Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Exploiting Anonymity in Approximate Linear Programming: Scaling to Large Multiagent MDPs

Philipp Robbel
MIT Media Lab
Cambridge, MA, USA

Frans A. Oliehoek
University of Amsterdam
University of Liverpool

Mykel J. Kochenderfer
Stanford University
Stanford, CA, USA

Abstract

Many solution methods for Markov Decision Processes (MDPs) exploit structure in the problem and are based on value function factorization. Especially multiagent settings, however, are known to suffer from an exponential increase in value component sizes as interactions become denser, restricting problem sizes and types that can be handled. We present an approach to mitigate this limitation for certain types of multiagent systems, exploiting a property that can be thought of as "anonymous influence" in the factored MDP. We show how representational benefits from anonymity translate into computational efficiencies, both for variable elimination in a factor graph and for the approximate linear programming solution to factored MDPs. Our methods scale to factored MDPs that were previously unsolvable, such as the control of a stochastic disease process over densely connected graphs with 50 nodes and 25 agents.

1 Introduction

Cooperative multiagent systems (MASs) present an important framework for modeling the interaction between agents that collaborate to solve a task. Decision-theoretic models like the Markov Decision Process (MDP) have seen widespread use to address such complex stochastic planning problems. Multiagent settings, however, have state and action spaces that tend to grow exponentially with the agent number, making common solution methods that rely on the full enumeration of the joint spaces prohibitive.

Many problem representations thus attempt to exploit structure in the domain to improve efficiency. Single and multiagent factored MDPs (F(M)MDPs) represent the problem in terms of state and action spaces that are spanned by a number of variables, or factors (Boutilier, Dean, and Hanks 1999; Guestrin, Koller, and Parr 2002). Unfortunately, the representational benefits from factored descriptions do not in general translate into gains for policy computation (Koller and Parr 1999). Still, many solution methods successfully exploit structure in the domain, both in exact and approximate settings, and have demonstrated scalability to large state spaces (Hoey et al. 1999; Raghavan et al. 2012; Cui et al. 2015).

In this paper we focus on approaches that additionally address larger numbers of agents through value factorization, assuming that smaller value function components can approximate the global value function well (Guestrin et al. 2003; Kok and Vlassis 2006). The approximate linear programming (ALP) approach of Guestrin et al. (2003) is one of the few approaches in this class that retains no exponential dependencies in the number of agents and variables through the efficient computation of the constraints in the linear program based on a variable elimination (VE) method. While the approach improved scalability dramatically, the method retains an exponential dependency on the induced tree-width (the size of the largest intermediate term formed during VE), meaning that its feasibility depends fundamentally on the connectivity and scale of the factor graph defined by the FMMDP and chosen basis function coverage.

We present an approach that aims to mitigate the exponential dependency of VE on the induced width, which is caused by the need to represent all combinations of state and action variables that appear in each manipulated factor. In many domains, however, different combinations lead to similar effects, or *influence*. Serving as a running example is a disease control scenario over large graphs (Ho et al. 2015; Cheng et al. 2013). In this setting the aggregate infection rate of the parent nodes, independent of their individual identity, fully defines the behavior of the propagation model. This observation extends to many MASs that are more broadly concerned with the control of dynamic processes on networks, e.g. with stochastic fire propagation models or energy distribution in power grids (Liu, Slotine, and Barabasi 2011; Cornelius, Kath, and Motter 2013).

We propose to exploit this anonymity of influences for more efficient solution methods for MMDPs. In particular:

1) We introduce a novel *redundant representation (RR)* for the factors that VE manipulates which involves count aggregators. *2)* We show how to derive an efficient VE algorithm, RR-VE, that makes use of the redundant representation, and prove its correctness. *3)* We then propose RR-ALP, which extends the ALP approach by making use of RR-VE, and maintains identical solutions. *4)* We show an empirical evaluation of our methods that demonstrates speed-ups of the ALP by an order of magnitude in a sampled set of random disease propagation graphs and scale to problem sizes that were previously infeasible to solve with the ALP.

2 Background

We first discuss the background on factored MMDPs and their solution methods that are based on value factorization.

Factored Multiagent MDPs

Markov decision processes are a general framework for decision making under uncertainty (Kochenderfer 2015). An infinite-horizon Markov decision process (MDP) is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$, where $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$ and $\mathcal{A} = \{a_1, \ldots, a_{|\mathcal{A}|}\}$ are the finite sets of states and actions, T the transition probability function specifying $P(s' \mid s, a)$, $R(s, a)$ the immediate reward function, and $\gamma \in [0, 1]$ the discount factor of the problem.

Factored multiagent MDPs (FMMDPs) exploit structure in state and action spaces by defining system state and joint action with an assignment to state and action variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ and $\mathbf{A} = \{A_1, \ldots, A_g\}$. Transition and reward function decompose into a two-slice temporal Bayesian network (2TBN) consisting of independent factors (Boutilier, Dean, and Hanks 1999; Guestrin, Koller, and Parr 2002). The FMMDP transition function can be written as

$$P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a}) = \prod_i T_i(x'_i \mid \mathbf{x}[\mathrm{Pa}(X'_i)], \mathbf{a}[\mathrm{Pa}(X'_i)]) \quad (1)$$

where $\mathrm{Pa}(X'_i)$ refers to the parent nodes of X'_i in the 2TBN and $\mathbf{x}[\mathrm{Pa}(X'_i)]$ to their value in state $\mathbf{x}$. Collaborative FMMDPs assume that each agent i observes part of the global reward and is associated with a local reward function, i.e. $R(\mathbf{x}, \mathbf{a}) = \sum_{i=1}^{g} R_i(\mathbf{x}[\mathbf{C}_i], \mathbf{a}[\mathbf{D}_i])$ given sets $\mathbf{C}_i$ and $\mathbf{D}_i$.

