Evolving Policies for Multi-Reward Partially Observable Markov Decision Processes (MR-POMDPs)

Harold Soh
haroldsoh@imperial.ac.uk

Yiannis Demiris
y.demiris@imperial.ac.uk

Dept. of Electrical and Electronic Engineering
Imperial College London, South Kensington Campus
London SW7 2AZ, United Kingdom

ABSTRACT

Plans and decisions in many real-world scenarios are made under uncertainty and to satisfy multiple, possibly conflicting, objectives. In this work, we contribute the multi-reward partially-observable Markov decision process (MR-POMDP) as a general modelling framework. To solve MR-POMDPs, we present two hybrid (memetic) multi-objective evolutionary algorithms that generate non-dominated sets of policies (in the form of stochastic finite state controllers). Performance comparisons between the methods on multi-objective problems in robotics (with 2, 3 and 5 objectives), web-advertising (with 3, 4 and 5 objectives) and infectious disease control (with 3 objectives), revealed that memetic variants outperformed their original counterparts. We anticipate that the MR-POMDP along with multi-objective evolutionary solvers will prove useful in a variety of theoretical and real-world applications.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; G.1.6 [**Optimization**]: Stochastic programming

General Terms

Algorithms

1. INTRODUCTION

Plans and decisions in many real-world scenarios are made under uncertainty and to satisfy multiple, possibly conflicting, objectives. For example, consider an autonomous rover exploring Mars; it has to develop a plan that maximises the area surveyed while minimising power consumption and ensuring safety. Furthermore, the robot has to cope with noisy sensors and slipping wheels. This example is illustrative of the difficulties caused by multiple-objectives, incomplete information and ineffective actions, common across many domains from inventory management to health-care.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

For modelling uncertain environments, the partially observable Markov decision process (POMDP) has emerged as a popular solution framework; a POMDP models an agent interacting with a (stochastic) system where the underlying state cannot be directly perceived. Depending on the actions taken and the underlying state, the agent receives a reward (or penalty). In our Mars rover example, the state space is the possible robot configurations on the Mars landscape and a reward is given for each newly explored sector. But what about the other objectives of power and safety, particularly in the presence of far/high-risk sectors?

Despite its generality, a given POMDP is still limited to a single reward function, inconsistent with situations where multiple objectives have to be considered. When faced with multiple objectives, POMDP modellers either have to disregard objectives or combine the different objectives into a single reward function. This has undesirable effects; for example, a change in one cost/reward component can have a significant impact on the resulting optimal policy. Furthermore, using the POMDP formulation does not permit the study of trade-offs between different goals. These limitations provide the main motivations for this work.

First, we contribute a generalisation of the POMDP framework to multiple-objectives, i.e., the multi-reward partially-observable Markov decision process (MR-POMDP). The MR-POMDP is a convenient framework for modellers who no longer have to combine objectives into a single reward structure (a potentially difficult and sometimes infeasible procedure). The solution for a MR-POMDP is not a single policy but a set of "best" policies that maximise objectives to varying degrees: the Pareto policy set. For regular POMDPs, there are two main classes of solvers: value-iteration and policy-search. Because value-iteration is generally slow, we focus on searching the policy space of *stochastic* finite state controllers (FSCs) [21] which are compact, interpretable, graph-based policy representations. With MR-POMDPs, FSCs are associated with value vectors that can be compared using the domination relation. However, the landscape of FSCs is characterised by many locally-optimal policies (local minima) [5], necessitating the use of a meta-optimisation approach such as the evolutionary algorithm (EA).

As our second contribution, we explore the use of leading *multi-objective evolutionary algorithms* (MOEAs) which have been demonstrably effective on many benchmark and real-world problems. In particular, we use the popular non-dominated sorting genetic algorithm (NSGA2) [27] and the recently-developed multi-objective covariance matrix adaptation evolutionary strategy (MO-CMA-ES in the literature

or MCMA for short in our paper) [14]. Since MOEAs can be slow to converge in large search spaces, we augment both algorithms with local-search, resulting in two *hybrid* or *memetic* algorithms: NSGA2-LS and MCMA-LS. To speed up the policy search, we demonstrate that in the special case of weighted linear combination (WLC), a defined preference vector reduces a given MR-POMDP to the familiar POMDP model with a linearly-combined reward function, which can be solved efficiently using an gradient-based optimisation algorithm. Performance comparisons between the algorithms on multi-objective problems (robotic loading, web-advertising and anthrax detection) reveal that the memetic variants outperformed their original counterparts.

In the next section, we briefly review the basic notions and notation relevant to POMDPs and FSCs. In Section 3, we introduce the MR-POMDP and present ideas related to multi-objective optimisation of FSCs. The MOEAs along with NSGA2-LS and MCMA-LS are described in Section 4. Our experimental setup and results are given in Section 5. We conclude with final remarks, along with more speculative ideas on future research, in Section 6.

2. PRELIMINARIES

2.1 The POMDP Model

A POMDP models an environment where states are not directly visible to an agent [17]. Instead, the agent makes observations, from which it has to infer actions to take. Depending on the underlying state and the action, the agent receives feedback in the form of a scalar-valued reward. In this work, we focus on situations where the agent wishes to maximise the expected discounted cumulative reward over time.

