{A}_i \ \forall i\}, \{p_i \ \forall i\}, \{r_i \ \forall i\}, G \rangle$ fully specifies a GMDP. In the next section, we discuss approximate linear programming methods for solving GMDPs.

F. Approximate Linear Programming (ALP) Methods

We consider the infinite horizon, discounted cost problem where the objective is to maximize the value function $v(x^t)$ over the policies $\pi(x^t)$. The optimal value function can be computed by linear programming using $|\mathcal{X}|$ variables where variable v_{x^t} refers to the value of state x^t, $v_{x^t} = v(x^t)$. The exact linear program is,

$$\begin{aligned}
\max_{v_{x^t} \forall x^t \in \mathcal{X}} \quad & \sum_{x^t \in \mathcal{X}} \alpha(x^t) v_{x^t} \\
\text{s. t.} \quad & v_{x^t} \geq \mathbb{E}_p \left[R(x^t, a^t, x^{t+1}) + \gamma v_{x^{t+1}} \right] \\
& \forall x^t \in \mathcal{X}, a^t \in \mathcal{A},
\end{aligned}$$

for given positive state relevance weights $\alpha(x^t)$ and discount factor γ. The notation $\mathbb{E}_p$ indicates the expectation is taken with respect to the joint probability (2). The program contains $|\mathcal{X}||\mathcal{A}|$ constraints which is prohibitive in GMDPs as there is one state for each assignment to all nodes in the graph, resulting in an exponentially large state space.

Therefore, one approach is to approximate the value function as a linear combination of n_w (possibly non-linear) basis functions, $v_w(x^t) = \sum_{i=1}^{n_w} w_i^T h_i(x^t_{O(i)})$ where $w_i \in \mathbb{R}^{k_i}$ and $h_i : \mathcal{X}_{O(i)} \subseteq \mathcal{X} \mapsto \mathbb{R}^{k_i}$ [17]. Each basis function h_i typically relies on relatively few nodes in the graph, $O(i) \subseteq V$ and $|O(i)| \ll m$. The purpose of these assumptions is to leverage the additive structure of the reward function and to greatly reduce the complexity of solving an approximate linear program (ALP). We occasionally omit the arguments of functions for clarity in the following discussion. The ALP for the assumed value function form is,

$$\begin{aligned}
\max_{w_i \in \mathbb{R}^{k_i} \forall i} \quad & \sum_{x^t \in \mathcal{X}} \alpha(x^t) \sum_{i=1}^{n_w} w_i^T h_i(x^t_{O(i)}) \\
\text{s. t.} \quad & \sum_{i=1}^{n_w} w_i^T h_i(x^t_{O(i)}) \geq \mathbb{E}_p \Big[R + \gamma \sum_{i=1}^{n_w} w_i^T h_i(x^{t+1}_{O(i)}) \Big] \\
& \forall x^t \in \mathcal{X}, a^t \in \mathcal{A},
\end{aligned}$$

where the program now uses $\sum_{i=1}^{n_w} k_i$ variables but still requires $|\mathcal{X}||\mathcal{A}|$ constraints. Guestrin et al. [17] replaced the linear constraints with an equivalent non-linear constraint and used variable elimination to construct the constraints and solve the ALP. Other work [1], [2], [18] improved the constraint set implementation for different problem structures.

For this ALP approach it is difficult to develop an actionable bound for the performance for a set of basis functions. de Farias and Van Roy [19] showed that the choice of state relevance weights impacts the approximation performance and developed a bound to quantify their influence. However, the bound relies on providing a Lyapunov-like function and prior work rarely uses a bound to estimate the performance. A complementary ALP approach for approximate value functions $v_w(x^t)$ is derived using a bound from [20].

