Decentralized Bayesian Reinforcement Learning for Online Agent Collaboration

W. T. L. Teacy[1], G. Chalkiadakis[2]
A. Farinelli[3]
[1]University of Southampton, UK
{wtlt,acr,nrj}@ecs.soton.ac.uk
[2]Technical University of Crete, Greece
gehalk@intelligence.tuc.gr

A. Rogers[1], N. R. Jennings[1]
S. McClean[4], G. Parr[4]
[3]University of Verona, Italy
alessandro.farinelli@univr.it
[4]University of Ulster, UK
{si.mcclean,gp.parr}@ulster.ac.uk

ABSTRACT

Solving complex but structured problems in a decentralized manner via multiagent collaboration has received much attention in recent years. This is natural, as on one hand, multiagent systems usually possess a structure that determines the allowable interactions among the agents; and on the other hand, the single most pressing need in a cooperative multiagent system is to coordinate the local policies of autonomous agents with restricted capabilities to serve a system-wide goal. The presence of uncertainty makes this even more challenging, as the agents face the additional need to learn the unknown environment parameters while forming (and following) local policies in an online fashion. In this paper, we provide the first Bayesian reinforcement learning (BRL) approach for distributed coordination and learning in a cooperative multiagent system by devising two solutions to this type of problem. More specifically, we show how the Value of Perfect Information (VPI) can be used to perform efficient decentralised exploration in both model-based and model-free BRL, and in the latter case, provide a closed form solution for VPI, correcting a decade old result by Dearden, Friedman and Russell. To evaluate these solutions, we present experimental results comparing their relative merits, and demonstrate empirically that both solutions outperform an existing multiagent learning method, representative of the state-of-the-art.

Categories and Subject Descriptors

I.2.6 [**Learning**]; I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms

Keywords

multiagent learning, Bayesian techniques, uncertainty

1. INTRODUCTION

In cooperative multiagent systems, the grand challenge is to ensure that a common, system-wide goal is achieved by coordinating the actions of individual agents. Often, however, this is difficult because each agent (1) only has a limited world view, and (2) has no direct control over the actions of its peers. Agents are therefore restricted to forming local policies subject to local information. In such cases, the goal is to coordinate the agents' local polices to form a joint optimal one—a problem that is, in general, computationally infeasible [2]. However, multiagent systems usually possess some form of structure, which can often be exploited to perform efficient coordination. For example, in Distributed Constraint Optimisation Problems (DCOPs), an agent's actions are only dependent on a subset of its peers—a fact that can be used to construct efficient coordination algorithms for the agents as a whole [9].

Indeed, solving such complex but structured problems is challenging, particularly in the context of reinforcement learning (RL) [18], in which agents must *explore* their environment to learn how best to act. However, existing collaborative reinforcement learning techniques [11, 13] are "point-based", i.e. they do not optimize decisions w.r.t. all possible world models. For this reason, they provide a suboptimal solution to the *exploration-exploitation* problem [18], in which agents must decide when to explore actions of uncertain value, which may yet prove to be optimal.

This is particularly important for problems involving real hardware, e.g. Unmanned Aerial Vehicles (UAVs), for two reasons: (1) repeated trials may be expensive and time consuming, and so control software must learn effectively from few interactions with the environment; and (2) exploratory actions may result in damage or injury, and so one must account for respective risks and value.

Here, we address this by making the following three key contributions to the state-of-the-art: (1) in order to provide a near optimal solution to the exploration-exploitation trade-off, we present the *first* model-free and model-based algorithms for decentralised Bayesian Reinforcement Learning (BRL) in a cooperative multiagent system; (2) by empirical analysis, we show that both algorithms *outperform* an existing state-of-the-art decentralised learning method, while at the same time provide different complementary trade-offs between computational complexity, and the amount of exploration required to learn an effective coordination strategy; (3) as part of our model-free method, we provide *a closed-form solution for the Value of Perfect Information (VPI)*, which we use to perform efficient exploration. The latter corrects a key result by Dearden, Friedman and Russell, central to their original contribution on Bayesian Q-learning [7].

In the rest of the paper, Sections 2 and 3 outline related work; Section 4 presents our closed-form solution for VPI; Section 5 describes how this can be used for decentralised BRL; Section 6 evaluates these algorithms empirically; and Section 7 concludes.

2. DECENTRALIZED COORDINATION

A decentralized coordination problem is one in which a set of agents must together choose how to act so that their joint utility is maximized. Here, we outline a solution to this class of problems, based on the use of the max-sum algorithm. Specifically, given a factored utility function $F(\mathbf{a}) = \sum_i F_i(\mathbf{a}_i)$, the goal of decentralized co-

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

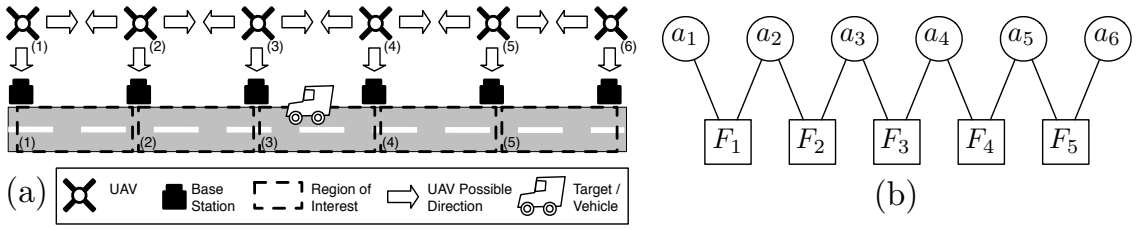

Fig. 1 (a) Six UAVs patrol a road for vehicles. Each has a base station where it can land during idle periods, and is responsible for patrolling its adjacent regions, east and west along the road. When a vehicle is detected in a region (e.g. by ground based motion sensors), the pair of UAVs bordering the region are alerted, and must both patrol the region simultaneously to observe the vehicle. **(b)** The corresponding factor graph with factors represented by squares, and actions by circles.

ordination is to find the joint action vector, $\boldsymbol{a}$, that maximizes the global utility function:

$$\arg\max_{\mathbf{a}} \sum_i F_i(\mathbf{a}_i) \tag{1}$$

Here, each $F_i(\mathbf{a}_i)$ represents a local utility function (*a factor*), and $\mathbf{a}_i \subseteq \mathbf{a}$ are *local action vectors*.[1] This is efficiently solved by the *max-sum* algorithm; a technique of the Generalized Distributive Law (GDL) [1], widely used for computing factored functions using local message passing [19].

In particular, the factored optimization problem described in Eq. 1 can be viewed as a DCOP and represented by a bipartite factor graph [14]. For example, the scenario illustrated in Fig. 1(a) can be represented by the bipartite factor graph in Fig. 1(b). Specifically, this represents the factored *reward* function $F = \sum_{i=1}^{5} F_i(a_i, a_{i+1})$, where each factor node, F_i, represents the local reward for observing a given region of road, the action nodes represent the decision variables for each UAV, and edges connect factors to the actions on which they depend. The max-sum algorithm operates on the factor graph by iterative message passing between neighbouring variable and factor nodes. When the graph is cycle-free, the messages are guaranteed to converge and the global optimal solution is computed. While no such guarantees exist for cyclic graphs, extensive empirical evidence demonstrates that good solutions can still be reached [9].