Formally, a POMDP is a tuple $\langle S, A, Z, T, O, R, \gamma \rangle$ where S is the set of states, A is the set of possible actions and Z is the set of observations available to the agent. T is transition function $T(s, a, s') = P(s'|s, a)$ which gives the probability of moving to state s' from s given action a and O is the observation function $O(s, z) = P(z|s)$ i.e. the probability of observing z in state s. The reward function, $R : S \times A \times S \rightarrow \mathbb{R}$ models the reward for arriving in state s' after executing action a in state s. Finally, the discount factor, γ ($0 \leq \gamma \leq 1$) regulates how much future rewards are discounted.

Because observations only give partial information about the current state, an agent has to rely on the complete history of its observations and actions. Let us define a finite history as $h_t = \{a_o, z_1, \ldots, a_{t-1}, z_t\} \in H$ where a_t and z_t are the action and observation at time t respectively. A policy π maps elements of H to actions $a \in A$ (or a distribution of actions in the case of *stochastic* policies). In other words, policies tell an agent what to do based on what it has seen up to that point. Given a distribution $\mathbf{b}_0$ over the starting states, the expected value of a policy π is

$$E_\pi = \mathbf{E}\left(\sum_{t=0}^{\infty} \gamma^t R_t \mid \pi, \mathbf{b}_0\right) \quad (1)$$

where R_t is the reward received at time t. The optimal policy π^* is one that maximises this expectation.

Unfortunately, finding π^* exactly is not easy and turns out to be PSPACE complete for finite-horizon POMDPs [23] and undecidable for the infinite-horizon case [20]. As such, research has focussed on finding approximated solutions with two classes of solvers emerging over time: value-iteration methods (e.g. [28, 25]) and policy-search methods.

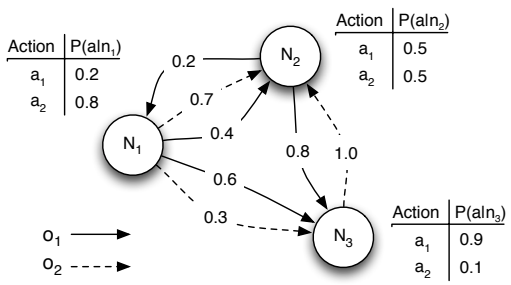


Figure 1: A FSC with three nodes $\{N_1, N_2, N_3\}$, two actions $\{a_1, a_2\}$ and two observations $\{o_1, o_2\}$. For each node, the action taken is governed by the probability distribution $\psi(n, a) = P(a|n)$. The transition from one node to another is dictated by another probability distribution $\eta(n', z, n) = P(n'|z, n)$. In this example, the probability of taking action a_1 in node N_1 is 0.2, after which a transition occurs depending on the observation received. If observation o_1 is received, a transition is made to N_2 with probability 0.4 or to N_3 with probability 0.6.

In this paper, we focus on policy-search for finite-state controllers (FSCs). Unlike value-iteration methods, policy-search algorithms work directly in the policy space and with a proper representation, good (locally optimal) policies can be found relatively quickly, even for systems with thousands of states. The computational properties of policy-search algorithms, such as gradient ascent[21], make them appealing, inducing a series of interesting developments in the past decade [26, 24, 5].

2.2 Finite State Controllers

A finite state controller (FSC) is a graph-based representation of a policy. Each node (also called a "memory state") dictates an action to take and depending on the observation received, we transition to another node in the graph (which defines the next action to be taken and so on). We work mainly with *stochastic* FSCs where each node defines a probability distribution over possible actions and nodes to transition to. As an example, a three-node FSC is shown in Fig. 1.

Assume S, A and Z to be finite. A stochastic FSC is a tuple $\langle \mathcal{N}, \psi, \eta \rangle$ where $\mathcal{N}$ is a set of nodes, $\psi(n, a) = P(a|n)$ is the action selection distribution for each node $n \in \mathcal{N}$ and $\eta(n', z, n) = P(n'|z, n)$ gives the probability of moving to node n' from node n after observing z. It can be shown that the cross-product between a POMDP and a FSC induces a finite Markov chain [21]. Following the derivations by Meuleau *et al.* [21], the transition matrix of this Markov chain is given by

$$\tilde{T}_\pi(\langle n, s \rangle, \langle n', s' \rangle) = \sum_{a \in A} \psi(n, a) T(s, a, s') \sum_{z \in Z} O(s', z) \eta(n, z, n') \quad (2)$$

and the expected immediate reward vector for each state

pair is

$$\tilde{C}_\pi(\langle n, s\rangle) = \sum_{a\in A} \psi(n,a) \sum_{s'\in S} T(s,a,s')R(s,a,s'). \quad (3)$$

The value of the FSC, $\tilde{\mathbf{V}}^\pi$, can then be found by solving the Bellman equation:

$$\tilde{\mathbf{V}}_\pi = \tilde{\mathbf{C}}_\pi + \gamma \tilde{\mathbf{T}}_\pi \tilde{\mathbf{V}}_\pi \quad (4)$$

and the policy value independent of the starting state is

$$\tilde{E}_\pi = \tilde{\mathbf{b}}_0 \tilde{\mathbf{V}}_\pi \quad (5)$$

where $\tilde{\mathbf{b}}_0$ is the joint probability distribution across the initial $\langle n, s\rangle$ pairs. The gradient of (5) with respect to each policy parameter is

$$\nabla \tilde{E}_\pi = \tilde{\mathbf{b}}_0 \nabla \tilde{\mathbf{V}}_\pi \quad (6)$$

With (5) and (6), a given FSC can be optimised using a numerical optimisation algorithm such as conjugate gradient. As we will see, these equations are similarly valid for MR-POMDPs, with multiple reward functions.