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

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

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

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

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 difference relative to the optimal value function. Minimizing $\phi = \max_{x^t \in \mathcal{X}} |v_w(x^t) - (\mathcal{B}v_w)(x^t)|$ leads to the non-linear program,

$$\begin{aligned}
\min_{w_i \in \mathbb{R}^{k_i} \forall i, \phi \in \mathbb{R}} \quad & \phi \\
\text{s. t.} \quad & \phi \geq v_w(x^t) - (\mathcal{B}v_w)(x^t), \\
& \phi \geq (\mathcal{B}v_w)(x^t) - v_w(x^t), \forall x^t \in \mathcal{X},
\end{aligned} \tag{4}$$

where the nonlinearity is the maximization in the Bellman operator. Forsell et al. [5] derived a tractable approach using the following assumptions:

1) Let $v_w(x^t) = \sum_{i=1}^m \sum_{j=1}^{|\mathcal{X}_i|} w_{ij} \mathbf{1}_j(x_i^t)$, where each node has an indicator function for each state value. There are $\sum_{i=1}^m |\mathcal{X}_i|$ total basis functions.
2) The node dynamics are $p_i(x_i^{t+1} \mid x^t_{N(i)}, a_i^t)$.
3) The reward is $R(x^t, a^t) = \sum_{i=1}^m r_i(x^t_{N(i)}, a_i^t)$.

Using these assumptions, the Bellman operator is $(\mathcal{B}v_w)(x^t) = \sum_{i=1}^m \max_{a_i^t \in \mathcal{A}_i} g_i(x^t_{N(i)}, a_i^t)$ where,

$$g_i(x^t_{N(i)}, a_i^t) = r_i(x^t_{N(i)}, a_i^t) + \gamma \mathbb{E}_{p_i} \Big[\sum_{j=1}^{|\mathcal{X}_i|} w_{ij} \mathbf{1}_j(x_i^{t+1}) \Big],$$

is the expected immediate reward and future value of each constituent MDP in the graph. The expectation is now with respect to the MDP dynamics (1). The difference between the value function and Bellman operator is,

$$\begin{aligned}
& v_w(x^t) - (\mathcal{B}v_w)(x^t) \\
&= \sum_{i=1}^m \sum_{j=1}^{|\mathcal{X}_i|} w_{ij} \mathbf{1}_j(x_i^t) - \max_{a_i^t \in \mathcal{A}} g_i(x^t_{N(i)}, a_i^t) = \sum_{i=1}^m d_i(x^t_{\Gamma(i)}).
\end{aligned}$$

Two approximations are then made to the constraints in Program (4). The error ϕ is decomposed into $\phi = \sum_{i=1}^m \phi_i$,

$$\begin{aligned}
\min_{\substack{w_i \in \mathbb{R}^{|\mathcal{X}_i|} \forall i \\ \phi \in \mathbb{R}}} \quad & \phi = \sum_{i=1}^m \phi_i & \Rightarrow \quad \min_{\substack{w_i \in \mathbb{R}^{|\mathcal{X}_i|} \\ \phi_i \in \mathbb{R}}} \quad & \phi_i \\
\text{s. t.} \quad & \phi \geq \sum_{i=1}^m d_i, & \text{s. t.} \quad & \phi_i \geq d_i, \\
& \phi \geq -\sum_{i=1}^m d_i & & \phi_i \geq -d_i, \forall i
\end{aligned} \tag{5}$$

in order to impose the constraints $\phi_i \geq |d_i(x^t_{\Gamma(i)})|$ instead of $\phi \geq |\sum_{i=1}^m d_i(x^t_{\Gamma(i)})|$. This over-approximates ϕ by imposing structure on the error contribution of each node but reduces the coupled non-linear program into m non-linear programs. The maximum operator in $d_i(x^t_{\Gamma(i)})$ is then replaced by adding a linear constraint for each action since,

$$\phi_i \geq -g_i(x^t_{N(i)}, a^t_i)\ \forall a^t_i \in \mathcal{A}_i \geq -\max_{a^t_i \in \mathcal{A}_i} g_i(x^t_{N(i)}, a^t_i),$$

$$\phi_i \geq \max_{a^t_i \in \mathcal{A}_i} g_i(x^t_{N(i)}, a^t_i) \Leftrightarrow \phi_i \geq g_i(x^t_{N(i)}, a^t_i)\ \forall a^t_i \in \mathcal{A}_i.$$