The solution to an (M)MDP is a (joint) policy that optimizes the expected sum of discounted rewards that can be achieved from any state. The optimal value function $\mathcal{V}^*(\mathbf{x})$ represents the maximum expected return posssible from every state (Puterman 2005). Such an (optimal) value function can be used to extract an (optimal) policy by performing a *back-projection* through the transition function to compute the so-called Q-function:

$$\forall_{\mathbf{x},\mathbf{a}} \quad \mathcal{Q}^*(\mathbf{x}, \mathbf{a}) = R(\mathbf{x}, \mathbf{a}) + \gamma \sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a}) \mathcal{V}^*(\mathbf{x}), \quad (2)$$

and subsequently acting greedy with respect to the Q-function: the optimal action at $\mathbf{x}$ is $\mathbf{a}^* = \arg\max \mathcal{Q}^*(\mathbf{x}, \mathbf{a})$.

Control of Epidemics on Graphs

The susceptible-infected-susceptible (SIS) model has been well-studied in the context of disease outbreak dynamics (Bailey 1957) but only few approaches consider the complex *control* problem of epidemic processes (Nowzari, Preciado, and Pappas 2015; Ho et al. 2015).

SIS dynamics with homogeneous parameters are modeled as an FMMDP as follows. We define the network as a (directed or undirected) graph $G = (V, E)$ with controlled and uncontrolled vertices $V = (V_c, V_u)$ and edge set $E \subseteq V \times V$. The state space $\mathcal{S}$ is spanned by state variables $X_1, \ldots, X_n$, one per associated vertex V_i, encoding the health of that node. The action set $\mathbf{A} = \{A_1, \ldots, A_{|V_c|}\}$ factors similarly over the controlled vertices V_c in the graph and denote an active modulation of the flow out of node $V_i \in V_c$. Note that this model assumes binary state variables $X_i = \{0, 1\} = \{\text{healthy}, \text{infected}\}$, and actions $A_i = \{0, 1\} = \{\text{do not vaccinate}, \text{vaccinate}\}$ and that $A_u = \{0\}$ for all uncontrolled nodes V_u.

The transition function factors on a per-node basis into $T_i(x'_i \mid \mathbf{x}[\mathrm{Pa}(X'_i)], a_i)$ defined as:

$$T_i \triangleq \begin{cases} (1 - a_i)(1 - \prod_j (1 - \beta_{ji} x_j)) & \text{if } x_i = 0 \\ (1 - a_i)\,(1 - \delta_i) & \text{otherwise} \end{cases} \quad (3)$$

distinguishing the two cases that X_i was infected at the previous time step (bottom) or not (top). Parameters β_{ji} and δ_i are the known infection transmission probabilities from node j to i, and node i's recovery rate, respectively. The reward function factors as $R(\mathbf{x}, \mathbf{a}) = -\lambda_1 \|\mathbf{a}\|_1 - \lambda_2 \|\mathbf{x}\|_1$ where the L_1 norm records a cost λ_2 per infected node and an action cost λ_1 per vaccination action at a controlled node.

Efficient Solution of Large FMMDPs

We build upon the work by Guestrin et al. (2003), who present approximate solution methods for FMMDPs that are particularly scalable in terms of the agent number. The approach represents the global Q-value function as the sum of appropriately chosen smaller value function components, each defined over a subset of state and action variables (referred to as factored linear value functions). To compute a factored linear value function, they present an efficient method to compactly represent the constraints in the approximate linear programming (ALP) solution to MDPs.

Factored Value Functions Value factorization represents the joint value function as a linear combination of locally-scoped terms. In the case of factored *linear* value functions given basis functions $H = \{h_1, \ldots, h_k\}$, $\mathcal{V}$ can be written as the linear combination $\mathcal{V}(\mathbf{x}) = \sum_{j=1}^{k} w_j h_j(\mathbf{x})$ where h_j is defined over some subset of variables $\mathbf{C}_{h_j} \subseteq \mathbf{X}$ (omitted for clarity), and w_j is the weight associated with basis h_j.

Factored linear (state) value functions induce factored Q-value functions if transitions and rewards are factored into local terms. This is because the expectation over an individual basis functions (called a basis back-projection) can be computed efficiently, avoiding the sum over exponentially many successor states in Equation 2.

Definition 1 (Basis back-projection)**.** Given a basis function $h_j : \mathbf{C} \to \mathbb{R}$, defined over scope $\mathbf{C} \subseteq \mathbf{X}$, and a factored 2TBN transition model $P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})$ (see Equation 1), define the *basis back-projection* of h_j (Guestrin 2003):

$$\begin{aligned} g_j(\mathbf{x}, \mathbf{a}) &= \textstyle\sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})\, h_j(\mathbf{x}'[\mathbf{C}]) \\ &= \textstyle\sum_{\mathbf{c}'} P(\mathbf{c}' \mid \mathbf{x}[\mathrm{Pa}(\mathbf{C})], \mathbf{a}[\mathrm{Pa}(\mathbf{C})])\, h_j(\mathbf{c}') \end{aligned} \quad (4)$$

where $\mathrm{Pa}(\mathbf{C}) \triangleq \bigcup_{X_i \in \mathbf{C}} \mathrm{Pa}(X_i)$ denotes the union of respective parent (state and action) variables in the 2TBN.