3. MULTI-REWARD POMDPS

In this section, we introduce a generalisation of POMDPs for modelling problems with multiple objectives. Recall that in the standard POMDP, there is a single reward function R. We replace this reward function by a vector of reward functions $\mathbf{R} = [R^{(1)}, R^{(2)}, \ldots, R^{(M)}]$ where each $R^{(i)}$ corresponds to the rewards obtained under objective i. The tuple $\langle S, A, Z, T, O, \mathbf{R}, \gamma\rangle$ is termed a multi-reward partially-observable Markov decision process (MR-POMDP) [1]. The closest related work is by Bryce [6] on conditional probabilistic planning, who introduced a multi-objective extension to the value function of belief-state MDPs. In contrast, we consider POMDPs with a vector of reward functions, which can then be solved using either value-iteration or policy-search methods. From another perspective, this work extends multi-objective variants of Markov decision processes (MDPs) [29, 1, 30] (which model systems with fully-observable states) to the partially-observable domain.

With MR-POMDPs, the agent receives a vector-valued (instead of a scalar) reward. Extending (1), we seek to maximise the expected cumulative discount reward for each objective:

$$\max \mathbf{E}\left(\sum_{t=0}^{\infty} \gamma^t R_t^{(i)} \mid \pi, \mathbf{b}_0\right) \text{ for } i = 1, 2, \ldots, M \quad (7)$$

where $R_t^{(i)}$ is the reward received at time t under reward function $R^{(i)}$ and M is the number of objectives.

3.1 Policy Domination and the Pareto Optimal Set

With multiple rewards, the value of a given policy is a vector $\mathbf{E}_\pi = [E_\pi^{(i)}]$ where $E_\pi^{(i)}$ gives the value of the policy under reward function $R^{(i)}$. To determine the optimal policy (or policies), we need to be able to compare policy value vectors: when is a policy π_k better than another policy π_l?

Intuitively, a policy is preferred over another if it possesses a higher value for at least one objective, and is no worse for all others. Formally, a policy π_k *dominates* policy π_l, denoted as $\pi_k \succ \pi_l$, if at least one of its value functions p is strictly better than that of policy l and none of its value functions are worse i.e. $E_{\pi_k}^{(i)} \geq E_{\pi_l}^{(i)}$ for all $i = 1, \ldots, M$ and there exists p such that $E_{\pi_k}^{(p)} > E_{\pi_l}^{(p)}$. In contrast, if $E_{\pi_k}^{(i)} \leq E_{\pi_l}^{(i)}$ for all i and there exists p such that $E_{\pi_k}^{(p)} < E_{\pi_l}^{(p)}$, then we say π_k is *dominated* by π_l, denoted $\pi_k \prec \pi_l$. Otherwise, π_k and policy π_l are *non-dominating*, $\pi_k \sim \pi_l$.

Given the above definitions, the best policies, $\pi_k^* \in \overline{\mathbf{P}}^*$, are those that are *not dominated* by any other policy; there does not exist π_l such that $\pi_l \succ \pi_k^*$. We call these policies *Pareto-optimal* and $\overline{\mathbf{P}}^*$ is the *Pareto optimal set*. The set of all value vectors for the policies in the Pareto optimal set is called the *Pareto optimal front*, $\overline{\mathbf{E}}^* = \{\mathbf{E}_{\pi*}\}$. Fig. 2 illustrates the concept of dominance and a sample Pareto optimal front. Given a MR-POMDP, our goal is to find the Pareto optimal set of policies. However, the set may be infinitely large and therefore, we seek a finite approximation.

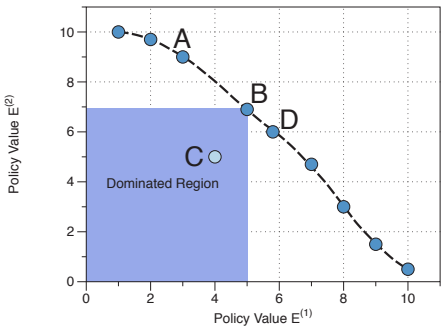


Figure 2: An illustration of dominance on a bi-objective problem. A, B and D are non-dominating solutions on the Pareto optimal front. C is dominated by B and D, but is *not dominated* by A.

3.2 Finite State Controllers for MR-POMDPs

FSCs can be evaluated for MR-POMDPs in a similar manner to POMDPs. The primary difference being that the value of the FSC under a MR-POMDP is a vector where each component $\tilde{E}_\pi^{(i)}$ is calculated using (5) with reward function $R^{(i)}$. Likewise, there are M gradient vectors $\nabla \tilde{E}_\pi^{(i)}$ for each objective.

One method of constructing the (approximated) Pareto optimal set is through repeated initialisations of a random policy and climbing the gradient using a multi-objective local-search algorithm [3, 12]. Unfortunately, each multi-objective gradient evaluation is expensive; our computational effort has increased due to the need to construct M different $\tilde{\mathbf{C}}_\pi$ vectors (Eq. 3). Next, we show that this effort can be reduced in the special case of linear weighted combination.