The result is one linear program per node in the graph,

$$\begin{aligned}
\min_{\substack{w_i \in \mathbb{R}^{|\mathcal{X}_i|} \\ \phi_i \in \mathbb{R}}} \quad & \phi_i \\
\text{s. t.} \quad & \phi_i \geq \sum_{j=1}^{|\mathcal{X}_i|} w_{ij} \mathbf{1}_j(x^t_i) - g_i(x^t_{N(i)}, a^t_i), \\
& \phi_i \geq \sum_{j=1}^{|\mathcal{X}_i|} w_{ij} \mathbf{1}_j(x^t_i) + g_i(x^t_{N(i)}, a^t_i) \\
& \forall x^t_{\Gamma(i)} \in \mathcal{X}_{\Gamma(i)}, a^t_i \in \mathcal{A}_i.
\end{aligned}$$

Each program uses $|\mathcal{X}_i| + 1$ variables and $2|\mathcal{A}_i| \prod_{j \in \Gamma(i)} |\mathcal{X}_j|$ constraints. The quantity $\phi = \sum_{i=1}^m \phi_i$ is a sub-optimality estimate of v_w compared to the optimal value function $v^\star$.

We note that the properties of an equivalence fully determine the linear program solution. In other words, the associated program for each node in a class will yield the same solution w_i and ϕ_i. Therefore, for a GMDP with a single equivalence class, only one program must be solved and the solution is reused for all nodes in the graph. In addition, if the number of nodes in the class changes, the per-node solution does not need to be recomputed which is a desirable property as many structured domains contain many nodes but few classes. We also note that enumerating $2|\mathcal{A}_i| \prod_{j \in \Gamma(i)} |\mathcal{X}_j|$ values to specify constraints may be computationally prohibitive depending on the graph properties. In the next section, we expand the previous derivation and exploit Anonymous Influence and Symmetry to create a more general framework for GMDPs.

III. ALP Methods for GMDPs with Anonymous Influence and Symmetry

A. Approximate Value Function Method

We derive a novel ALP approach for constrained value functions based on Proposition 1. The value function is approximated as $v_w(x^t) = \sum_{i=1}^m w_i^T h_i(x^t_{O(i)})$ with $w_i \in \mathbb{R}^{k_i}$ and $h_i : \mathcal{X}_{O(i)} \subseteq \mathcal{X} \mapsto \mathbb{R}^{k_i}$. We assume binary actions, $a^t_i \in \{0, 1\}$, and a capacity constraint is imposed, $\sum_{i=1}^m a^t_i \leq C$, where C is a non-negative integer. The constrained Bellman operator is,

$$\begin{aligned}
&(\mathcal{B}v_w)(x^t) = \\
&\max_{a^t \in \mathcal{A}} \mathbb{E}_p \left[\sum_{i=1}^m r_i(x^t_{\Gamma(O(i))}, a^t_{\Gamma(O(i))}, x^{t+1}_{O(i)}) + \gamma w_i^T h_i(x^{t+1}_{O(i)}) \right] \\
&\text{s. t.} \ \sum_{i=1}^m a^t_i \leq C, \qquad a^t_i \in \{0, 1\}.
\end{aligned}$$

Computing operations over the full state space and the feasible action set is intractable so upper and lower bounds for the Bellman operator are used instead, $(\overline{\mathcal{B}v_w})(x^t) \geq (\mathcal{B}v_w)(x^t) \geq (\underline{\mathcal{B}v_w})(x^t)$. With these bounds, the following constraints are imposed,

$$\begin{aligned}
&\phi \geq v_w(x^t) - (\underline{\mathcal{B}v_w})(x^t) \geq v_w(x^t) - (\mathcal{B}v_w)(x^t), \\
&\phi \geq (\overline{\mathcal{B}v_w})(x^t) - v_w(x^t) \geq (\mathcal{B}v_w)(x^t) - v_w(x^t), \forall x^t \in \mathcal{X},
\end{aligned} \tag{6}$$

and the original non-linear program constraints are still satisfied. The arguments to functions are omitted at times in the following discussion for clarity. A lower bound is any action that satisfies the constraint. We use $a^t_i = 0\ \forall i$ and denote this action $\overline{a}^t$,

$$(\underline{\mathcal{B}v_w}) = \sum_{i=1}^m \mathbb{E}_{p_i} \left[r_i(x^t_{\Gamma(O(i))}, \overline{a}^t_{\Gamma(O(i))}, x^{t+1}_{O(i)}) + \gamma w_i^T h_i \right].$$