Functions g_j are thus again locally-scoped, defined precisely over the parent scope $\mathrm{Pa}(\mathbf{C})$ (omitted for clarity in the remainder of the presentation). Basis back-projections are used to compute a factored Q-value function:

$$\mathcal{Q}(\mathbf{x}, \mathbf{a}) = \textstyle\sum_r R_r(\mathbf{x}[\mathbf{C}_r], \mathbf{a}[\mathbf{D}_r]) + \gamma \sum_j w_j g_j(\mathbf{x}, \mathbf{a}) \quad (5)$$

The factor graph spanned by a factored Q-value function is in this context often referred to as a *coordination graph* (CG).

VE The variable elimination (VE) algorithm can be used for computing the max over a set of locally-scoped functions in a factor graph efficiently. Similarly to maximum a posteriori (MAP) estimation in Bayesian networks, VE maximizes over single variables at a time rather than enumerating all possible joint configurations followed by picking the maximizing one (Koller and Friedman 2009). Variable elimination performs two operations, AUGMENT and REDUCE, for every variable X_l to be eliminated. When considering the maximization over the state space, AUGMENT corresponds to the sum of functions that depend on X_l, and REDUCE to the maximization over X_l in the result. The execution time is exponential in the size of the largest intermediate term formed which depends on the chosen elimination order. While the problem of determining the optimal elimination order is NP-complete, effective heuristics for variable ordering exist in practice (Koller and Friedman 2009).

ALP The ALP method for solving MDPs computes the best approximation (in a weighted L_1 norm sense) to the optimal value function in the space spanned by the basis functions H (Puterman 2005). The ALP yields a solution in time polynomial in the sizes of $\mathcal{S}$ and $\mathcal{A}$ but these are exponential for MASs.

Guestrin (2003) introduces an efficient implementation of the ALP for factored linear value functions that avoids exponentially many constraints in the ALP. It is based on the insight that all (exponentially many) constraints in the ALP can be reformulated as follows:

$$\begin{array}{lrcl} \forall \mathbf{x}, \mathbf{a} & \mathcal{V}(\mathbf{x}) & \geq & R(\mathbf{x},\mathbf{a}) + \gamma \sum_i w_i\, g_i(\mathbf{x},\mathbf{a}) \\ \forall \mathbf{x}, \mathbf{a} & 0 & \geq & R(\mathbf{x},\mathbf{a}) + \sum_i w_i[\gamma g_i(\mathbf{x},\mathbf{a}) - h_i(\mathbf{x})] \\ \Rightarrow & 0 & \geq & \max_{\mathbf{x},\mathbf{a}}[\sum_r R_r(\mathbf{x}[\mathbf{C}_r], \mathbf{a}[\mathbf{D}_r]) + \\ & & & \sum_i w_i[\gamma g_i(\mathbf{x},\mathbf{a}) - h_i(\mathbf{x})]] \end{array} \tag{6}$$

The reformulation replaces the exponential set of linear constraints with a *single* non-linear constraint (last row in Equation 6). Using a procedure similar to VE, this max constraint can be implemented with a small set of linear constraints.

3 Anonymous Influence

At the core of the ALP solution above lies the assumption that VE can be carried out efficiently in the factor graph spanned by the local functions that make up the max constraint of Equation 6, i.e. that the scopes of all intermediate terms during VE remain small. This assumption is often violated in many graphs of interest, e.g. in disease control where nodes may possess large in- or out-degrees.

In this section we develop a novel approach to deal with larger scope sizes when only the *joint effects* of sets of variables—rather than their identity—suffices to compactly describe the factors that appear in the max constraint and are manipulated during VE. We introduce a novel representation that is exponentially smaller than the equivalent full encoding of intermediate terms and show how VE retains correctness. We assume binary variables but the results carry over to the discrete variable setting.

Mixed-Mode Functions

We define count aggregator functions to summarize the "anonymous influence" of a set of variables. In the disease propagation scenario for example, the number of active parents uniquely defines the transition model T_i.

Definition 2 (Count Aggregator)**.** Let $\mathbf{Z} = \{Z_1, \ldots, Z_{|\mathbf{Z}|}\}$ be a set of binary variables, $Z_i \in \{0, 1\}$. The *count aggregator (CA)* $\#\{\mathbf{Z}\} : Z_1 \times \ldots \times Z_{|\mathbf{Z}|} \mapsto \{0, \ldots, |\mathbf{Z}|\}$ is defined as: $\#\{\mathbf{Z}\}(\mathbf{z}) \triangleq \sum_{i=1}^{|\mathbf{Z}|} z_i$. $\mathbf{Z}$ is also referred to as the *count scope* of CA $\#\{\mathbf{Z}\}$.

Hence, CAs simply summarize the number of variables that appear 'enabled' in its domain. Conceptual similarities with generalized (or 'lifted') counters in first-order inference are discussed in Section 7. Functions that rely on CAs can be represented compactly.

Definition 3 (Count Aggregator Function)**.** A *count aggregator function (CAF)*, is a function $f : \mathbf{Z} \to \mathbb{R}$ that maps $\mathbf{Z}$ to the reals by making use of a CA. That is, there exists a function $\mathfrak{f} : \{0, \ldots, |\mathbf{Z}|\} \to \mathbb{R}$ such that f is defined as

$$f(\mathbf{z}) \triangleq [\mathfrak{f} \circ \#\{\mathbf{Z}\}](\mathbf{z}). \tag{7}$$

To make clear f's use of a CA, we use the notation $f(\#(\mathbf{z}))$.