3.3 Linear Weighted Combination of Reward Functions

One class of methods for optimising multi-objective problems combine the different objective functions via a *scalarising function*, such as the weighted linear combination (WLC). Intuitively, a weight vector specifies a preference over the different objectives. Given a specified weight vector $\mathbf{w} = [w^{(i)}]$

[1]For simplicity, we assume the discount factor is equal across rewards.

where $\sum_{i=1}^{M} w^{(i)} = 1$, we define the weighted value of a policy,

$$\tilde{E}_{\pi}^{\mathbf{w}} = \sum_{i=1}^{M} w^{(i)} \tilde{E}_{\pi}^{(i)} \tag{8}$$

Given $\mathbf{w}$, we can efficiently optimise a FSC for a weighted MR-POMDP by constructing the combined reward function $R^{\mathbf{w}} = \sum_{i}^{M} w^{(i)} R^{(i)}$, i.e., from (8),

$$\begin{aligned} \tilde{E}_{\pi}^{\mathbf{w}} &= \sum_{i=1}^{M} w_i \tilde{\mathbf{b}}_0 \tilde{\mathbf{V}}_{\pi}^{(i)} \\ &= \tilde{\mathbf{b}}_0 (\mathbf{I} - \gamma \tilde{\mathbf{T}}_{\pi})^{-1} \sum_{i=1}^{M} w_i \tilde{\mathbf{C}}_{\pi}^{(i)} \end{aligned} \tag{9}$$

Expanding the last part of the above using (3), for each pair $\langle n, s \rangle$, we have:

$$\begin{aligned} &\sum_{i=1}^{M} w^{(i)} \tilde{C}_{\pi}^{(i)}(\langle n, s \rangle) \\ &\quad = \sum_{a \in A} \psi(n, a) \sum_{s' \in S} T(s, a, s') \sum_{i=1}^{M} w^{(i)} R^{(i)}(s, a, s') \\ &\quad = \sum_{a \in A} \psi(n, a) \sum_{s' \in S} T(s, a, s') R^{\mathbf{w}}(s, a, s') \end{aligned} \tag{10}$$

Therefore, we can simply combine all the $R^{(i)}$'s through WLC (a relatively cheap $O(|M||S|^2|A|)$ operation) and use gradient ascent as in the single-reward case. The same conclusion can be reached more generally from Eq. 7 with some simple algebraic manipulation and noting that WLC results in a single reward function provided that discount factors are equal across rewards:

$$\mathbf{E}\left(\sum_{i=0}^{M} w^{(i)} \sum_{t=0}^{\infty} \gamma^t R_t^{(i)} \mid \pi, \mathbf{b}_0 \right) = \mathbf{E}\left(\sum_{t=0}^{\infty} \gamma^t R_t^{\mathbf{w}} \mid \pi, \mathbf{b}_0 \right)$$

where $R_t^{\mathbf{w}} = \sum_{i=0}^{M} w^{(i)} R_t^{(i)}$. In other words, WLC with a specified $\mathbf{w}$ reduces a MR-POMDP to a POMDP with reward function $R^{\mathbf{w}}$.

Using this result, an approximate Pareto optimal set can be constructed by systematically sweeping through weights and performing gradient ascent for each weight vector. Unfortunately, there are three drawbacks to this approach:

1. local optimisation of FSCs is known to lead to poor local minima [5],
2. WLC only finds solutions on the convex hull of the Pareto optimal front and finally,
3. the objective space grows exponentially with the number of objectives making weight sweeps progressively more expensive.

These limitations motivate the need for a higher-level global optimisation algorithm, such as the multi-objective evolutionary methods described in the next section.

4. MULTI-OBJECTIVE HYBRID EAS FOR MR-POMDPS

Since genetic algorithms were first popularised in 1975 [13], evolutionary computation has grown into a significant research area encompassing a wide variety of methods. Of interest to us is the application of the evolutionary algorithms (EAs) to multi-objective problems. Multi-objective evolutionary algorithms (MOEAs), such as NSGA2 [27] and the strength pareto evolutionary algorithm (SPEA2) [33], have been successfully applied to a variety of complex test and real-world problems [8, 32, 10]. In this work, we used NSGA2 (as a representative of MOEAs using standard real-coded recombination and mutation operators) and MO-CMA-ES or MCMA [14], which represented the estimation-of-distribution (EDA) class of methods. The algorithms used in this work are "steady-state" in that they generate only one solution per iteration. Steady-state variants have been shown to have similar performance to their generation-based counterparts but with the added benefit of being easily parallelized for cluster or multi-core systems.

4.1 Steady-state NSGA2 and MCMA

NSGA2 is one of the most popular MOEAs in the literature, due to its simplicity and its effectiveness on a large class of problems. Similar to other MOEAs, NSGA2 iteratively generates new solutions using the simulated binary crossover (SBX) and polynomial mutation operators [27]. At each iteration, the population (together with the offspring) are sorted using a fast non-dominated sorting algorithm. Each solution is assigned a rank (lower ranks are better) and a second preference criteria, *crowding distance*, which approximates the density of solutions around the individual. NSGA2 is *elitist* in that it preserves only the top $|\mathbf{P}|$ solutions in the population $\mathbf{P}$ at each iteration.

More recently, there has been a growing interest in *estimation of distribution* (EDA) algorithms; in particular, MCMA, a multi-objective variant of the successful CMA-ES, has been shown to surpass NSGA2 on several benchmark problems [14]. MCMA is similar to NSGA2 in that it is also elitist, uses non-dominated sorting and density approximation (either crowding distance or the hypervolume indicator). A primary difference is in how they generate new solutions: while NSGA2 relies on "standard" evolutionary operators, MCMA builds a probabilistic model of good solutions from which it can sample from. The models built are multi-variate Gaussian and at each iteration, MCMA updates its estimate of the mean and covariance matrix, as well as strategy parameters. It is also worth mentioning that, unlike NSGA2, MCMA possesses nice invariance properties (e.g., invariance under rotation of the search space). In the next subsections, we discuss FSC representation and how we augmented both MOEAs with local-search using *dynamic operator selection*. Our algorithms are summarised in the flowchart shown in Fig. 3.