Although the max-sum on its own can be applied in a variety of settings, the coordination problem as defined above only deals with cases in which agents must choose a single joint action to receive a single immediate reward. However, in sequential decision making, agents must also consider their action's effect on the future world state, which in turn influences their future rewards. In detail, consider an agent's decision-making problem in a stochastic environment modeled as a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, \Pr, R \rangle$, with finite state and action sets $\mathcal{S}, \mathcal{A}$, transition dynamics $\Pr$, and reward function $R : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, where $\Pr(s'|s, a)$ is the probability of reaching state s' after taking action a at s. Similarly, $R(s, a)$ denotes the expected reward which is obtained when action a is performed at state s. The agent then needs to construct an optimal policy, $\pi : \mathcal{S} \mapsto \mathcal{A}$, derived by solving a system of Bellman equations [18]:

$$Q(s, a) = R(s, a) + \gamma \sum_{s' \in \mathcal{S}} Pr(s'|s, a) Q(s', \pi(s)) \tag{2}$$

Here, γ is a discount factor that places more weight on immediate rewards; the *Q-value* function, $Q : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, is equal to the expected total sum of future discounted rewards given the current state and action; and from this, the optimal policy is $\pi(s) = \arg\max_a Q(s, a)$. This standard formulation was originally extended to multiagent MDPs by defining $\mathcal{A}$ as the cartesian product of individual action spaces associated with each agent [3]. Thus, $\boldsymbol{a} \in \mathcal{A}$ is a joint action vector comprising individual agents' actions. Note that actions are chosen in the context of a state, which is represented here as a vector $\boldsymbol{s} \in \mathcal{S}$. That is, $\boldsymbol{s}$ comprises global state variables shared by all agents, as in a direct extension of the classic factored state representation [4], but may also contain state variables specific to individual agents, such as a UAV's battery life.

In principle, such multiagent MDPs may be solved like any other MDP, except that by taking the cartesian product of individual action spaces, the computational complexity is exponential in the number of agents. Fortunately, in many coordination problems (e.g. Fig. 1), the actions of each agent are only strongly dependent on a subset of their peers. In such cases, good approximations to the optimal policy can often be found by representing the problem as a *factored* MDP [10]. Specifically, these can be defined by assuming a factored structure for the Q-value function, $Q(\boldsymbol{s}, \boldsymbol{a}) = \sum_i Q_i(\boldsymbol{s}_i, \boldsymbol{a}_i)$, where each factor, Q_i, depends only on a subset of states, $\boldsymbol{s}_i \subseteq \boldsymbol{s}$, and actions $\boldsymbol{a}_i \subseteq \boldsymbol{a}$. Using this assumption, algorithms for solving factored MDPs can make efficiency savings similar to those made by max-sum. For example, [10] presents a dynamic programming solver for factored MDPs, which can converge in approximately logarithmic time, despite the state-action space growing exponentially.

While, in general, assuming a factorisation of the Q-value function might result in suboptimal solutions, a trade-off between complexity and optimality can be found by decomposing the problem in different ways. For example, at one extreme, associating a factor with each agent favours computational efficiency by achieving a large degree of factorisation. In contrast, we guarantee an optimal solution when no factorisation is performed, but the complexity of finding this solution is now exponential in the total number of state and action variables.

Here, however, we adopt a more graded approach, by decomposing coordination problems into *regional* subproblems. Specifically, we associate each factor with a *region*, which can represent any part of the overall decision problem that depends only on a subset of agents. For example, in Fig. 1, each region represents part of the road, in which observations are only made by neighbouring UAVs. However, we could equally decrease the amount of decomposition by aggregating adjacent regions. In principle, this would enable better polices at the expense of more computation, because each factor would now account for dependencies between the actions of a larger number agents [10].

3. BAYESIAN RL

In standard Reinforcement Learning (RL), a single agent is faced with a decision problem, typically modelled as an MDP with *unknown* reward and transition dynamics. Due to this additional uncertainty, the agent cannot simply solve the MDP to find the best policy, but must instead *learn* it by *exploring* different states and ac-

[1] We abuse notation here slightly by applying set operators to vectors. The intended interpretation is that a vector's elements form a set, related to another vector's elements.

tions. To achieve this, classical RL techniques require some form of heuristic to encourage learning, by exploring actions with unknown outcomes. Generally, however, these heuristics are based on intuition alone, and so lack theoretical foundations based on any notion of optimality.

In contrast, Bayesian RL (BRL) methods [5] formulate the problem as a *belief-state MDP*, in which an agent's beliefs are explicitly modelled as part of the state. In this way, an agent can *infer* the exploratory value of an action, by reasoning about how the information obtained by performing an action may enable better future decisions. The solution to the belief-state MDP provides the optimal solution to the action selection problem, taking full account of the informative value of exploratory actions. Therefore, BRL *does not* require the use of an explicit exploration heuristic; instead, agents need only act greedily w.r.t. the Bayesian Q-values to achieve optimal learning. No other method outperforms the Bayesian one in expectation, when using the same prior information [15].

Unfortunately, solving the belief-state MDP is, in general, computationally infeasible. Nevertheless, the Bayesian approach does provide a theoretical framework from which we can construct and evaluate practical near-optimal solutions. In particular, *model-based* BRL approaches work by explicitly modelling the belief-state MDP; while *model-free* approaches, such as *Bayesian Q-learning*, attempt to learn action values directly, without solving an MDP. The following subsections discuss each of these in detail.

3.1 Model-Based BRL

In general, *model-based* BRL methods work by maintaining a density P over all possible dynamics D and reward functions R, which is updated with each observed tuple, $\langle s, a, r, s' \rangle$, where s' is the next state after action a is performed at s and reward r is received. This density describes the agent's belief state regarding the world, and is used to choose appropriate actions, given the current state of knowledge. Typically, updates are rendered tractable by assuming a convenient conjugate prior [8], which allows the belief state to be represented using a small set of *hyperparameters*, updated using a set of simple closed form equations.

For example, [6] models rewards and states as multinomial random variables, such that, for each s and a, a parameter set $\{\theta^{s'}_{s,a} | s' \in \mathcal{S}, \theta^{s'}_{s,a} = Pr(s'|s,a)\}$ defines the distribution of s' given s and a. In the same way, a similar set, $\{\theta^{r}_{s,a}\}$, models the conditional distribution over possible rewards. However, since these parameters are themselves unknown, each is assigned a conjugate prior, in this case a *Dirichlet*, specified by hyperparameters $\{\alpha^{s'}_{s,a}\}$ and $\{\alpha^{r}_{s,a}\}$. For example, if we observe a specific tuple $\langle s, a, r, s' \rangle$, the corresponding $\alpha^{s'}_{s,a}$ and $\alpha^{r}_{s,a}$ are both incremented by 1. In particular, if all hyperparameters are initialised to 1, this results in uniform densities, which become peaked around the true parameter values as more evidence is observed. In this way, the Dirichlets capture both the relative likelihood of possible multinomials, and the amount of uncertainty given the evidence. Given this, [6] proposes a tractable approximate solution to the belief-state MDP based on a myopic estimation of the expected Value of Perfect Information (VPI), defined by the expected *gain* in reward received, if the agent learns the *true* value of choosing action a in state s:

Definition 1 (VPI) *Let a_1 be an agent's current best action in state s, with expected Q-value m_{s,a_1}, and a_2 its 2nd best action, with expected Q-value m_{s,a_2}. According to [7], the* gain *for learning the true expected Q-value of an action, a, is then*

$$Gain_{s,a}(\mu_{s,a}) = \begin{cases} m_{s,a_2} - \mu_{s,a} & \text{if } a = a_1 \wedge \mu_{s,a} < m_{s,a_2}, \\ \mu_{s,a} - m_{s,a_1} & \text{if } a \neq a_1 \wedge \mu_{s,a} > m_{s,a_1}, \\ 0 & \text{otherwise.} \end{cases}$$

where $\mu_{s,a} = Q(s,a)$ is the true Q-value for a in s. Based on this, the value of perfect information (VPI) *for selecting a in s is defined as $VPI(s,a) = E[Gain_{s,a}(\mu_{s,a})]$.*

Intuitively, the gain reflects the effect on decision quality of learning the true $Q(s,a)$. In the first two cases, what is learned results in a change of decision: either because the estimated optimal action is found to be worse than predicted, or because some other action is found to be optimal. Otherwise, the information is irrelevant, since no change in decision is induced. Based on this, [6] proposes that an agent should choose actions that maximise $E[Q(s,a)] + VPI(s,a)$. In this way, exploration is encouraged by VPI when an agent is uncertain about its estimates, but the resulting policy approaches the optimal w.r.t the true Q-value, as VPI decreases in light of accumulated evidence.

3.2 Bayesian Q-Learning

The main problem with model-based BRL methods is that solving a belief-state MDP is generally intractable, and even approximate solutions can scale poorly in large problems. For example, in model-based BRL, VPI cannot be calculated analytically, but instead must be estimated by solving multiple MDPs sampled from an agent's belief state [6]. Fortunately, model-free techniques offer a simpler alternative, in which an agent directly learns the value for choosing an action in a given state, without explicitly solving an MDP. While this requires an agent to explore more to learn the true value of its actions (since the implications of observed evidence cannot be fully determined without modelling the MDP), the computational complexity of choosing an action is greatly reduced, which may be an important advantage in some on-line decision making problems.

In particular, standard Q-learning works by directly maintaining a point estimate of the Q-value, $Q(s,a)$, for each state and action, updated w.r.t. observed rewards. Unfortunately, it is *not* clear from this single estimate how much an agent should explore actions that are believed to be suboptimal, but may yet prove to be optimal.

In Bayesian Q-learning [7], this limitation is addressed by maintaining *a probability distribution* over $Q(s,a)$, which measures the uncertainty in the current estimate that can be used to guide exploration. More specifically, for each state-action pair, the total discounted reward is assumed to be normally distributed with unknown mean, $\mu_{s,a}$, and precision,[2] $\tau_{s,a} = 1/\sigma^2_{s,a}$, where $\sigma^2_{s,a}$ is the unknown variance of the distribution. Since $Q(s,a)$ is defined as the expected total discounted reward, we have $Q(s,a) = \mu_{s,a}$.

Now, to model the uncertainty in their estimates, Dearden et al. adopt the standard Bayesian approach of using conjugate parameter distributions for each pair of latent parameters, $(\mu_{s,a}, \tau_{s,a})$ [7, 8]. In this case, the joint distribution of $\mu_{s,a}$ and $\tau_{s,a}$ for each state-action pair is assumed to be a *normal-gamma* (*NG*) distribution, which is conjugate for normal densities with unknown mean and precision. More formally, we say that $(\mu_{s,a}, \tau_{s,a}) \sim NG(m_{s,a}, \lambda_{s,a}, \alpha_{s,a}, \beta_{s,a})$, where $\rho_{s,a} = \langle m_{s,a}, \lambda_{s,a}, \alpha_{s,a}, \beta_{s,a} \rangle$ are hyperparameters, updated according to the equations[3] in Theorem 1, which produce densities of the following form [8]:

$$p(\mu, \tau) \propto \tau^{\frac{1}{2}} e^{-\frac{1}{2}\lambda\tau(\mu-m)^2} \tau^{\alpha-1} e^{-\beta\tau} \quad (3)$$

Note that, in [7], the last term is *incorrectly stated* as $e^{\beta\tau}$, and so has the wrong sign within the exponent. Of course, we could always define a new hyperparameter $\hat{\beta} = -\beta$, and substitute this

[2] Here, the precision is used in place of the variance, because it simplifies the later Bayesian Analysis [8].

[3] In Bayesian Q-Learning, these update equations cannot be used directly because, although the latent distribution is over total discounted rewards, only *immediate* rewards can be directly observed. However, this technical detail [7] is not relevant to our discussion.

for β to correct the equation. However, in this case, the hyperparameter updates as stated in [7, 8] (Theorem 1) would also have to change, so in this sense, [7] is inconsistent.[4]

Theorem 1 (Posterior Hyperparameters) *Suppose that the prior density for the unknown parameters of a normal distribution is $p(\mu, \tau) = NG(m, \lambda, \alpha, \beta)$, and let $D = \{x_k\}_{k=1}^n$ be a set of n i.i.d. observations drawn from this distribution, with sample mean, $\bar{x} = \frac{1}{n}\sum_{k=1}^n x_k$, and sum of squares, $s^2 = \sum_{k=1}^n (x_k - \bar{x})^2$. As stated in [7, 8], the posterior is thus $p(\mu, \tau) \sim NG(m', \lambda', \alpha', \beta')$, with hyperparameters $\lambda' = \lambda + n$, $m' = (\lambda m + n\bar{x})/\lambda'$, $\alpha' = \alpha + n/2$ and $\beta' = \beta + s^2/2 + n\lambda(\bar{x} - m)^2/(2\lambda')$.*

From these *NG* distributions, a good estimate of $Q(s, a)$ can be obtained from $E[Q(s, a)] = E[\mu_{s,a}] = m_{s,a}$. However, in addition to such estimates, the parameter distributions also provide a representation of uncertainty. In particular, the marginal posterior distribution of $\mu_{s,a}$ generally becomes more peaked around its true value as more rewards are observed, and so the width of the distribution gives an indication of uncertainty. This can be used to guide principled exploration in Bayesian Reinforcement learning in a number of ways, the most promising of which is VPI action selection [7]. This works in the same way as described in Sec. 3 for model-based BRL, except that VPI can now be computed efficiently using a closed-form equation, without the need to sample and solve multiple MDPs. Unfortunately, the closed-form solution provided in [7] is inconsistent with the definition of VPI, and thus cannot be correct. In the next section, we highlight these inconsistencies in detail, and provide the *correct* analytical solution.