CAFs have a *compact representation* which is precisely the function $\mathfrak{f}$. It is compact, since it can be represented using $|\mathbf{Z}| + 1$ numbers and $|\mathbf{Z}| + 1 \ll 2^{|\mathbf{Z}|}$.

We now introduce so-called "mixed-mode" functions f that depend both on CAs and on other variables $\mathbf{X}$ that are not part of any CA:

Definition 4 (Mixed-Mode Function)**.** A function $f : \mathbf{X} \times \mathbf{Z} \to \mathbb{R}$ is called a *mixed-mode function* (MMF), denoted $f(\mathbf{x}, \#(\mathbf{z}))$, if and only if $\forall \mathbf{x}\ \exists f_{\mathbf{X}}\ s.t.\ f(\mathbf{x}, \mathbf{z}) = f_{\mathbf{X}}(\#(\mathbf{z}))$. That is, for each instantiation $\mathbf{x}$, there exists a CAF $f_{\mathbf{X}}(\#(\mathbf{z}))$. We refer to $X_i \in \mathbf{X}$ as *proper variables* and $Z_j \in \mathbf{Z}$ as *count variables* in the scope of f.

Consider $T_i(X_i \mid \mathrm{Pa}(X_i))$ in the (binary) disease propagation graph. Then $T_i(X_i \mid \#\{\mathrm{Pa}(X_i)\})$ is a *mixed-mode function* that induces two CAFs, one for x_i and one for $\bar{x}_i$.

Mixed-mode functions generalize simply to those with multiple CAs, $f : \mathbf{X} \times \mathbf{Z}_1 \times \ldots \times \mathbf{Z}_N \to \mathbb{R}$, denoted $f(\mathbf{x}, \#_1(\mathbf{z}_1), \ldots, \#_N(\mathbf{z}_N))$. The following cases can occur:

1) MMFs with *fully disjoint scopes* have mutually disjoint proper and count variable sets (i.e., $\mathbf{X} \cap \mathbf{Z}_i = \emptyset\ \forall i = 1, \ldots, N$ and $\mathbf{Z}_i \cap \mathbf{Z}_j = \emptyset\ \forall i \neq j$); *2)* MMFs have *shared proper and count variables* (if and only if $\exists i\ s.t.\ \mathbf{X} \cap \mathbf{Z}_i \neq \emptyset$); *3)* MMFs have *non-disjoint counter scopes* (if and only if $\exists (i, j), i \neq j\ s.t.\ \mathbf{Z}_i \cap \mathbf{Z}_j \neq \emptyset$).

Summarizing, it is possible to represent certain anonymous influences using mixed-mode functions. In the following we will show that these can be compactly represented, which subsequently forms the basis for a more efficient VE algorithm.

Compact Representation of MMFs

Just as CAFs, a mixed-mode function f has a compact *representation* $\mathfrak{f} : \mathbf{X} \times \{0, ..., |\mathbf{Z}|\} \to \mathbb{R}$ where $f(\mathbf{x}, \#(\mathbf{z})) \triangleq$

$\mathfrak{f}(\mathbf{x}, \sum_{i=1}^{|\mathbf{Z}|} z_i)$. A mixed-mode function f can thus be described with (at most) $K^{|\mathbf{X}|}(|\mathbf{Z}|+1)$ parameters where K is an upper bound on $|Dom(X_i)|$.

As mentioned before, we also consider MMFs with multiple CAs. In particular, let us examine a function $f(\#_1(a,b), \#_2(b,c))$ with two CAs that have a overlapping scope since both depend on shared variable B. In order to consistently deal with overlaps in the count scope, previous work has considered so-called shattered representations (Taghipour et al. 2013; Milch et al. 2008). A MMF with overlapping count scopes $f(\#_1(a,b), \#_2(b,c))$ can always be transformed into an equivalent one without overlapping count scopes $f'(\#'_1(a), \#'_2(c), \#(b)) \triangleq f(\#_1(a,b), \#_2(b,c))$.

We can now distinguish between different *representations* of these MMFs with overlapping count scopes.

Definition 5 (Shattered Representation)**.** The *shattered representation* of f is the representation of f', i.e.

$$f(\#_1(a,b), \#_2(b,c)) \triangleq \mathfrak{f}(k_1, k_2, k_3)$$

where $k_1 := a$, $k_2 := c$, $k_3 := b$ and $\mathfrak{f} : \{0,1\} \times \{0,1\} \times \{0,1\} \to \mathbb{R}$.

We introduce a novel *redundant representation* of f. Redundant representations retain compactness with many overlapping count scopes. This becomes relevant when we introduce operations on MMFs (e.g., for variable elimination).

Definition 6 (Redundant Representation)**.** The redundant representation of MMF $f(\#_1(a,b), \#_2(b,c))$ is a function $\mathfrak{f} : \{0,1,2\} \times \{0,1,2\} \to \mathbb{R}$:

$$f(\#_1(a,b), \#_2(b,c)) \triangleq \mathfrak{f}(k_1, k_2)$$

where $k_1 := a + b$ and $k_2 := b + c$.

If we choose to store MMFs with redundant representations, we may introduce incompatible assignments to variables that appear in overlapping count scopes. The following definition formalizes this observation.