4.2 FSC Representation

Each FSC is represented with a vector (genome) $\mathbf{x} \in \mathbb{R}^{|N||A|+|N|^2|Z|}$. There are two segments to this genome for the action selection distribution $\psi(n, a)$ and node transition distribution $\eta(n, z, n')$ respectively. We refer to segments of the genome by $\mathbf{x}^{\psi}$ and $\mathbf{x}^{\eta}$. To ensure that probability distributions remained valid and the resulting evaluation function was differentiable, we use the soft-max function, e.g. $\psi(n, a) = P(a|n, \mathbf{x}^{\psi}) = \exp(\mathbf{x}^{\psi}[n, a])/Q$ where $\mathbf{x}^{\psi}[n, a]$ is the associated variable for $\psi(n, a)$ and Q is the normalisation factor.

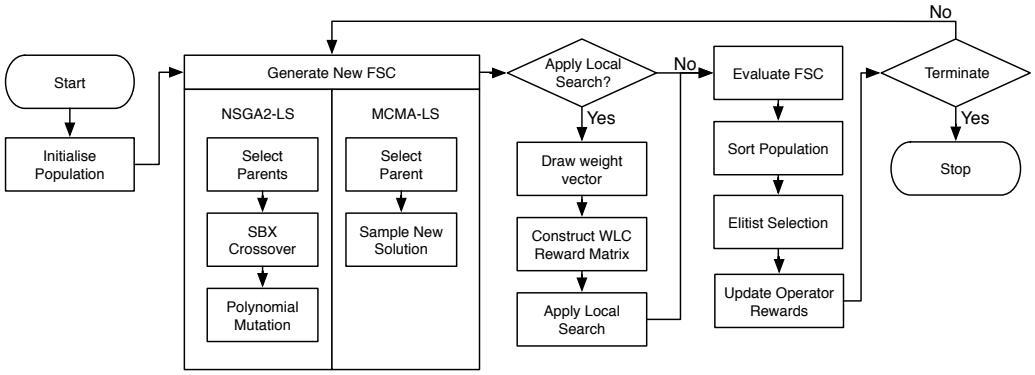


Figure 3: Flowchart summarising the NSGA2-LS and MCMA-LS hybrid/memetic algorithms.

4.3 Hybridisation with Local Search (WLC)

The FSCs we are attempting to optimise are large in the number of parameters; as stated, the genome consists of $|N||A|+|N|^2|Z|$ real variables. For example, given a problem with 4 actions and 5 observations, a 6-node FSC consists of 204 variables. While effective, multi-objective EAs may be slow to converge in such large search spaces and may not find solutions with sufficient precision.

A potential solution to this problem are *hybrid* or *memetic algorithms* [18, 22] that combine MOEAs with local-search methods; the intuition being that local-search can quickly locate good solutions that the evolutionary operators can build upon. We hybridised both NSGA2 and MCMA with gradient-based local-search as an operator and to keep our search computationally-feasible, we used our derivations in Section 3.3 to transform the MR-POMDP to a POMDP with reward function $R^{\mathbf{w}}$. We use a weight vector $\mathbf{w}$ drawn from a uniform distribution and apply an efficient conjugate-gradient method [11] until convergence or for a maximum of $5|\mathbf{x}|$ iterations.

4.4 Dynamic Local-Search Operator Selection

The question that naturally arises when using a local-search operator is how often we should apply it. On one hand, local-search often results in good solutions. On the other, it is resource-intensive and may cause the search to converge prematurely. In fact, the best operator rates are population and problem-dependent. In this work, we used a dynamic operator selection scheme based on operator rewards similar to [4]. Intuitively, we want more successful operators to be used more frequently. The reward given to each operator is a *cost-benefit* ratio where the "benefit" of using a particular operator is defined as the proportion of solutions the offspring is better than *relative to its parent* and the "cost" is simply the processing time used by the operator.

To clarify, suppose an operator o produces a FSC $\mathbf{x}_A$ from parent $\mathbf{x}_P$. We say $\mathbf{x}_A$ is better than some other FSC $\mathbf{x}_i$, denoted $\mathbf{x}_A \rhd \mathbf{x}_i$, if:

- $\mathbf{x}_A$ dominates $\mathbf{x}_i$ **OR**
- $\mathbf{x}_A$ and $\mathbf{x}_i$ are non-dominating but $\mathbf{x}_A$ has a better density measure (e.g., smaller crowding distance.)