4. ANALYTICAL VPI FOR Q-LEARNING

As part of the Bayesian Q-Learning approach discussed above, [7] provides an *(incorrect)* analytical solution for the VPI, under the assumption that each $(\mu_{s,a}, \tau_{s,a})$ has a normal-gamma density with hyperparameters $m_{s,a}, \lambda_{s,a}, \alpha_{s,a}, \beta_{s,a}$. This result formed a critical part of the contribution of this paper, since without it, VPI would have to be calculated using numerical integration techniques, such as Monte Carlo sampling, thus introducing a significant computational overhead. In this section, we address this problem by (1) proving beyond doubt that the original equations are incorrect, and (2) providing the *correct* solution, which we prove in the appendix. With this in mind, we begin by restating the original result presented in [7], which we quote using the identity $E[\mu_{s,a}] = m_{s,a}$ for all s and a:

Proposition 1 (Dearden's Solution) *$VPI(s, a)$ is equal to $c + (m_{s,a_2} - m_{s,a_1})Pr(\mu_{s,a_1} < m_{s,a_2})$ when $a = a_1$, and $c + (m_{s,a} - m_{s,a_1}) \cdot Pr(\mu_{s,a} > m_{s,a_1})$ when $a \neq a_1$, where*

$$c = \frac{\Gamma\left(\alpha_{s,a} + \frac{1}{2}\right)\sqrt{\beta_{s,a}}}{\left(\alpha_{s,a} - \frac{1}{2}\right)\Gamma\left(\alpha_{s,a}\right)\Gamma\left(\frac{1}{2}\right)\sqrt{2\lambda_{s,a}}}\left(1 + \frac{m_{s,a}^2}{2\alpha_{s,a}}\right)^{-\alpha_{s,a}+\frac{1}{2}}$$

Here, the main problem is that c should *not* be constant w.r.t. m_{s,a_1} and m_{s,a_2}, but instead should depend on the difference between these two values and $m_{s,a}$. The following Lemma and Theorems show why this leads to inconsistent results.

Lemma 1 (Asymptotic Behaviour) *When $\alpha_{s,a} > \frac{1}{2}$, and $|m_{s,a}| \to \infty$, the term c from Proposition 1 goes to 0.*

PROOF. *If $\alpha_{s,a} > 1/2$ then $1/2 - \alpha_{s,a} < 0$. Therefore, since $\lim_{|m_{s,a}|\to\infty}(1 + m_{s,a}^2/2\alpha_{s,a}) = \infty$, $\lim_{|m_{s,a}|\to\infty} c = 0$.* □

[4]Theorem 1 is stated slightly differently in [7] and [8]. However, both are equivalent, differing only by some trivial transformations. Here, we follow the original reference [8] more closely.

Theorem 2 (Sensitivity to Value Changes) *If $d \in \mathbb{R}$, is added to all $\mu_{s,y}$ and $m_{s,y}$ for each action y, then this will change VPI according to Proposition 1. However, this is inconsistent with Definition 1, in which VPI is defined to be invariant to such changes.*

PROOF. *By Definition 1, if we add a constant, d, to $\mu_{s,a}$, m_{s,a_1}, and m_{s,a_2}, then $Gain_{s,a}(\mu_{s,a})$ and hence $VPI(s, a)$ remain unchanged. However, in Proposition 1, the term c depends only on $m_{s,a}$, and so it is sensitive to the addition of d, to which Definition 1 is invariant. In fact, in the limit $|d| \to \infty$ when $\alpha_{s,a} > 1/2$, $m_{s,a}$ also approaches ∞, and so from Lemma 1, c goes to zero.* □

Theorem 3 (Negative VPI) *By Proposition 1, VPI can be negative. However, this is inconsistent with Definition 1, in which VPI is strictly non-negative.*

PROOF. *Let $f(x, y) = (E[x] - y)\,Pr(x > y)$. By Definition 1, $Gain_{s,a}(\mu_{s,a}) \geq 0$, $\therefore VPI(s, a) = E[Gain_{s,a}(\mu_{s,a})] > 0$. In contrast, if $a \neq a_1 \wedge m_{s,a} < m_{s,a_1}$ then $f(\mu_{s,a}, m_{s,a_1})$ will be negative for non-zero $Pr(\mu_{s,a} > m_{s,a_1})$. However, if we add $d \in \mathbb{R}$, to $\mu_{s,a}$, $m_{s,a}$ and m_{s,a_1}, then $f(\mu_{s,a}, m_{s,a_1})$ will remain constant, while from Lemma 1, $c \to 0$ when $|d| \to \infty$. Therefore, according to Proposition 1, $VPI(s, a)$ will be negative for sufficiently large d, which is thus inconsistent with Definition 1. A similar argument can also be made when $a = a_1$.* □

From Theorems 2 and 3 it is clear that Proposition 1 cannot be true. As we shall show, however, the *correct* solution can be obtained by replacing Dearden et al.'s constant c with a *truncation bias* function, which we now define:

Definition 2 (Truncation Bias Function) *For hyperparameters $\rho = \langle m, \lambda, \alpha, \beta\rangle$, we define the* truncation bias function, *$\mathcal{B}_\rho : \mathbb{R} \to \mathbb{R}$, as follows.*

$$\mathcal{B}_\rho(x) = \frac{\Gamma\left(\alpha - \frac{1}{2}\right)\sqrt{\beta}\left(1 + \frac{\lambda(x-m)^2}{2\beta}\right)^{-\alpha+\frac{1}{2}}}{\Gamma(\alpha)\Gamma(1/2)\sqrt{2\lambda}}$$

Given this definition, the correct closed-form solution for VPI in Bayesian Q-Learning is given by Theorem 4:

Theorem 4 (VPI Solution) *According to an agent's beliefs, let a_1 be its current best action in state s, with expected reward m_{s,a_1}; and a_2 is its second best action, with expected reward m_{s,a_2}. Similarly, let a be an action whose reward in s is normally distributed, with unknown parameters $\langle \mu, \tau\rangle \sim NG(m, \lambda, \alpha, \beta)$, and hyperparameters $\rho = \langle m, \lambda, \alpha, \beta\rangle$. The VPI for choosing a in s is then $VPI(s, a) =$*

$$\begin{cases} (m_{s,a_2} - m) \cdot Pr(\mu|\mu < m_{s,a_2}) + \mathcal{B}_\rho(m_{s,a_2}) & \text{for } a = a_1 \\ (m - m_{s,a_1}) \cdot Pr(\mu|\mu > m_{s,a_1}) + \mathcal{B}_\rho(m_{s,a_1}) & \text{otherwise.} \end{cases}$$

As mentioned, this result is proved in the appendix, thus showing that it can be used to calculate VPI in Bayesian Q-learning, without the computational expense of numerical integration. In particular, we now introduce a general approach for decentralised BRL, including an efficient model-free algorithm based on this result.

5. DECENTRALIZED BAYESIAN RL

Sec. 2 described how certain multiagent MDPs can be decomposed into a set of regional reward and transition functions, and showed how this can be used to generate tractable solutions that approximate the optimal policy. However, this still assumes that the reward and transition functions are *known*, which is not the case in RL problems. As discussed in Sec. 3, the Bayesian RL approach deals with such cases by constructing and solving a belief state MDP, and so explicitly handles uncertainty over dynamics.