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,

$$(\overline{\mathcal{B}v_w}) = \sum_{i=1}^m \max_{a^t_{\Gamma(O(i))} \in \mathcal{A}_{\Gamma(O(i))}} \mathbb{E}_{p_i} \left[r_i + \gamma w_i^T h_i \right].$$

Removing the constraint over-approximates the value of each node but is critical in dividing the original intractable non-linear program into m tractable programs. The expected immediate reward and future value for every node is,

$$g_i(x^t_{\Gamma(O(i))}, a^t_{\Gamma(O(i))}) = \mathbb{E}_{p_i} \left[r_i + \gamma w_i^T h_i \right].$$

The constraints in (6) simplify to,

$$\begin{aligned}
&\phi \geq \sum_{i=1}^m -w_i^T h_i + \max_{a^t_{\Gamma(O(i))} \in \mathcal{A}_{\Gamma(O(i))}} g_i(x^t_{\Gamma(O(i))}, a^t_{\Gamma(O(i))}), \\
&\phi \geq \sum_{i=1}^m w_i^T h_i - g_i(x^t_{\Gamma(O(i))}, \overline{a}^t_{\Gamma(O(i))}), \forall x^t \in \mathcal{X}.
\end{aligned}$$

We now utilize the same approximations as [5]. The error ϕ is decomposed (5) and the maximization over actions is replaced by adding one linear constraint per action. The resulting per-node linear program is,

$$\begin{aligned}
\min_{\substack{w_i \in \mathbb{R}^{k_i} \\ \phi_i \in \mathbb{R}}} \quad & \phi_i \\
\text{s. t.} \quad & \phi_i \geq w_i^T h_i(x^t_{O(i)}) - g_i(x^t_{\Gamma(O(i))}, \overline{a}^t_{\Gamma(O(i))}), \\
& \quad \forall x^t_{\Gamma(O(i))} \in \mathcal{X}_{\Gamma(O(i))}. \\
& \phi_i \geq -w_i^T h_i(x^t_{O(i)}) + g_i(x^t_{\Gamma(O(i))}, a^t_{\Gamma(O(i))}), \\
& \quad \forall x^t_{\Gamma(O(i))} \in \mathcal{X}_{\Gamma(O(i))}, a^t_{\Gamma(O(i))} \in \mathcal{A}_{\Gamma(O(i))}.
\end{aligned} \tag{7}$$

Each linear program requires $k_i + 1$ variables and $|\mathcal{X}_{\Gamma(O(i))}| \left(|\mathcal{A}_{\Gamma(O(i))}| + 1 \right)$ constraints. The approximate value function is determined after finding the weights w_i. For arbitrary discrete node action spaces, the lower bound requires any action that satisfies the control constraint. The following theorem provides the the relationship between our ALP approach and equivalence classes.

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

Proof. The constraints in Program (7) are uniquely defined by the reward function r_i, transition distribution p_i (used by the expectation in g_i), and basis approximation $w_i^T h_i$, i.e. the properties of an equivalence class. Therefore, for m equivalence classes, the solution to each program w_i, ϕ_i is unique as no two nodes share the same class. When there are $s < m$ classes, there is at most s unique solutions for all m linear programs. In this case, only one linear program per equivalence class must be solved as the solution for the per-node program (7) is identical for all nodes within a class. $\square$

Similar to [5], specifying the constraints in Program (7) requires enumerating $\mathcal{X}_{\Gamma(O(i))}$ and $\mathcal{A}_{\Gamma(O(i))}$ and may be computationally prohibitive. Therefore, we now leverage the Anonymous Influence property to circumvent this potential issue. Consider the wildfire model (Section II-D) and let $O(i) = \Gamma(i)$ and $|N(i)| = 4$ for all nodes. Without MMFs, Program (7) requires $|\mathcal{X}_{\Gamma(O(i))}|(|\mathcal{A}_{\Gamma(O(i))}|+1) = 3 \cdot 3^4 \cdot (2 \cdot 2^4+1) = 8019$ constraints. As the node transition distribution (Table I) does not rely on neighbor identity, a mixed-mode function (MMF) is used instead, $p_i(x_i^{t+1} \mid x_i^t, \#(c_i^t), a_i^t)\ \forall i$, where $c_i^t \in \{0, \dots, |N(i)|\}$ is a count aggregator (CA) summarizing the number of neighbors on fire. The number of constraints is reduced to $|\mathcal{X}_i|(|N(i)|+1)(|\mathcal{A}_{\Gamma(O(i))}|+1) = 3 \cdot 5 \cdot (2 \cdot 2^4 + 1) = 495$. Further reduction can be achieved for GMDPs with actions also modeled according to the Anonymous Influence property. Our framework is still tractable for graphs where nodes may have many neighbors or the node state spaces are large.