We compare $\mathbf{x}_A$ to all other members of the population $\mathbf{x}_i \in \mathbf{P}$ and if $\mathbf{x}_A$ is better than $\mathbf{x}_i$ but its parent is not, then we increase the total benefit of the operator B_k^o. Hence:

$$B_k^o = \frac{1}{|\mathbf{P}|}\sum_{i=1}^{|\mathbf{P}|} \varrho(\mathbf{x}_A, \mathbf{x}_P, \mathbf{x}_i)$$

where

$$\varrho(\mathbf{x}_A, \mathbf{x}_P, \mathbf{x}_i) = \begin{cases} 1 & (\mathbf{x}_A \rhd \mathbf{x}_i) \wedge !(\mathbf{x}_P \rhd \mathbf{x}_i) \\ 0 & \text{otherwise} \end{cases}$$

The reward for an operator o at iteration t, R_t^o, is then the total benefit, B_k^o, divided by the processing time, C_k^o, needed to produce the offspring, averaged across the past β iterations:

$$R_t^o = \frac{1}{\beta}\sum_{k=1}^{\beta} \frac{B_k^o}{C_k^o}$$

The probability of selecting an operator is proportional to its reward and each update to the total operator reward occurs after β iterations using a memory decay technique:

$$T_t^o = \alpha T_{t-1}^o + (1-\alpha)R_t^o$$

where α is the learning-rate. In our work, $\beta = |\mathbf{P}|$ and $\alpha = 0.75$. Also, to prevent zero probabilities that would prevent the local-search operator from ever being called, the minimum selection probability is 0.001.

5. EXPERIMENTAL RESULTS

In this section, we present empirical results comparing NSGA2, MCMA, their hybridised versions NSGA2-LS and MCMA-LS, as well as a population-based local-search method (MLS). The MLS algorithm is identical to NSGA2 *except* that instead of using evolutionary operators, it *only* uses local-search. In our experiments, MLS served as a "baseline" algorithm that the MOEAs should surpass in order to be considered useful.

The aim of our experiments was to evaluate the relative performance of each of the aforementioned algorithms and our main hypothesis was that hybridisation with WLC and dynamic operator selection would improve the EAs. We begin this section by describing the three test problems used in this study, followed by details on our implementation, experimental setup and performance assessment methodology.

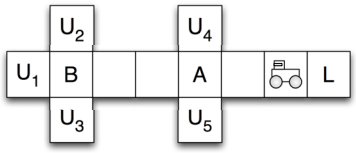


Figure 4: The multi-load-unload problem with one load point (L) and five unload points (U_1 to U_5). The robot starts at A and has to travel to L, load a box, travel to any one of the unload points and unload its cargo. Note that since the robot can only sense the walls around it, it can only partially determine its state; for example, A and B look identical to the robot.

5.1 MR-POMDP Test Problems

We used three different multi-objective problems in robotics, web-advertising and infectious disease control. Brief descriptions of each problem are given below and complete MR-POMDP models are available by contacting the author:

- **Multi-load-unload (MLU)**: In the single-objective load-unload problem, a robot has to traverse a corridor to the end where it "loads" an item, which it has to deliver and "unload" at a defined position to receive a reward. Here, we present a 24-state version with up to five unload points, each giving a separate (but equal) reward of 100 (Fig. 4). The robot can only sense the walls around it, giving it seven possible distinct observations. It has four movement actions (up, down, left and right) as well as two actuator actions (load and unload). Each action (except an unload in at the right point) results in a -1 reward. Furthermore, the robot experiences the possibility of sensor and motor failure.
- **Multi-Product Web Advertising (MWA)**: This problem is adapted from the POMDP by Cassandra [7] where the objective was to maximise revenue from the sale of two products on an online-store. An intelligent advertising agent has to decide, based on the webpages that a customer visits, which product he is likely to be interested in and advertise accordingly. The right advertisement can result in a purchase but the wrong advertisement might cause the person to leave the store. We extended this problem to multiple objectives by modelling each product as an separate reward and by adding an additional *reputation* objective. The reputation of the site would decrease every time a person left the site as a result of a wrong advertisement.
- **Multi-Criteria Anthrax Response Policy (MARP)**: The problem of anthrax outbreak detection was formulated as a POMDP by Izadi and Buckeridge [16] alongside public health experts. This POMDP is comprised of six states ("normal", " outbreak day 1" to "outbreak day 4" and "detected") with two observations ("suspicious" and "not suspicious") and four actions ("declare outbreak", "review records", "systematic studies" and "wait"). The original POMDP used a relatively complex reward function that combined the economic costs from multiple sources such as productivity loss, investigative costs, hospitalisation and medical treatment. In our multi-objective formulation, we have three-objectives to minimise: loss of life, number of false alarms and cost of investigation (in man-hours).

Note that a suffix is added to indicate the number of objectives, e.g., MLU-3 refers to the three-objective MLU problem (two of the unload points were disregarded). For each problem, we optimised 6-node stochastic FSCs.