To put this in a multiagent MDP context, we define $\boldsymbol{b}$ as the joint belief state of all the agents, corresponding to some probability distribution over all possible models. More formally, $\boldsymbol{b}$ has the form $\boldsymbol{b} = \langle P_M; \boldsymbol{s} \rangle$, where P_M is some density over possible models (i.e., transition and reward dynamics); and $\boldsymbol{s}$ is the current state of the system (a vector of state variables). Given experience $\langle \boldsymbol{s}; \boldsymbol{a}; r; \boldsymbol{s}' \rangle$, where r is the observed global reward, $\boldsymbol{b}$ can be updated to $\boldsymbol{b}' = \boldsymbol{b}(\langle \boldsymbol{s}; \boldsymbol{a}; r; \boldsymbol{s}' \rangle) = \langle P'_M; \boldsymbol{s}' \rangle$ with updates given by Bayes rule (and implemented using standard Bayesian methods):

$$P'_M(m) = zPr(\boldsymbol{s}'; r|\boldsymbol{s}; \boldsymbol{a}; m)P_M(m) \tag{4}$$

where z is a normalising constant. Notice that the states and actions in the formulation above are *global* states and *joint* agent actions. However, by adopting a decomposition similar to that described in Sec. 2, we can formulate the problem based on local beliefs, in a way that facilitates tractable solutions. Specifically, we achieve this by making the following three assumptions. First, we assume that the global reward, r, can be factored into regional rewards, r_i, such that $r = \sum_i r_i$ over all regions, i. Second, we assume that the global belief state is decomposed into local beliefs states of the form $\boldsymbol{b}_i = \langle P_{M_i}; \boldsymbol{s}_i \rangle$, for each region i. These are updated as before, except that only local states, actions and rewards are observed:

$$\boldsymbol{b}'_i = \boldsymbol{b}_i(\langle \boldsymbol{s}_i; \boldsymbol{a}_i; r_i; \boldsymbol{s}'_i \rangle) = \langle P'_{M_i}; \boldsymbol{s}'_i \rangle$$

Finally, based on these two assumptions, we assume that the Q-value function can be factored as before, such that $Q(\boldsymbol{a}, \boldsymbol{b}) = \sum_i Q_i(\boldsymbol{a}_i, \boldsymbol{b}_i)$. While the addition of the first two assumptions may seem more restrictive than the factored MDP formulation in Sec. 2, the alternative is to assume global visibility of the full global state, joint actions and rewards. In fact, this is a more unrealistic assumption in coordination problems involving large numbers of distributed agents, so the assumptions above only make explicit what is already true in realistic settings.

Moreover, as we now show, these assumptions enable tractable coordinated reinforcement learning, in a way that explicitly accounts for uncertainty in the agents local beliefs, and so provide a near-optimal solution to the exploitation-exploration problem [18]. To achieve this, we propose a multiagent Bayesian RL method based on VPI, which consists of the following two steps. First, since (in general) no single agent has a complete view of the global problem, there is no global belief state from which to calculate VPI. Instead, within each region, the agents evaluate the informative value of performing a given local action w.r.t. the local belief state. Second, the agents coordinate their actions via message passing, in order to maximise the sum of all the regional expected Q-values and VPI. In this way, the agents not only coordinate their actions in a way that exploits the sum of their existing knowledge, but also *explore* joint actions that are informative for the regional belief states.

In detail, suppose that, given i's current belief state, the expected value of joint regional action $\boldsymbol{a}_i$ is given by $\overline{Q}_i(\boldsymbol{a}_i, \boldsymbol{s}_i)$. Letting $\boldsymbol{a}^1$ denote the regional action with highest expected q-value at $\boldsymbol{s}_i$ and $\boldsymbol{a}^2$ the second-highest, the regional VPI is defined as the *gain*, denoted $Gain_{\boldsymbol{a}_i, \boldsymbol{s}_i}(Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i))$, from learning that the *true* Q_i value of taking $\boldsymbol{a}_i$ at $\boldsymbol{s}_i$ is in fact q:

$$Gain_{\boldsymbol{a}_i, \boldsymbol{s}_i}(q) = \begin{cases} \overline{Q}_i(\boldsymbol{a}^2, \boldsymbol{s}_i) - q, & \text{if } \boldsymbol{a}_i = \boldsymbol{a}^1 \wedge q < \overline{Q}_i(\boldsymbol{a}^2, \boldsymbol{s}_i) \\ q - \overline{Q}_i(\boldsymbol{a}^1, \boldsymbol{s}_i), & \text{if } \boldsymbol{a}_i \neq \boldsymbol{a}^1 \wedge q > \overline{Q}_i(\boldsymbol{a}^1, \boldsymbol{s}_i) \\ 0, & \text{otherwise} \end{cases}$$

The regional $VPI(\boldsymbol{a}_i, \boldsymbol{s}_i)$, defined as $E[Gain_{\boldsymbol{a}_i, \boldsymbol{s}_i}(Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i))]$, is thus a direct analog of the standard notion of VPI, and can be added to the corresponding expected regional Q-value to boost the desirability of local actions with uncertain value. Thus, the *regional* value for taking the local joint action $\boldsymbol{a}_i$ in $\boldsymbol{s}_i$ can be defined as $\overline{Q}_i(\boldsymbol{a}_i, \boldsymbol{s}_i) + VPI(\boldsymbol{a}_i, \boldsymbol{s}_i)$. To use this definition for coordinated multiagent learning, we now propose two decentralised methods for BRL: (1) model-based decentralised BRL, which uses a distributed sampling approach to approximate the solution to the belief state MDP, and (2) decentralised Bayesian Q-learning, a model-free approach, which side-steps the computational complexity of solving the belief-state MDP, by learning the regional Q-values directly.

5.1 Decentralised Model-based BRL

As in standard model-based BRL (Sec. 3), model-based decentralised BRL works by sampling multiple MDPs, and using the solution to the MDPs to approximate the expected Q-value function, and the associated VPI. Specifically, we propose a four-step procedure:

1. For each region i, an agent representing i maintains a density, P_{M_i}, over all possible local transition and reward dynamics, updated using local observations only. For example, as used in our experiments (Sec. 6) this may be achieved by (1) modelling local state transition probabilities, $Pr(\boldsymbol{s}'_i|\boldsymbol{s}_i, \boldsymbol{a}_i)$, as a set of unknown multinomial distributions with associated Dirichlet priors; and (2) modelling local reward distributions, $Pr(r_i|\boldsymbol{s}_i, \boldsymbol{a}_i)$, as unknown Gaussians with associated normal-gamma priors.[5]

2. In each region, the representative agent samples a finite set of z local (reward and transition) models from the corresponding density, P_{M_i}; that is, z samples for every $i \in [1, Y]$ are specified ($z * Y$ in total). These are used to form a set of z distinct factored MDPs, such that the kth factored MDP comprises the kth local reward and transition functions sampled from each of the Y regions. Each of these factored MDPs represents one possible instance of the joint decision problem, which are solved to produce the set of local $Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i)$ values, for the corresponding joint optimal policy. In this paper, we achieve this using our own decentralised dynamic programming algorithm (not described here) based on max-sum. However, this choice does not significantly change the end result, and so may be replaced by any suitable algorithm for factored MDPs (e.g. [10]).

3. For each region, we calculate the average $Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i)$ from the z sampled MDPs, and use this to approximate $\overline{Q}_i(\boldsymbol{a}_i, \boldsymbol{s}_i)$. Similarly, we compute $Gain_{\boldsymbol{a}_i, \boldsymbol{s}_i}(Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i))$ for each of the z MDPs w.r.t. i, and approximate $VPI(\boldsymbol{a}_i, \boldsymbol{s}_i)$ by their average.

4. The local value of $\boldsymbol{a}_i$ (for region i) is defined to be $\overline{Q}_i(\boldsymbol{a}_i, \boldsymbol{s}_i) + VPI(\boldsymbol{a}_i, \boldsymbol{s}_i)$; $\boldsymbol{a}_i$'s desirability is thus boosted by its expected VPI. When the agents come to act, these are then evaluated w.r.t. *current* state to form the factors of a factor graph that can be operated on by the standard max-sum algorithm. The max-sum output at each variable node (one per agent j) gives the action choice for j. As a consequence, each agent's decision is informed by the entire global state through its affect on local rewards.