Definition 7 (Consistent Count Combination)**.** Let $\#_1\{A,B\}, \#_2\{B,C\}$ be two CAs with overlapping count scopes. We say that a pair (k_1, k_2) is a *consistent count combination (consistent CC) for* $\#_1, \#_2$ if and only if there exists an assignment (a,b,c) such that $(k_1,k_2) = (\#_1(a,b), \#_2(b,c))$. If no such (a,b,c) exists, then (k_1,k_2) is called an *inconsistent* CC. Further, let $f(\#_1, \#_2)$ be a MMF. We say that a consistent CC (k_1,k_2) for $\#_1, \#_2$ is a *consistent entry* $\mathfrak{f}(k_1,k_2)$ of the representation of f. Similarly, if (k_1,k_2) is an inconsistent CC, then $\mathfrak{f}(k_1,k_2)$ is referred to as an *inconsistent entry*.

Inconsistent entries can only occur in redundant representations since the shattered representation of f is defined for f' without overlapping count scopes. Even though redundant representations appear to have a disadvantage since they contain incostent entries, they also have a big advantage since they can be exponentially more compact than shattered ones:

Lemma 1. *Consider MMF* $f : \mathbf{X} \times \mathbf{Z}_1 \times \ldots \times \mathbf{Z}_N \to \mathbb{R}$, $N \geq 2$. *Let* $\mathbf{Z} = \bigcup_{i=1}^{N} \mathbf{Z}_i$. *In the worst case, a partition of* $\mathbf{Z}$ *requires* $p = \min\{2^N - 1, |\mathbf{Z}|\}$ *splits into mutually disjoint sets and the shattered representation of* f *is of size* $O(S^p)$ *where* S *is an upper bound on the resulting set sizes. The same function has a redundant representation of size* $O(K^N)$ *where* K *is an upper bound on* $|\mathbf{Z}_i| + 1$.

Consider, e.g., an MMF $f(\#_1\{A,B,C,D,E\}, \#_2\{A,B,X,Y,Z\}, \#_3\{A,C,W,X\})$ with overlapping count scopes. The redundant representation of f requires $6 \cdot 6 \cdot 5 = 180$ parameters but contains *inconsistent entries*. The shattered representation defined using equivalent MMF $f'(\#\{A\}, \#\{B\}, \#\{C\}, \#\{D,E\}, \#\{X\}, \#\{W\}, \#\{Y,Z\})$ requires 288 parameters.

We now show how mixed-mode functions with compact redundant representations can be exploited during variable elimination and during constraint generation in the ALP.

4 Efficient Variable Elimination

Here we describe how AUGMENT and REDUCE are efficiently implemented to work *directly* on the redundant representations of MMFs. Particular care has to be taken to ensure correctness since we observed previously that reduced representations contain inconsistent entries.

AUGMENT takes a set of MMFs and adds them together. We implement this operation directly in the redundant representation. AUGMENT$(\mathfrak{g}, \mathfrak{h})$ returns a function $\mathfrak{f}$ that is defined as: $\forall x, y, k_1 \in \{0, \ldots, N_1\}, k_2 \in \{0, \ldots, N_2\}$

$$\mathfrak{f}(x, y, k_1, k_2) = \mathfrak{g}(x, k_1) + \mathfrak{h}(y, k_2). \tag{8}$$

The implementation simply loops over all x, y, k_1, k_2 to compute all entries (consistent or inconsistent).

REDUCE maxes out a variable. Here we show how this operation is implemented for MMFs directly using the redundant representation. Let $g(x, y, z, \#_1(a,b,z), \#_2(b,c))$ be a MMF with redundant representation $\mathfrak{g}(x, y, z, k_1, k_2)$. We discriminate different cases:

1) Maxing out a proper variable: If we max out x, $\mathfrak{f}(y, z, k_1, k_2) \triangleq \max\{\mathfrak{g}(0, y, z, k_1, k_2), \mathfrak{g}(1, y, z, k_1, k_2)\}$

2) Maxing out a non-shared count variable: If we max out a, $\mathfrak{f}(x, y, z, k_1, k_2) \triangleq \max\{\mathfrak{g}(x, y, z, k_1, k_2), \mathfrak{g}(x, y, z, k_1 + 1, k_2)\}$. The resulting function has signature $f(x, y, z, \#'_1(b,z), \#'_2(b,c))$. The values of x, y, z, b, c are fixed (by the l.h.s. of the definition) in such a way that $\#'_1(b,z) = k_1$ and $\#'_2(b,c) = k_2$. The maximization that we perform over $a \in \{0,1\}$ therefore has the ability to increase k_1 by 1 or not, which leads to the above maximization in the redundant representation.

3) Maxing out a shared count variable: If we max out b, $\mathfrak{f}(x, y, z, k_1, k_2) \triangleq \max\{\mathfrak{g}(x, y, z, k_1, k_2), \mathfrak{g}(x, y, z, k_1 + 1, k_2 + 1)\}$ This is similar to the previous case, but since b occurs in both $\#'_1$ and $\#'_2$, it may either increase both k_1 and k_2, or neither.

4) Maxing out a shared proper/count variable: In case we max out z, $\mathfrak{f}(x, y, k_1, k_2) \triangleq \max\{\mathfrak{g}(x, y, 0, k_1, k_2), \mathfrak{g}(x, y, 1, k_1 + 1, k_2)\}$. Since z occurs as both proper and count variable (in $\#_1$), a choice of $z = 1$ also increments k_1 by 1 while $z = 0$ does not.

We refer to VE with the elementary operations defined as above as *redundant representation VE (RR-VE)*.