5.2 Experimental Setup

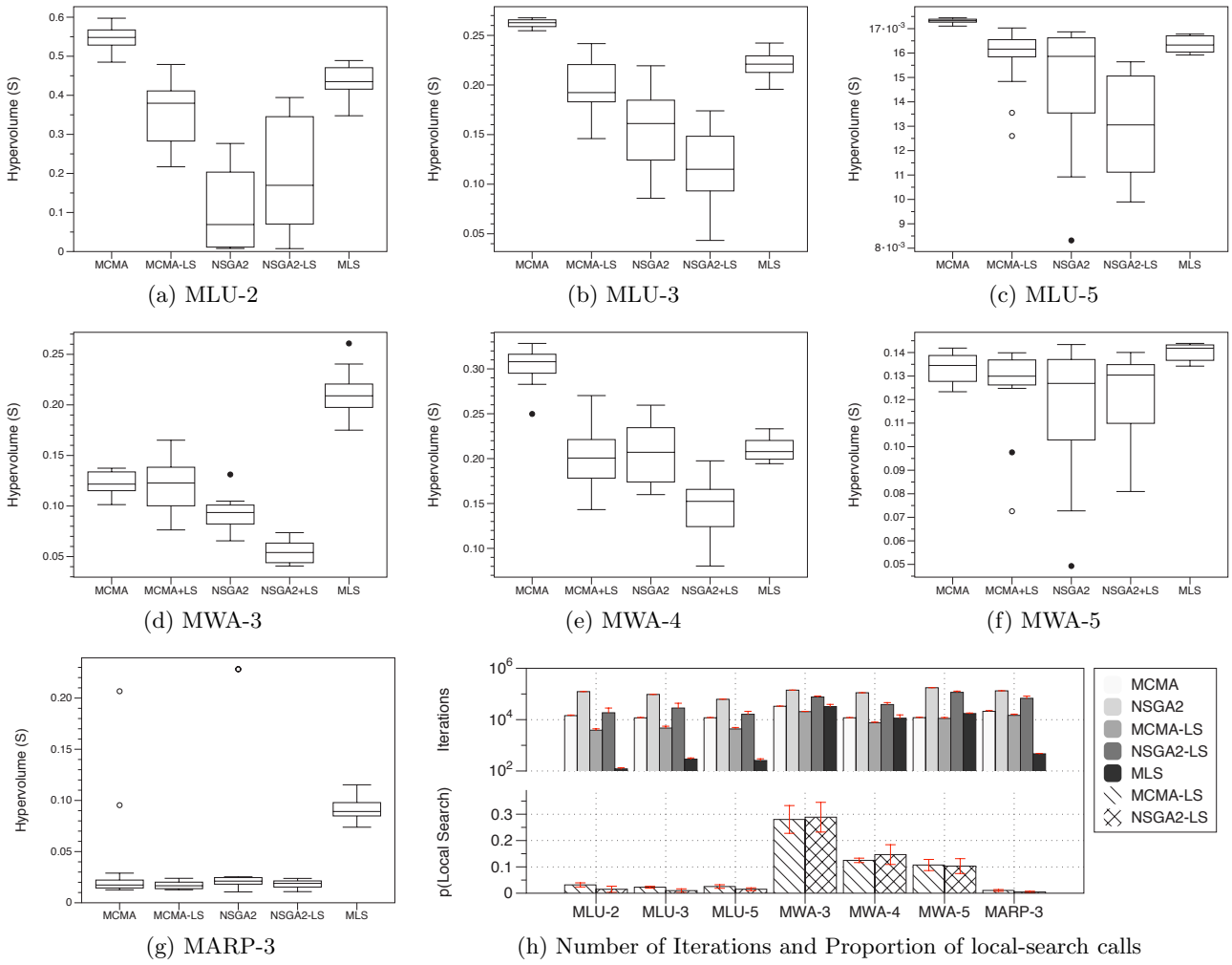
All our algorithms were implemented in C++ using the Shark Machine Learning library [15] and comparisons were performed using the PISA performance assessment framework [2, 19]. Non-dominated set comparisons were made using the hypervolume [31] indicator and significance tests on the indicator distributions were conducted using the non-parametric Mann-Whitney U test. Population sizes were set to 200 (for the bi-objective MLU problem) and 300 for the remaining problems. Other parameters were set to the defaults given in [27] (NSGA2) and [14] (MCMA). We take perspective of a user with a fixed computational budget and as such, each algorithm was allowed to run for a maximum running process time of 3600 CPU seconds. Each run was repeated 15 times and the reference (approximate) Pareto optimal sets were constructed by combining the final populations across all the runs.

5.3 Performance Results

The hypervolume indicator distributions of the final populations produced are summarised in boxplots (Figs. 5(a) to 5(g)). Overall, NSGA2-LS was the most successful algorithm and was the only method to consistently outperform MLS on every problem (at $p < 0.01$). NSGA2-LS also surpassed all other evolutionary algorithms on MLU-3/5 and MWA-3/4. On the remaining problems, the indicator distributions were not statistically different (at $p < 0.05$), i.e., NSGA2-LS did not perform worse.

Focussing on the question whether hybridisation with WLC was beneficial, we observed that MCMA-LS improved upon MCMA in four out of the seven tests, generating statistically better populations at $p < 0.01$ for MLU-2/3/5 and MWA-4. NSGA2-LS was better than NSGA2 in four of the tests ($p < 0.05$ for MLU-3/5 and $p < 0.01$ for MWA-3/4). When NSGA2-LS and MCMA-LS were not better than their regular counterparts, the indicator distributions were not statistically different at $p < 0.05$. Although more confirmatory tests are needed, these results favour our hypothesis that the hybridisation (with WLC and dynamic operator selection) improved the MOEAs. From Fig. 5(h), we observed that the proportion of local-search calls, p_{LS}, differed significantly between the test problems (ranging from 0.005 to 0.288). That said, p_{LS} was similar between algorithms and between changes in the number of objectives. In addition, local-search appeared more effective on MWA.