Although this procedure does not guarantee an optimal solution to the generally intractable decision problem, it does provide a practical alternative that maintains several useful features of the theoretical optimum. In particular, the look-ahead performed by solving the factored MDPs takes into account the likely impact of each agent's current actions on the future rewards obtained by the system as a whole. Moreover, by employing VPI, we explicitly account for

[5]This differs from [6], in which rewards are assumed to be multinomial rather than normally distributed. However, both approaches are valid, the first being appropriate when rewards are drawn from a known finite set, while the latter allows for any real value.

the exploratory value of each joint action for obtaining relevant information about regional rewards. In this way, agents will only explore joint actions that are likely to produce beneficial gains in their future rewards. Equally important, however, is the scalability of the procedure, which it achieves through the use of max-sum and factored MDPs (Sec. 2).

5.2 Decentralised Model-Free BRL

Although the above procedure scales well to problems involving large numbers of agents, sampling and solving multiple factored MDPs may still present a significant overhead when computational resources are at a premium, such as in sensor networks or UAVs with embedded CPUs. As we have already seen however, this problem can be side-stepped using model-free methods that attempt to learn the Q-value functions directly. With this in mind, we now adapt Bayesian Q-learning for decentralised settings, by modifying the model-based procedure above in the following way.

1. Rather than maintain a density over all possible transition and reward dynamics, each P_{M_i} now becomes a normal-gamma (*NG*) density, which *directly* models the distribution over all possible regional Q-value functions. This density is maintained and updated using the same procedures proposed in [7], except we now maintain separate models for each regional Q-value, rather than a single global one.
2. Rather than approximate the regional value functions by sampling, $E[Q_i(\boldsymbol{a}_i, \boldsymbol{s}_i)]$ is given directly by the mean of the corresponding *NG* distribution, and $VPI(\boldsymbol{a}_i, \boldsymbol{s}_i)$ can be calculated analytically using our closed-form solution in Sec. 4. As before, by summing these two values together for each region, we obtain a factor graph that can be operated on directly by max-sum to coordinate the agents' actions w.r.t. the current global state.

This procedure is similar to the decentralised Q-learning algorithm in [13], except that by adopting a Bayesian approach, we perform more efficient exploration of the state-action space. This ability is demonstrated empirically in the next section, using the algorithm in [13] as a benchmark.

6. EMPIRICAL EVALUATION

We now evaluate our proposed decentralised BRL methods by simulating the scenario in Fig. 1. Here, two factors may influence performance: (1) the priors required by each algorithm, and (2) the complexity of the task being learnt.

To investigate the former, we ran multiple simulations using priors with varying hyperparameter values. In particular, as suggested in Sec. 5.1, we used Dirichlet priors to model the regional state transition probabilities used by our model-based algorithm, along with normal-gamma priors for the regional rewards. Similarly, for our decentralised Bayesian Q-learning algorithm, we used normal-gamma priors to directly model the regional Q-value distributions, thus avoiding the need for separate transition and reward models.

To investigate the latter, we simulated two variants of the Fig. 1 scenario. In the first variant, we simulated the scenario exactly as described in Fig. 1, with 6 UAVs bordering on 5 regions with 1 target. Specifically, each UAV has a base station where it can land during idle periods, and is responsible for patrolling regions adjacent to its base station, east and west along the road. When a vehicle is detected in a region (e.g. by ground based motion sensors), the pair of UAVs bordering the region are alerted, and, importantly, must *both* patrol the region *simultaneously* to observe the vehicle. Since UAVs 1 and 6 border only one region each, their actions are limited to remaining idle and patrolling east or west respectively. All other UAVs can patrol both east and west, or remain idle. A region incurs a cost of -1 if one of its UAVs is active (regardless of direction), -2 if both are active, and receives a reward of 30 after every 3 observations.[6] Each region only communicates directly with its immediate neighbouring regions, and is only aware of its two bordering UAVs' actions. The number of target observations is visible to all regions, but its location is known only to its current region, and those immediately east and west. In the second variant, the state-action space complexity is reduced by decreasing the number of regions to 3, while at the same, the lookahead required is increased, by changing the number of observations required to receive a positive reward of 30 from 3 observations to 4.

The combined size of the local state and action spaces in the first variant is thus 54 for regions *a* and *b*, and 108 elsewhere; compared to 4860 for the global problem. This illustrates the reductive power of decomposition to simplify the combinatorial problem faced by the agents, thus turning a potentially intractable problem into a solvable one. Despite this simplification, we can still learn an effective policy for the global problem, as we now demonstrate.

In each scenario variant, we benchmark against two other strategies: (1) *random*, which selects actions with equal likelihood, and thus represents a basic solution that any algorithm should outperform; and (2) *Kok*, the decentralized Q-learning policy proposed in [13] (named after the lead author). The latter maintains separate estimates for each regional Q-value, and is the only other algorithm in the literature that uses max-sum decentralised reinforcement learning. However, unlike our model-free method, this maintains point estimates of the Q-values only, and uses ϵ-greedy exploration with a fixed exploration probability of $\epsilon = 0.2$ (see [13] for details).

Fig. 2 plots the mean cumulative rewards at each timestep of these experiments, calculated using $\approx$ 100 independent runs per control condition for statistical significance. As we discuss below, the most interesting effect induced by the choice of prior can be observed when the normal-gamma λ hyperparameter is varied, while all other hyperparameters remain constant. For this reason, we focus on λ in this discussion. In particular, Fig. 2(a) and (b), show the results for our model-based algorithm (labelled *MB*) in the 3 and 5 region problems respectively, when the λ hyperparameter of the prior distribution over rewards was varied in the range $[10^{-2}, 10^{-5}]$. In each case, a uniform Dirichlet prior was used for the state transition probabilities, while the other hyperparameters for the rewards where initialised to $\mu = 0$, $\alpha = 1$, $\beta = 1$. There are two main results of these experiments.

First, for all hyperparameter values tested, our model-based approach outperforms both the random strategy, and the *Kok* algorithm. This is because our model-based performs targeted exploration early on, taking account of its initial uncertainty. In contrast, although *Kok* learns quickly to prefer idle states that cost nothing (allowing it to dominate at the beginning) it takes significantly longer to learn that positive rewards can be achieved by coordinated observations of the target. In fact, in the harder 5 region problem (Fig. 2 b), *Kok* fails to learn how to observe the target at all within 4000 timesteps, a result which is backed by experiments in [13], which required >10,000 episodes to learn in a similar setting. Thus, although our model-based approach incurs an initial cost by performing early exploration, this enables the UAVs to learn how to coordinate their observations in significantly fewer timesteps than *Kok*.[7]

[6] Here, by requiring UAVs to perform multiple observations, we are able to evaluate our algorithms' ability to learn non-myopic policies. The UAVs must learn to balance the immediate cost of observing the target, against the expected gain in future reward.

[7] Although the learning time may seem large, the no. states & actions is equally large, and strategies start with uninformative priors.