Theorem 1. *RR-VE is correct, i.e., it arrives at the identical solution as VE using the full tabular representation of intermediate functions.*

Sketch of proof. The full proof is given in the extended version of the paper (Robbel, Oliehoek, and Kochenderfer 2015). Intuitively, the modified AUGMENT and REDUCE operations ensure that, *for consistent entries*, the factors they produce are correct, since (for those consistent entries) they will only query consistent entries of the input factors to produce the result. As such, we can show that VE will never access inconsistent entries. Since there are no other modifications to the regular VE algorithm, RR-VE is correct. □

5 Exploiting Anonymity in the ALP

The results for RR-VE can be exploited in the ALP solution method that was introduced in Section 2. The non-linear max constraint in Equation 6 is defined over functions $c_i \triangleq \gamma g_i - h_i \; \forall h_i \in H$ and reward factors $R_j, j = 1, \ldots, r$, which are all locally-scoped and together span a factor graph. As outlined previously, a VE procedure over this factor graph can translate the non-linear constraint into a set of linear constraints that is reduced compared to the standard formulation of the ALP.

The key insight of this section is that for a class of factored (M)MDPs defined with count aggregator functions in the 2TBN the same intuition about reduced representations as in the previous section applies to implement the non-linear max constraint even more compactly.

We first establish that basis functions $h_i \in H$, when back-projected through the 2TBN (which now includes mixed-mode functions), retain correct basis back-projections g_i with reduced representations. The basis back-projection is computed with summation and product operations only (Equation 4). We have previously shown that summation (AUGMENT) of mixed-mode functions is correct for its consistent entries. The same result holds for multiplication when replacing the sum operation with a multiplication. It follows that g_i (and c_i) share the compact reduced representations derived in Section 3 and that they are correctly defined on their consistent entries.

The exact implementation of the max constraint in Equation 6 with RR-VE proceeds as for the regular VE case. All correctness results for RR-VE apply during the computation of the constraints in the RR-ALP. The number of variables and constraints is exponential only in the *size of the representation* of the largest MMF formed during RR-VE. Further, the representation with the smaller set of constraints is exact and yields the identical value function solution as the ALP that does not exploit anonymous influence.

6 Experimental Evaluation

We evaluate our methods on undirected disease propagation graphs with 30 and 50 nodes. For the first round of experiments, we contrast runtimes of the normal VE/ALP method (where possible) with those that exploit "anonymous influence". We then consider a disease control problem with 25 agents in a densely connected 50-node graph that cannot be solved with the normal ALP. Problems of this size

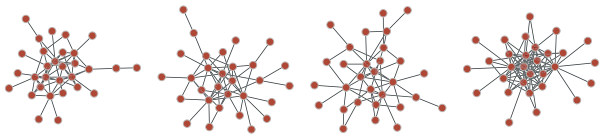

Figure 1: Sample of three random graphs in the test set with 30 nodes and a maximum out-degree of 10. Rightmost: test graph with increased out-degree sampled from $[1, 20]$.

$\lvert\mathbf{C1}\rvert$, VE, ALP	$\lvert\mathbf{C}_{RR}\rvert$, RR-VE,-ALP	$\frac{\lvert\mathbf{C}_{RR}\rvert}{\lvert\mathbf{C1}\rvert}$	RR-VE / VE	RR-ALP / ALP
131475, 6.2s, **1085.8s**	94023, 1.5s, **25.37s**	0.72	0.24	**0.02**
24595, 1.1s, 3.59s	12515, 0.17s, 1.2s	0.51	0.15	0.33
55145, 3.5s, 30.43s	27309, 0.4s, 8.63s	0.5	0.11	0.28
74735, 3.0s, 115.83s	41711, 0.69s, 12.49s	0.56	0.23	0.11
71067, 4.16s, 57.1s	23619, 0.36s, 8.86s	0.33	0.08	0.16
24615, **1.6s**, 1.15s	**4539**, **0.07s**, 0.35s	**0.18**	**0.04**	0.30
63307, 2.2s, 141.44s	34523, 0.39s, 4.03s	0.55	0.18	0.03
57113, 0.91s, **123.16s**	40497, 0.49s, **2.68s**	0.71	0.54	**0.02**
28755, 0.54s, 17.16	24819, 0.36s, 3.86s	0.86	0.67	0.22
100465, 2.47s, 284.75s	38229, 0.62s, 36.76s	0.38	0.25	0.13
	Average relative size:	0.53	0.25	0.16

Table 1: Constraint set sizes, VE and ALP solution times for normal (column 1) and methods exploiting anonymous influence (column 2). The last three columns show their relative magnitudes. Maximal reductions are shown in bold.

($|\mathcal{S}| = 2^{50}$, $|\mathcal{A}| = 2^{25}$) are prohibitively large for exact solution methods to apply and are commonly solved heuristically. To assess quality of the RR-ALP solution, we evaluate its policy performance against a vaccination heuristic.

In all experiments, we use indicator functions I_{X_i}, $I_{\bar{X}_i}$ on each state variable (covering the two valid instantiations $\{\text{healthy}, \text{infected}\}$) as the basis set H in the (RR-)ALP. We use identical transmission and node recovery rates throughout the graph, $\beta = 0.6$, $\delta = 0.3$. Action costs are set to $\lambda_1 = 1$ and infection costs to $\lambda_2 = 50$. All experiments use the identical greedy elimination heuristic for both VE and RR-VE, which minimizes the scope size of intermediate terms at the next iteration.

Runtime Comparison We use graph-tool (Peixoto 2014) to generate 10 random graphs with an out-degree k sampled from $P(k) \propto 1/k$, $k \in [1, 10]$. Out-degrees per node thus vary in $[1, 10]$; the mean out-degree in the graphs in the test set ranges from 2.8 (graph 1) to 4.2 (graph 10). Figure 1 illustrates a subset of the resulting networks.