B. Approximate $\mathcal{Q}$-Function Method

We now derive a novel ALP approach to produce a $\mathcal{Q}$-function to approximate a constrained value function using the following bound [20].

Proposition 2 ($\mathcal{Q}$-function approximation error)**.** *The difference $\epsilon = \max_{x^t\in\mathcal{X}} |v_w(x^t) - v^\star(x^t)|$ is bounded by the maximum difference between the approximate $\mathcal{Q}$-function $\mathcal{Q}_w(x^t, a^t)$ and the Bellman operator on $\mathcal{Q}_w(x^t, a^t)$,*

$$\epsilon \le \frac{2}{1-\gamma} \max_{x^t\in\mathcal{X}, a^t\in\mathcal{A}} |\mathcal{Q}_w(x^t, a^t) - (\mathcal{B}\mathcal{Q}_w)(x^t, a^t)|,$$

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

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

We assume the approximate $\mathcal{Q}$-function form,

$$\mathcal{Q}_w(x^t, a^t) = \sum_{i=1}^m w_i^T b_i(x^t_{\Gamma(O(i))}) + a_i^t w_i^T c_i(x^t_{\Gamma(O(i))}), \quad (8)$$

with $w_i \in \mathbb{R}^{k_i}$ and $b_i, c_i : \mathcal{X}_{\Gamma(O(i))} \subseteq \mathcal{X} \mapsto \mathbb{R}^{k_i}$, specifically for our capacity constrained formulations (Section IV). We assume actions are binary, $a_i^t \in \{0,1\}$, and a capacity constraint is imposed, $\sum_{i=1}^m a_i^t \le C$ where C is a non-negative integer. The set $\mathcal{A}_c$ denotes the set of feasible constrained actions. Minimizing $\phi = \max_{x^t\in\mathcal{X}, a^t\in\mathcal{A}_c} |\mathcal{Q}(x^t, a^t) - (\mathcal{B}\mathcal{Q})(x^t, a^t)|$ results in the non-linear program,

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

where the non-linearity is due to the maximization in the Bellman operator. We follow a similar procedure as Section III-A to develop a tractable and scalable ALP method.

Theorem 2. *For a GMDP containing $s \le m$ unique equivalence classes, the $\mathcal{Q}$-function ALP method requires solving s linear programs and $\sum_{k=1}^s n_k\phi_k$ is the approximation error.*

Proof. The constrained Bellman operator is,

$$\begin{aligned} (\mathcal{B}\mathcal{Q})(x^t, a^t) = \mathbb{E}_p\Bigg[& \sum_{i=1}^m r_i(x^t_{\Gamma(O(i))}, a^t_{\Gamma(O(i))}, x^{t+1}_{O(i)}) + \dots \\ & \gamma \max_{a^{t+1}\in\mathcal{A}} \sum_{i=1}^m w_i^T b_i(x^{t+1}_{O(i)}) + a_i^{t+1} w_i^T c_i(x^{t+1}_{O(i)}) \Bigg] \\ & \text{s. t.} \sum_{i=1}^m a_i^{t+1} \le C, \qquad a_i^{t+1} \in \{0,1\}. \end{aligned}$$

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

$$\begin{aligned} & \phi \ge \mathcal{Q}_w(x^t, a^t) - (\underline{\mathcal{B}\mathcal{Q}_w})(x^t, a^t), \\ & \phi \ge (\overline{\mathcal{B}\mathcal{Q}_w})(x^t, a^t) - \mathcal{Q}_w(x^t, a^t), \forall x^t \in \mathcal{X}, a^t \in \mathcal{A}_c. \end{aligned} \quad (9)$$

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 $a_i^{t+1} = 0\ \forall i$ thus $\left(\underline{\mathcal{B}\mathcal{Q}_w}\right) = \sum_{i=1}^m \mathbb{E}_{p_i}\left[r_i + \gamma w_i^T b_i\right]$. An upper bound is calculated by removing the capacity constraint and choosing actions to improve the total value of $\mathcal{Q}_w$, $\left(\overline{\mathcal{B}\mathcal{Q}_w}\right) = \sum_{i=1}^m \mathbb{E}_{p_i}\left[r_i + \gamma w_i^T b_i + \gamma \max\{0, w_i^T c_i\}\right]$.