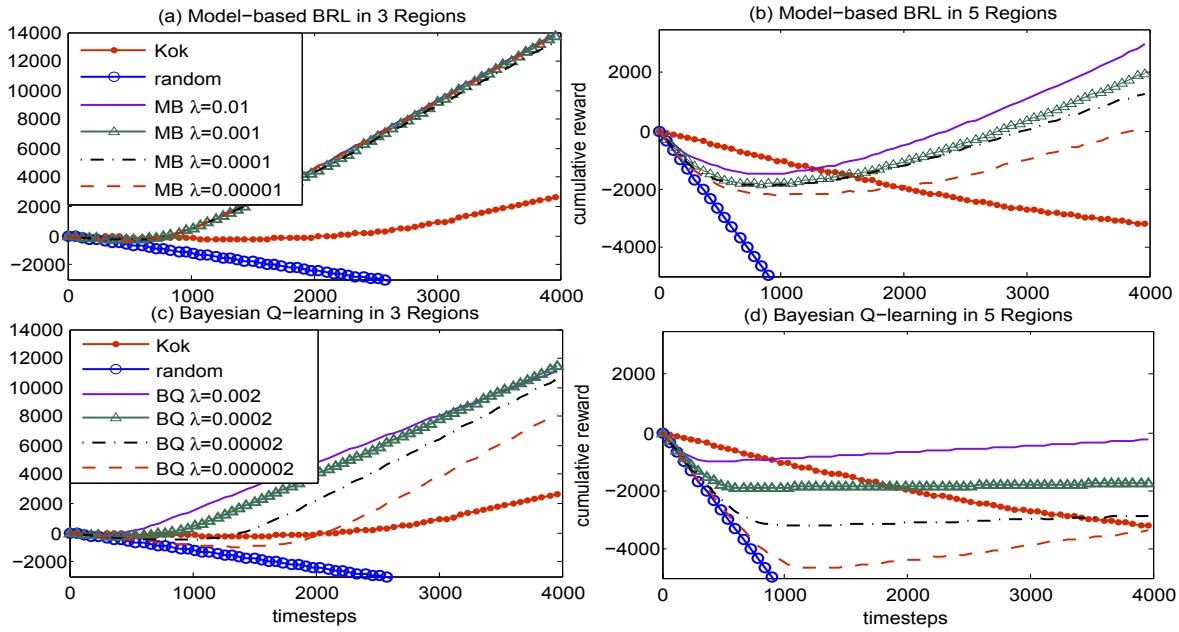

Figure 2: UAV Surveillance Scenario Results

Second, although our model-based algorithm performs well generally, changing the prior can have a significant effect on performance, for example, when the λ hyperparameter of the reward's NG prior is varied in the 5-region problem (Fig. 2 b).[8] In general, this is to be expected, because strong prior information will always bias inference in a certain way. Nevertheless, at first sight, the results here are somewhat surprising, since in these experiments, we use supposedly uninformative priors, which should be quickly dominated by observed evidence. Closer inspection provides two reasons for this result. First, since the state-action space is relatively large, the prior's effect can persist over parts of the MDP, because of the time required to fully explore all state-action pairs. Second, the range of rewards considered probable can significantly effect the amount of exploration performed. In particular, although changes in $\lambda \ll 1$ have little impact on posterior NG distributions, *a priori*, each decrease in λ by a factor of 10 produces an equivalent increase in the range of probable rewards. As a result, agents become more optimistic about the value of potential rewards, and so are incentivised to explore otherwise suboptimal policies, on the chance that they may (even occasionally) return very high rewards. While this dependence on priors may seem like a disadvantage, it should be noted that non-Bayesian approaches usually rely on tuning parameters with less obvious interpretations. In contrast, prior distributions do have an intuitive interpretation, and in most domains, the range of likely rewards is known *a priori*.

As shown in Fig. 2(c) and (d), our model-free approach achieves similar results, except for a greater dependence on the correct choice of lambda.[9] This is because, without explicitly modelling the underlying MDP, it cannot infer the full consequences of its observations, and so requires more exploration to rule out occasionally high rewards from the full range of policies. As such, although the model-based algorithm has a higher computational complexity (see below), it can learn effectively from less evidence. This may be particularly advantageous in robotics, where the cost of performing actions with real hardware may outweigh the additional computational overhead. In this sense, even our model-free approach significantly outperforms *Kok*, making it a useful compromise in domains requiring both computational and learning efficiency.

In terms of time complexity, it is true that our methods take longer to choose actions compared to the simpler *Kok* approach. For example, using our implementation (which leaves significant room for optimization), it took the model-based learner on average 11 ± 6 secs. to choose each action, compared to 0.2 ± 0.1 secs for the Bayesian Q-Learner, and 0.04 ± 0.02 secs. for the *Kok* approach. However, notice that our approaches outperform other methods in terms of the *timesteps* required to learn. This is important in many real-world domains that use real hardware (e.g. UAVs), where repeated interactions with the environment may be time consuming, costly, or potentially dangerous. In such domains, it makes sense to deliberate over each action for longer to save time and resources in the long run; as fewer, as opposed to more, interactions for learning are strongly preferred. BRL is ideally suited to this in general; and by exploiting regional decomposition, our approach can address otherwise intractable coordination problems.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the *first* approach for performing cooperative multiagent BRL; and provide the *correct* closed form equations for VPI in Bayesian Q-Learning [7] — a crucial result with implications *beyond* our decentralised setting. Key to our approach is the use of factored MDPs, which significantly reduce complexity in structured coordination problems. In this sense, our experiments are somewhat preliminary, since factored MDPs can be applied to problems larger than those attempted here [10]. Nevertheless, our results still demonstrate the potential of BRL to outperform existing multiagent learning algorithms, and so, in future work, we plan to evaluate our approach in larger problems, by taking advantage of advances in related areas, such as Monte-Carlo Planning [17] and ND-POMDPs [16].

Typically, informative priors significantly reduce learning times.

[8]This and all other claims made here are verified by t-tests with a confidence level of at least 95%.

[9]In the model-free experiments λ was scaled by 0.2 due to the change from modelling immediate rewards to Q-values.

8. REFERENCES

[1] S. M. Aji and R. J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, 2000.

[2] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes. In *Proc. of UAI-2000*, pages 32–37, 2000.

[3] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proc. of IJCAI-99*, pages 478–485, 1999.

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

[5] G. Chalkiadakis and C. Boutilier. Sequentially optimal repeated coalition formation under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 24(3):441–484, 2012.

[6] R. Dearden, N. Friedman, and D. Andre. Model based bayesian exploration. In *Proc. of UAI'99*, 1999.

[7] R. Dearden, N. Friedman, and S. Russell. Bayesian Q-Learning. In *Proc. of AAAI-98*, 1998.

[8] M. DeGroot and M. Schervish. *Probability & Statistics*. Pearson Education, 3rd edition, 2002.

[9] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *Proc. of AAMAS 2008*, pages 639–646, 2008.

[10] C. Guestrin, D. Koller, and R. Parr. Max-norm projections for factored mdps. In *Proc. of AAAI-01*, pages 673–680, 2001.

[11] C. Guestrin, M. Lagoudakis, and R. Parr. Coordinated reinforcement learning. In *Proc. of ICML-02*, pages 227–234, 2002.

[12] H. J. Kim. Moments of truncated student-t distribution. *Journal of the Korean Statistical Society*, 37:81–87, 2008.

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

[14] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. on Information Theory*, 42(2):498–519, 2001.

[15] J. Martin. *Bayesian decision problems and Markov chains*. Wiley, 1967.