The runtime results comparing the VE/ALP method to RR-VE/RR-ALP are summarized in Table 1. Shown are the number of constraints for each method, the wall-clock times for VE to generate the constraints, and the ALP runtimes to solve the value function after the constraints have been computed. The last three columns show the relative magnitude of each measure, i.e. the gains in efficiency of the methods exploiting anonymous influence in each of the 10 random graphs. On average, the RR-ALP solution time reduces to 16% of the original ALP runtime while maintaining the identical solution. Reductions by a factor of 50 are observed for two of the random graphs in the set (corresponding to the

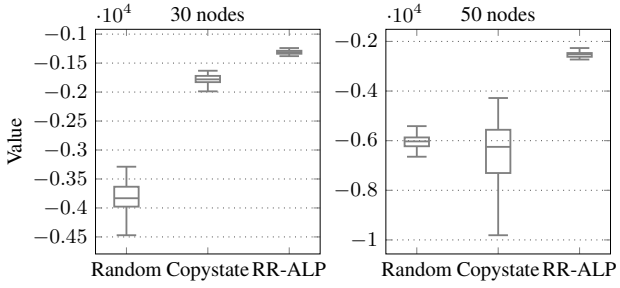

Figure 2: Statistics of the mean returns of the three policies for the disease control problems. Mean returns are computed over 50 randomly sampled starting states after 200 steps of policy simulation (each mean return is computed over 50 independent runs from a given s_0). Visualized in the box plots are median, interquartile range (IQR), and $\pm 1.5 \cdot IQR$ (upper and lower whiskers) of the mean returns.

highlighted entries in the last column).

We performed a final experiment with a graph with a larger out-degree (k sampled from the interval $[1, 20]$, shown at the right of Figure 1). The disease propagation problem over this graph *cannot be solved* with the normal VE/ALP because of exponential blow-up of intermediate terms. The version exploiting anonymous influence completes successfully, performing constraint computation using RR-VE in 124.7s and generating $|\mathbf{C}_{RR}| = 5816731$ constraints.

Policy Performance In this section we show results of policy simulation for three distinct policies in the disease control task over two random graphs (30 nodes with 15 agents and 50 nodes with 25 agents, both with a maximum out-degree per node set to 15 neighbors). The disease control problem over both graphs is infeasible for the regular VE/ALP due to the exponential increase of intermediate factors during VE. We compare the solution of our RR-ALP method to a random policy and a heuristic policy that applies a vaccination action at X_i if X_i is infected in the current state and is controlled. The heuristic is reactive and referred to as the "copystate" heuristic in our evaluation. It serves as our main comparison metric for these large and densely connected graphs where optimal solutions are not available.

To evaluate the policy performance, we compute the mean returns from 50 randomly sampled starting states s_0 after 200 steps of policy simulation (each mean return is computed over 50 independent runs from a given s_0). Figure 2 shows statistics of these mean returns and provides an indication of the sensitivity to the initial conditions in the disease graph.

The "copystate" heuristic works reasonably well in the 30-node/15-agent problem (left-hand side of Figure 2) but is consistently outperformed by the RR-ALP solution which can administer anticipatory vaccinations. This effect actually becomes more pronounced with *fewer* agents: we experimented with 6 agents in the identical graph and the results (not shown) indicate that the "copystate" heuristic performs significantly worse than the random policy. This is presumably because blocking out disease paths early becomes more important with fewer agents since the lack of agents in other regions of the graph cannot make up for omissions later.

In the 50-node/25-agent scenario the reactive "copystate" heuristic does not provide a statistically significant improvement over a random policy (right-hand side of Figure 2). It is outperformed by the RR-ALP solution by roughly a factor of 3 in our experiments. In the same figure it is also apparent that the performance of the heuristic depends heavily on the initial state of the disease graph.

7 Related Work

Many recent algorithms tackle domains with large (structured) state spaces. For exact planning in factored domains, SPUDD exploits a decision diagram-based representation (Hoey et al. 1999). Monte Carlo tree search (MCTS) has been a popular online approximate planning method to scale to large domains (Silver, Sutton, and Müller 2008). These methods do not apply to exponential action spaces without further approximations. Ho et al. (2015), for example, evaluated MCTS with three agents for a targeted version of the disease control problem. Recent variants that exploit factorization (Amato and Oliehoek 2015) may be applicable.

Our work is based on earlier contributions of Guestrin (2003) on exploiting factored value functions to scale to large factored action spaces. Similar assumptions can be exploited by inference-based approaches to planning which have been introduced for MASs where policies are represented as finite state controllers (Kumar, Zilberstein, and Toussaint 2011). There are no assumptions about the policy in our approach. The variational framework of Cheng et al. (2013) uses belief propagation (BP) and is exponential in the cluster size of the graph. A more detailed comparison with (approximate) loopy BP is future work.

Generalized counts in first-order (FO) models eliminate indistiguishable variables in the same predicate in a single operation (Sanner and Boutilier 2009; Milch et al. 2008). Our contributions are distinct from FO methods. Anonymous influence applies in propositional models and to node sets that are not necessarily indistiguishable in the problem. We also show that shattering into disjoint counter scopes is not required during VE and show how this results in efficiency gains during VE.

There is a conceptual link to approaches that exploit anonymity or influence-based abstraction in decentralized or partially-observable frameworks. Oliehoek, Witwicki, and Kaelbling (2012) define influence-based policy abstraction for factored Dec-POMDPs, which formalizes how different *policies* of other agents may lead to the same influence. Algorithms for search in this influence space have been presented for the subclass of TD-POMDPs (Witwicki, Oliehoek, and Kaelbling 2012). Their use in our problem, however, would require imposing decentralization constraints (i.e., restrictions on what state factors agents can base their actions on) for MMDPs. We provide a more scalable approach for MMDPs by introducing a practical way of dealing with aggregation operators.