The maximization in the upper bound is replaced by two linear constraints and the error ϕ is decomposed via (5). The result is the following per-node linear program,

$$\begin{aligned} \min_{w_i\in\mathbb{R}^{k_i}, \phi_i\in\mathbb{R}} \quad & \phi_i \\ \text{s. t.} \quad & \\ & \phi_i \ge w_i^T b_i + a_i^t w_i^T c_i - \mathbb{E}\left[r_i + \gamma w_i^T b_i\right], \\ & \phi_i \ge \mathbb{E}_{p_i}\left[r_i + \gamma w_i^T b_i\right] - w_i^T b_i - a_i^t w_i^T c_i, \\ & \phi_i \ge \mathbb{E}_{p_i}\left[r_i + \gamma w_i^T b_i + \gamma w_i^T c_i\right] - w_i^T b_i - a_i^t w_i^T c_i, \\ & \quad \forall x^t_{\Gamma(O(i))} \in \mathcal{X}_{\Gamma(O(i))}, a^t_{\Gamma(O(i))} \in \mathcal{A}_{\Gamma(O(i))}. \end{aligned} \quad (10)$$

Each linear program contains $k_i + 1$ variables and $3|\mathcal{X}_{\Gamma(O(i))}||\mathcal{A}_{\Gamma(O(i))}|$ constraints. Solving Program (10) for each node 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 (10) cannot lower the error ϕ_i.

The approximate $\mathcal{Q}$-function is determined after solving for the weights w_i. The linear program for two nodes have identical solutions w_i, ϕ_i if both share the same reward, transition, and basis functions, i.e. both are in the same equivalence class. Therefore, for m classes, m programs are solved to determine the approximate $\mathcal{Q}$-function. Only s programs are solved for $s < m$ classes as the solution is identical for all nodes in the same class. $\square$

For the general case of arbitrary discrete action spaces, the lower bounds any feasible action. Enumerating the sets $\mathcal{X}_{\Gamma(O(i))}$ and $\mathcal{A}_{\Gamma(O(i))}$ for Program (10) may be computationally prohibitive so we exploit Anonymous Influence to instead use CAFs and MMFs where possible to reduce the cardinality of these sets.

IV. Capacity Constrained Formulations

We now derive an approximate method for applying and satisfying a control constraint for the case of binary actions, $a_i^t \in \{0,1\}$. We first introduce a class of linear programs that have a capacity constraint and an explicit solution. For the following discussion, we represent the action $a^t \in \mathcal{A}$ as a vector $a^t \in \mathbb{R}^m$ with elements $a_i^t \in \{0,1\}$ and use 1 to represent a vector of all ones.

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

$$\begin{aligned} \max_{a^t \in \mathcal{A}} \quad & d + l^T a^t \\ \text{s. t.} \quad & 1^T a^t \leq C, \quad a_i^t \in \{0,1\}, \end{aligned} \tag{11}$$

where $d \in \mathbb{R}$, $l \in \mathbb{R}^m$, and C is a non-negative integer, has an explicit solution. Assume l is ordered such that $l_1 \geq l_2 \geq \cdots \geq l_m$. An optimal solution is,

$$a_i^t = \begin{cases} 1 & \text{if } i \leq C \text{ and } l_i \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Proof. The values l_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 $i \in \{1, \ldots, j\}$ and zero otherwise. If $j = C$, then choosing $a_k^t = 1$ for $k > j$ will violate the constraint. If $j < C$, then choosing $a_k^t = 1$ for $k > j$ lowers the objective as l_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 l_k must be non-negative. $\square$

Theorem 3. *For approximate value functions v_w, if the form of the transition distribution (1), reward functions (3), and basis functions result in,*

$$\mathbb{E}_p\left[R(x^t, a^t, x^{t+1}) + \gamma v_w(x^{t+1})\right] = d + l^T a^t, \tag{13}$$

then the constrained policy is determined by (12). Furthermore, for approximate $\mathcal{Q}$-functions of the form in (8), the constrained policy is determined by (12).