[16] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In *Proceedings of AAAI-05*, pages 133–139. 2005.

[17] D. Silver and J. Veness. Monte-carlo planning in large pomdps. In *Neural Information Processing Systems 23*, pages 2164–2172. 2010.

[18] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[19] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. on Information Theory*, 47(2):723–735, 2001.

Acknowledgments

This research was funded by the UK Engineering and Physical Sciences Research Council (EPSRC) as part of the SUAAVE and ORCHID projects (grant references EP/F06358X/1 and EP/I011587/1). Georgios Chalkiadakis has been partially supported by the European Commission FP7-ICT Cognitive Systems, Interaction, and Robotics under the contract #270180 (NOPTILUS).

APPENDIX

We now prove that Theorem 4 provides the *correct* solution for VPI in Bayesian Q-Learning, which [7] states incorrectly. We start with the case when $a = a_1$, and for simplicity, drop the subscripts for the hyperparameters of a, so that $\mu = \mu_{s,a}$ and so on. Then from Definition 1, we have $VPI(s,a) = E[m_{s,a_2} - \mu]$ given $\mu < m_{s,a_2}$, and 0 otherwise. However, since the truth of $\mu < m_{s,a_2}$ is unknown, we must marginalise to derive the correct expectation:

$$VPI(s,a) = (m_{s,a_2} - E[\mu|\mu < m_{s,a_2}])\, Pr(\mu < m_{s,a_2})$$

Similarly, when $a \neq a_1$, we find that

$$VPI(s,a) = (E[\mu|\mu > m_{s,a_1}] - m_{s,a_1})\, Pr(\mu > m_{s,a_1})$$

Therefore, to prove Theorem 4, we need only show that

$$\begin{aligned} \forall x \in \mathbb{R} \quad E[\mu|\mu < x]Pr(\mu < x) &= m \cdot Pr(\mu < x) - \mathcal{B}_\rho(x) \\ \wedge \quad E[\mu|\mu > x]Pr(\mu > x) &= m \cdot Pr(\mu > x) + \mathcal{B}_\rho(x) \end{aligned}$$

To achieve this, we first state some prerequisite results, which are then used to prove these equations in Lemma 5.

Lemma 2 *Let X and Y be continuous random variables such that the c.d.f. of X is $F(x)$ and $Y = \sigma X + \mu$. By substitution [8], $Pr(Y < y) = F\left[(y-\mu)/\sigma\right]$, where $x = (y-\mu)/\sigma$, and so $Pr(Y > y) = 1 - F\left[(y-\mu)/\sigma\right]$.*

Lemma 3 *Suppose that X is t-distributed with v degrees of freedom, and $F_v(\cdot)$ its c.d.f. From [12], when $X \in (a,b)$ is given, its expected value is*

$$E[X|a < X < b] = \frac{\Gamma\left(\frac{v-1}{2}\right) v^{v/2} \left(A_{(v)}^{-(v-1)/2} - B_{(v)}^{-(v-1)/2}\right)}{2\left[F_v(b) - F_v(a)\right]\Gamma(v/2)\Gamma(1/2)}$$

for $v > 1$, where $A_{(v)} = v + a^2$ and $B_{(v)} = v + b^2$.

Corollary 1 *By taking the limits $a \to -\infty$ and $b \to \infty$ respectively, when $v > 1$, it follows from Lemma 3 that*

$$\begin{aligned} E[X|X < b] &= -\frac{\Gamma\left(\frac{v-1}{2}\right) v^{v/2} \left(v + b^2\right)^{-(v-1)/2}}{2F_v(b)\Gamma(v/2)\Gamma(1/2)} \\ E[X|X > a] &= \frac{\Gamma\left(\frac{v-1}{2}\right) v^{v/2} \left(v + a^2\right)^{-(v-1)/2}}{2\left[1 - F_v(a)\right]\Gamma(v/2)\Gamma(1/2)} \end{aligned}$$

Lemma 4 *If $\langle \mu, \tau \rangle \sim NG(m, \lambda, \alpha, \beta)$ are the unknown parameters of a normal distribution, then $Z = (\mu - m)\sqrt{\lambda\alpha/\beta}$ is t-distributed with 2α degrees of freedom [8], from Lemmas 2 & 3, we thus have $Pr(\mu < y) = F_{2\alpha}\left[(y-m)\sqrt{\lambda\alpha/\beta}\right]$.*

Lemma 5 *If $\langle \mu, \tau \rangle \sim NG(m, \lambda, \alpha, \beta)$ are the unknown parameters of a normal p.d.f. and $\rho = \langle m, \lambda, \alpha, \beta \rangle$, then*

$$E[\mu|\mu < x]Pr(\mu < x) = m \cdot Pr(\mu < x) - \mathcal{B}_\rho(x) \quad (5)$$

$$E[\mu|\mu > x]Pr(\mu > x) = m \cdot Pr(\mu > x) + \mathcal{B}_\rho(x) \quad (6)$$

PROOF. *If $Z = (\mu - m)\sqrt{\lambda\alpha/\beta}$ and $y = (x-m)\sqrt{\lambda\alpha/\beta}$ then $Pr(\mu < x) = Pr(Z < y)$, $Pr(\mu > x) = Pr(Z > y)$, and*

$$E[\mu|\mu < x] = m + \sqrt{\beta/\lambda\alpha} \cdot E[Z|Z < y] \quad (7)$$

$$E[\mu|\mu > x] = m + \sqrt{\beta/\lambda\alpha} \cdot E[Z|Z > y] \quad (8)$$

From Lemma 4, Z is t-distributed with $v = 2\alpha$ degrees of freedom, and so $\forall y \in \mathbb{R}, 0 < Pr(Z < y) = F_v(y) < 1$. Thus, by substitution into Corollary 1 we have

$$E[Z|Z < y] = -\frac{\Gamma\left(\alpha - \frac{1}{2}\right)(2\alpha)^\alpha \left(2\alpha + \frac{\lambda\alpha(x-m)^2}{\beta}\right)^{-\alpha+\frac{1}{2}}}{2Pr(Z < y)\Gamma(\alpha)\Gamma(1/2)}$$

$$E[Z|Z < y] = -\frac{\Gamma\left(\alpha - \frac{1}{2}\right)\sqrt{\alpha}\left(1 + \frac{\lambda(x-m)^2}{2\beta}\right)^{-\alpha+\frac{1}{2}}}{\sqrt{2}Pr(Z < y)\Gamma(\alpha)\Gamma(1/2)}$$

$$E[Z|Z < y] = -\sqrt{\lambda\alpha/\beta} \cdot \mathcal{B}_\rho(x)/Pr(Z < y) \quad (9)$$

By following the same procedure for $\mu > x$, we obtain

$$E[Z|Z > y] = \sqrt{\lambda\alpha/\beta} \cdot \mathcal{B}_\rho(x)/Pr(Z > y) \quad (10)$$

By substituting Eqs. 9 & 10 into Eqs. 7 & 8, we obtain

$$E[\mu|\mu < x] = m - \mathcal{B}_\rho(x)/Pr(\mu < x) \quad (11)$$

$$E[\mu|\mu > x] = m + \mathcal{B}_\rho(x)/Pr(\mu > x) \quad (12)$$

From this, Eqs. 5 & 6 follow directly, thus proving the lemma, and proving Theorem 4 as a consequence. □