Also closely related is the work by Varakantham, Adulyasak, and Jaillet (2014) on exploiting agent anonymity

in transitions and rewards in a subclass of Dec-MDPs with specific algorithms to solve them. Our definition of anonymity extends to both action and state variables; our results on compact, redundant representation of anonymous influence further also applies outside of planning (e.g., for efficient variable elimination).

8 Conclusions and Future Work

This paper introduces the concept of "anonymous influence" in large factored multiagent MDPs and shows how it can be exploited to scale variable elimination and approximate linear programming beyond what has been previously solvable. The key idea is that both representational and computational benefits follow from reasoning about influence of variable sets rather than variable identity in the factor graph. These results hold for both single and multiagent factored MDPs and are exact reductions, yielding the identical result to the normal VE/ALP, while greatly extending the class of graphs that can be solved. Potential future directions include approximate methods (such as loopy BP) in the factor graph to scale the ALP to even larger problems and to support increased basis function coverage in more complex graphs.

Acknowledgments

F.O. is supported by NWO Innovational Research Incentives Scheme Veni #639.021.336. We thank the anonymous reviewers for helpful suggestions.

References

Amato, C., and Oliehoek, F. A. 2015. Scalable planning and learning for multiagent POMDPs. In *AAAI Conference on Artificial Intelligence (AAAI)*, 1995–2002.

Bailey, N. T. J. 1957. *The Mathematical Theory of Epidemics*. London: C. Griffin & Co.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11:1–94.

Cheng, Q.; Liu, Q.; Chen, F.; and Ihler, A. 2013. Variational planning for graph-based MDPs. In *Advances in Neural Information Processing Systems (NIPS)*, 2976–2984.

Cornelius, S. P.; Kath, W. L.; and Motter, A. E. 2013. Realistic control of network dynamics. *Nature Commun.* 4(1942):1–9.

Cui, H.; Khardon, R.; Fern, A.; and Tadepalli, P. 2015. Factored MCTS for large scale stochastic planning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 3261–3267.

Guestrin, C.; Koller, D.; Parr, R.; and Venkataraman, S. 2003. Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research* 19:399–468.

Guestrin, C.; Koller, D.; and Parr, R. 2002. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems (NIPS)*, 1523–1530.

Guestrin, C. 2003. *Planning Under Uncertainty in Complex Structured Environments*. Ph.D. Dissertation, Computer Science Department, Stanford University.

Ho, C.; Kochenderfer, M. J.; Mehta, V.; and Caceres, R. S. 2015. Control of epidemics on graphs. In *IEEE Conference on Decision and Control (CDC)*.

Hoey, J.; St-Aubin, R.; Hu, A. J.; and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

Kochenderfer, M. J. 2015. *Decision Making Under Uncertainty: Theory and Application*. MIT Press.

Kok, J. R., and Vlassis, N. A. 2006. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research* 7:1789–1828.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Koller, D., and Parr, R. 1999. Computing factored value functions for policies in structured MDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1332–1339.

Kumar, A.; Zilberstein, S.; and Toussaint, M. 2011. Scalable multiagent planning using probabilistic inference. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2140–2146.

Liu, Y.-Y.; Slotine, J.-J.; and Barabasi, A.-L. 2011. Controllability of complex networks. *Nature* 473(7346):167–173.

Milch, B.; Zettlemoyer, L. S.; Kersting, K.; Haimes, M.; and Kaelbling, L. P. 2008. Lifted probabilistic inference with counting formulas. In *AAAI Conference on Artificial Intelligence (AAAI)*, 1062–1068.

Nowzari, C.; Preciado, V. M.; and Pappas, G. J. 2015. Analysis and control of epidemics: A survey of spreading processes on complex networks. Technical Report arXiv:1505.00768.

Oliehoek, F. A.; Witwicki, S.; and Kaelbling, L. P. 2012. Influence-based abstraction for multiagent systems. In *AAAI Conference on Artificial Intelligence (AAAI)*, 1422–1428.

Peixoto, T. P. 2014. The graph-tool python library. *figshare*. DOI: 10.6084/m9.figshare.1164194.

Puterman, M. L. 2005. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley.

Raghavan, A.; Joshi, S.; Fern, A.; Tadepalli, P.; and Khardon, R. 2012. Planning in factored action spaces with symbolic dynamic programming. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Robbel, P.; Oliehoek, F. A.; and Kochenderfer, M. J. 2015. Exploiting anonymity in approximate linear programming: Scaling to large multiagent MDPs (extended version). *ArXiv e-prints* arXiv:1511.09080.

Sanner, S., and Boutilier, C. 2009. Practical solution techniques for first-order MDPs. *Artificial Intelligence* 173(5-6):748–788.

Silver, D.; Sutton, R. S.; and Müller, M. 2008. Sample-based learning and search with permanent and transient memories. In *International Conference on Machine Learning (ICML)*, 968–975.

Taghipour, N.; Fierens, D.; Davis, J.; and Blockeel, H. 2013. Lifted variable elimination: Decoupling the operators from the constraint language. *Journal of Artificial Intelligence Research* 47:393–439.

Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2014. Decentralized stochastic planning with anonymity in interactions. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2505–2512.

Witwicki, S.; Oliehoek, F. A.; and Kaelbling, L. P. 2012. Heuristic search of multiagent influence space. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 973–981.