Proof. The approximate value function is determined after solving for the weights of the basis representation. If the relationship in (13) holds, then the constrained policy is,

$$\begin{aligned} \pi(x^t) = \operatorname*{argmax}_{a^t \in \mathcal{A}} \; & d + l^T a^t \\ \text{s. t. } & 1^T a^t \leq C, \quad a_i^t \in \{0,1\}. \end{aligned}$$

Therefore, the policy $\pi(x^t)$ is determined by (12). For approximate $\mathcal{Q}$-functions, if the assumed form (8) is used then the constrained policy is,

$$\begin{aligned} \pi(x^t) = \operatorname*{argmax}_{a^t \in \mathcal{A}} \; & \sum_{i=1}^{m} w_i^T b_i(x^t_{\Gamma(O(i))}) + a_i^t w_i^T c_i(x^t_{\Gamma(O(i))}) \\ \text{s. t. } & 1^T a^t \leq C, \quad a_i^t \in \{0,1\}. \end{aligned} \tag{14}$$

Let $d = \sum_{i=1}^{m} w_i^T b_i(x^t_{\Gamma(O(i))})$ and $l_i = w_i^T c_i(x^t_{\Gamma(O(i))})$. The constrained maximization (14) is equivalent to Program (11) so the policy is determined by (12). $\square$

Although the policy (12) exactly solves Program (11), it is necessary in our derivations to remove the capacity constraint when determining an approximate function v_w or $\mathcal{Q}_w$ to develop tractable methods. As a result, the approximate functions are analogous to unconstrained solutions found via dynamic programming (DP). The use of (12) with an unconstrained approximate solution is therefore an approximation to the true constrained policy determined by a method that includes the control constraint (e.g. constrained DP).

V. Experiments

A. Wildfire Management

We use the value function ALP method with reward and basis functions,

$$r_i(x^t_{\Gamma(i)}) = \mathbf{1}_H(x_i^t) - \mathbf{1}_F(x_i^t) \sum_{j \in N(i)} \mathbf{1}_H(x_j^t),$$

$$w_i^T h_i(x^t_{\Gamma(i)}) = w_0 + w_1 \mathbf{1}_H(x_i^t) + w_2 \mathbf{1}_F(x_i^t) \sum_{j \in N(i)} \mathbf{1}_H(x_j^t).$$

The parameters are $\alpha = 0.2$, $\beta = 0.90$, $\Delta\beta = 0.54$, and $\gamma = 0.95$. The forest is a 50×50 sized grid with an initial square grid of 16 fires at the center and the capacity is $C = 4$. Simulations terminate when no trees are on fire. The performance metric for a simulation is $z = \sum_{i=1}^{m} \mathbf{1}_H(x_i^{t_{\text{final}}})/50^2$, the fraction of remaining healthy trees, and 1,000 simulations were run. We assume $N(i) = 4 \; \forall i$ so that there is one equivalence class. The derived policy is then applied to the original graph model. The median value of z and the program error ϕ_i are given in Table II. The node reward functions are chosen so that [5] can be combined with a capacity constraint and used for a comparison as other methods are not tractable for the wildfire model. The resulting policy from [5] performs poorly compared to our method; see Figure 3 and Table II.

B. 2014 West Africa Ebola Outbreak

A graph-based model was used to describe three countries affected by the 2014 Ebola outbreak (Figure 1). A discrete model is used for each community where the state x_i^t is one of three values, $\mathcal{X}_i = \{\text{susceptible, infected, removed}\} = \{S, E, R\}$. The transition distribution is shown in Table III

TABLE II: Results for wildfire management.