Surprisingly, MCMA appeared to be the worst among the evolutionary methods, often generating the poorest populations. A possible reason for this is suggested by Fig. 5(h) which shows that the number of iterations that MCMA was able to complete in the allotted time is an order of magnitude less than NSGA2; model maintenance appears to be expensive (given the relatively large search space) and the algorithm ran "slower" than NSGA2, which used computationally-cheap operators. When NSGA2 and MCMA were compared using an equal number of function evaluations ($\approx 10^4$), a different picture emerged with MCMA



(a) MLU-2 (b) MLU-3 (c) MLU-5

(d) MWA-3 (e) MWA-4 (f) MWA-5

(g) MARP-3 (h) Number of Iterations and Proportion of local-search calls

Figure 5: 5(a)-5(g) show boxplots of hypervolume indicator of final populations for the Multi-load-unload (MLU), Multi-Product Web Advertising (MWA) and Multi-Criteria Anthrax Response Policy (MARP) problems. 5(h) shows the number of iterations and the proportion of local-search calls.

producing better non-dominated sets for all problems ($p < 0.01$). Therefore, on larger problems with more expensive function evaluations, MCMA and MCMA-LS may produce competitive or superior non-dominated sets.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented the MR-POMDP for modelling multiple-objective problems in partially-observable environments. To solve MR-POMDPs, we explored the use of NSGA2, MCMA and developed two memetic variants, NSGA2-LS and MCMA-LS, for optimising non-dominated policies in the form of FSCs. As discussed, empirical results were showed that NSGA2-LS was statistically superior to the baseline method and the hybrid/memetic algorithms outperformed their regular counterparts. We consider this work as a step towards effective multi-objective solvers for MR-POMDPs. Both MOEAs and POMDP solvers have developed significantly in the past two decades and a cross-pollination of ideas is likely to yield novel developments. Certainly, different optimisation schemes should be explored, e.g., it is possible to use the MOEA to restrict the "structure" of the FSCs which would limit the number of variables that need to be locally-optimised. Moving beyond FSCs, EAs can also evolve other representations such as bayesian-based influence diagrams or biologically inspired models [9].

7. ACKNOWLEDGEMENTS

The authors thank members of the BioART lab for their advice and help in the preparation of this manuscript, and Masoumeh Tabaeh Izadi for sharing her POMDP model.

8. REFERENCES

[1] L. Barrett and S. Narayanan. Learning all optimal policies with multiple criteria. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 41–47, New York, NY, USA, 2008. ACM.

[2] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA — a platform and programming language independent interface for search algorithms. In C. M.

Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization (EMO 2003)*, Lecture Notes in Computer Science, pages 494 – 508, Berlin, 2003. Springer.

[3] P. A. N. Bosman and E. D. de Jong. Exploiting gradient information in numerical multi–objective evolutionary optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 755–762, New York, NY, USA, 2005. ACM.

[4] P. A. N. Bosman and E. D. de Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 627–634, New York, NY, USA, 2006. ACM.

[5] D. Braziunas and C. Boutilier. Stochastic local search for POMDP controllers. In *Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, 2004.

[6] D. Bryce, W. Cushing, and S. Kambhampati. Probabilistic planning is multi-objective. Technical report, Arizona State University, 2007.

[7] T. Cassandra. Tony's pomdp file repository page, Jan 1999.

[8] K. Deb and D. Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 1 edition, June 2001.

[9] Y. Demiris and B. Khadhouri. Hierarchical attentive multiple models for execution and recognition (hammer). *Robotics and Autonomous Systems*, 54(361-369), 2006.

[10] C. M. Fonseca and P. J. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3:1–16, 1995.

[11] W. W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM J. on Optimization*, 16(1):170–192, 2005.

[12] K. Harada, J. Sakuma, and S. Kobayashi. Local search for multiobjective function optimization: pareto descent method. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 659–666, New York, NY, USA, 2006. ACM.

[13] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[14] C. Igel, N. Hansen, and S. Roth. Covariance Matrix Adaptation for Multi-objective Optimization. *Evolutionary Computation*, 15(1):1–29, 2007.

[15] C. Igel, V. Heidrich-Meisner, and T. Glasmachers. Shark. *Journal of Machine Learning Research*, 9:993–996, 2008.

[16] M. Izadi and D. Buckeridge. Decision theoretic analysis of improving epidemic detection. In *AMIA Annu Symp Proc.*, pages 354–358, 2007.

[17] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1995.

[18] J. Knowles and D. Corne. Memetic algorithms for multiobjective optimization: Issues, methods and prospects. In *Recent Advances in Memetic Algorithms*, pages 313–352. 2005.

[19] J. Knowles, L. Thiele, and E. Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. 214, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, July 2005.

[20] O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision problems. In *Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 541–548, 1999.

[21] N. Meuleau, K.-E. Kim, L. P. Kaelbling, and A. R. Cassandra. Solving POMDPs by searching the space of finite policies. In *In Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 417–426. Morgan Kaufmann, 1999.

[22] Y. S. Ong and A. J. Keane. Meta-lamarckian learning in memetic algorithms. *Evolutionary Computation, IEEE Transactions on*, 8(2):99–110, April 2004.

[23] C. Papadimitriou and J. N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[24] J. Peters and S. Schaal. Policy Gradient Methods for Robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225, 2006.

[25] J. M. Porta, N. Vlassis, M. T. Spaan, and P. Poupart. Point-Based Value Iteration for Continuous POMDPs. *J. Mach. Learn. Res.*, 7:2329–2367, 2006.

[26] P. Poupart and C. Boutilier. Bounded finite state controllers. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[27] M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors. *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

[28] R. D. Smallwood and E. J. Sondik. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Operations Research*, 21(5):1071–1088, 1973.

[29] White. Vector maxima for infinite-horizon stationary markov decision process. *IMA J Management Math*, 9(1):1–17, January 1998.

[30] M. A. Wiering and E. D. de Jong. Computing optimal stationary policies for multi-objective markov decision processes. april 2007.

[31] E. Zitzler. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, August 2002.

[32] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[33] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, 2001.