Wildfires	Error ϕ_i	Remaining Healthy Trees
Approximate $v_w(x^t)$	1.98	98%
Approximate $v_w(x^t)$ [5]	2.30	1%
No control	—	1%

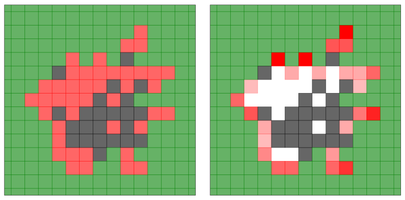

Fig. 3: (both) Policies for a single time step of a wildfire simulation. Green cells are healthy trees and black are burnt trees. Fire color indicates control preference: white is low and dark red is high. (left) Policy using basis from [5] which treats all fires equally. (right) Our policy which prioritizes fires based on number of neighboring healthy trees.

where $e_i^t = \sum_{j \in N(i)} \mathbf{1}_S(x_j^t)$. The $\mathcal{Q}$-function ALP is used with reward and basis functions,

$$r_i(x_i^t, x_i^{t+1}) = \mathbf{1}_S(x_i^t) - \mathbf{1}_E(x_i^{t+1}),$$
$$w_i^T b_i(x_i^t) = w_0 + w_1 \mathbf{1}_S(x_i^t) + w_2 \mathbf{1}_E(x_i^t),$$
$$a_i^t w_i^T c_i(x_{\Gamma(i)}^t) = a_i^t w_3 \mathbf{1}_E(x_i^t) \sum_{j \in N(i)} \mathbf{1}_S(x_j^t).$$

As a simplification, we assume that each community has the same number of neighbors, $|N(i)| = 4 \; \forall i$, so there is a single equivalence class. The derived policy is then applied to the original graph model.

The parameters $\eta = 0.14$ and $\nu = 0.12$ were derived from World Health Organization [3] data. The capacity is $C = 3$ and the discount factor is $\gamma = 0.9$. Each simulation is initialized with three communities as infected: Guieckedou (Guinea), Kailahun (Sierra Leone), and Lofa (Liberia). Simulations terminate when no communities are infected. The performance metric z is the median number of weeks a community is infected, i.e. the median value of $y_{\text{community}} = \sum_{\tau=1}^{t_{\text{final}}} \mathbf{1}_E(x_i^\tau) \; \forall i \in V$. From the data, the median of $y_{\text{community}}$ is 36 weeks and the maximum is 112 weeks. For 1,000 simulations of our policy, the median of z is 36 weeks and the maximum is 125.5 weeks.

VI. Conclusion

This work formalizes a class of graph-based models with structure based on inter-MDP influence, Symmetry, and Anonymous Influence. We derived two methods that leverage this structure to generate approximate value and $\mathcal{Q}$-functions as well as constrained policies for domains in which enumerating the state and action spaces is intractable. For future work, we plan to investigate the relationship between the performance of the constrained policy and the approximations required to develop tractable methods. In addition, the Anonymous Influence property may be useful as

TABLE III: Single community dynamics for Ebola model.

x_i^t \ x_i^{t+1}	S	E	R
S	$1-\eta e_i^t$	ηe_i^t	0
E	0	$1-\nu a_i^t$	νa_i^t
R	0	0	1

a simplification for domains where this property is not valid. Experiments in other application domains are also planned to demonstrate the versatility of our methods.

References

[1] 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.

[2] P. Robbel, F. Oliehoek, and M. Kochenderfer, "Exploiting anonymity in approximate linear programming: Scaling to large multiagent mdps," in *AAAI Conference on Artificial Intelligence*, 2016.

[3] World Health Organization (WHO), "Ebola data and statistics," http://apps.who.int/gho/data/node.ebola-sitrep, accessed 2018-03-19.

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

[5] 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.

[6] 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.

[7] 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.

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

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] A. Somanath, S. Karaman, and K. Youcef-Toumi, "Controlling stochastic growth processes on lattices: Wildfire management with robotic fire extinguishers," in *IEEE Conference on Decision and Control (CDC)*, 2014, pp. 1432–1437.

[16] D. Blatner, *Spectrums: Our Mind-boggling Universe from Infinitesimal to Infinity*. A&C Black, 2013.

[17] C. Guestrin, D. Koller, and R. Parr, "Multiagent planning with factored mdps," in *Advances in Neural Information Processing Systems*, 2002.

[18] 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.

[19] D. P. De Farias and B. Van Roy, "The linear programming approach to approximate dynamic programming," *Operations research*, vol. 51, no. 6, pp. 850–865, 2003.

[20] R. Williams and L. C. Baird, "Tight performance bounds on greedy policies based on imperfect value functions," Tech. Rep., 1993.