Abstract

Many real-world tasks require making decisions that involve multiple possibly conflicting objectives. To succeed in such tasks, intelligent systems need planning or learning algorithms that can efficiently find different ways of balancing the trade-offs that such objectives present.

In this tutorial, we provide an introduction to decision-theoretic approaches to coping with multiple objectives. We first present an overview of multi-objective decision problems, with real-world examples. Then, we show that different assumptions about these problems lead to different solution concepts such as the convex hull and the Pareto front. Next, we provide an overview of state-of-the-art algorithms for tackling them, such as multi-objective variants of dynamic programming. Finally, we highlight some applications of multi-objective methods and discuss some of the most important open questions.
Multi-Objective Decision Making

Shimon Whiteson & Diederik M. Roijers
Informatics Institute
University of Amsterdam
July 25, 2015

Schedule
- 8:45-9:30: Motivation & Concepts (Shimon)
- 9:30-9:40: Short Break
- 9:40-10:30: Motivation & Concepts cont’d (Shimon)
- 10:30-11:00: Coffee Break
- 11:00-11:45: Methods & Applications (Diederik)
- 11:45-11:55: Short Break
- 11:55-12:45: Methods & Applications cont’d (Diederik)

Note
- Get the latest version of the slides at:
  https://staff.fnwi.uva.nl/d.m.roijers/notutorial.html
- This tutorial is based on our survey article:

Part 1: Motivation & Concepts
- Multi-Objective Motivation
- MDPs & MOMDPs
- Problem Taxonomy
- Solution Concepts

Medical Treatment
Chance of being cured, having side effects, or dying

Traffic Coordination
Latency, throughput, fairness, environmental impact, etc.
Mining Commodities

Gold collected, silver collected

[Roijers et al. 2013, 2014]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 7 / 104

Grid World

Getting the treasure, minimising fuel costs

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 8 / 104

Do We Need Multi-Objective Models?

Sutton’s Reward Hypothesis: “All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward).”

Source: http://rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html

V: ⇧ ! R
V⇡ = E⇡[Pt rt]
⇡⇤ = max⇡ V⇡

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 9 / 104

Do We Need Multi-Objective Models?

The weak argument: real-world problems are multi-objective!

V : Π → ℜn

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 10 / 104

Why Multi-Objective Decision Making?

Sutton’s Reward Hypothesis: “All of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward).”

Source: http://rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html

V: ⇧ ! R
V⇡ = E⇡[Pt rt]
⇡⇤ = max⇡ V⇡
**Why Multi-Objective Decision Making?**

- **The weak argument:** real-world problems are multi-objective!  
  \[ \mathbf{V} : \Pi \to \mathbb{R}^n \]

- **Objection:** why not just scalarize?

**Scalarization function** projects multi-objective value to a scalar:

\[ V^\omega = f(V^\pi, w) \]

Linear case:

\[ V^\omega = \sum_{i=1}^{n} w_i V^{\pi}_i = w \cdot V^\pi \]

A priori prioritization of the objectives

- The weak argument is necessary but not sufficient

---

**Why Multi-Objective Decision Making?**

- **The strong argument:** a priori scalarization is sometimes impossible, infeasible, or undesirable

- Instead produce the **coverage set** of undominated solutions

**Unknown-weights scenario**

- Weights known in **execution phase** but not in **planning phase**
- Example: mining commodities [Roijers et al. 2013]

**Known-weights scenario:** scalarization yields intractable problem

- **Decision-support** scenario
  - Quantifying priorities is infeasible
  - Choosing between options is easier
  - Example: medical treatment

---

**Why Multi-Objective Decision Making?**

- **The strong argument:** a priori scalarization is sometimes impossible, infeasible, or undesirable

- Instead produce the **coverage set** of undominated solutions

**Unknown-weights scenario**

- Weights known in **execution phase** but not in **planning phase**
- Example: mining commodities [Roijers et al. 2013]
Why Multi-Objective Decision Making?

- **Decision-support scenario**
  - Quantifying priorities is infeasible
  - Choosing between options is easier
  - Example: medical treatment

- **Known-weights scenario**: scalarization yields intractable problem

Summary of Motivation

Multi-objective methods are useful because many problems are naturally characterized by multiple objectives and cannot be easily scalarized a priori.

The burden of proof rests with the a priori scalarization, not with the multi-objective modeling.

Markov Decision Process (MDP)

- A single-objective MDP is a tuple \( (S, A, T, R, µ, γ) \) where:
  - \( S \) is a finite set of states
  - \( A \) is a finite set of actions
  - \( T : S \times A \times S \rightarrow [0, 1] \) is a transition function
  - \( R : S \times A \times S \rightarrow \mathbb{R} \) is a reward function
  - \( µ : S \rightarrow [0, 1] \) is a probability distribution over initial states
  - \( γ \in [0, 1) \) is a discount factor

Returns & Policies

- Goal: maximize expected return, which is typically additive:
  \[
  R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}
  \]

- A stationary policy conditions only on the current state:
  \[
  π : S \times A \rightarrow [0, 1]
  \]

- A deterministic stationary policy maps states directly to actions:
  \[
  π : S \rightarrow A
  \]
Value Functions in MDPs

- A state-independent value function $V^\pi$ specifies the expected return when following $\pi$ from the initial state:
  \[ V^\pi = E[R_0 | \pi] \quad (1) \]
- A state value function of a policy $\pi$:
  \[ V^\pi(s) = E[R_0 | \pi, s_0 = s] \]
- The Bellman equation restates this expectation recursively for stationary policies:
  \[ V^\pi(s) = \sum_a \pi(s,a) \sum_{s'} T(s,a,s')(R(s,a,s') + \gamma V^\pi(s')) \]

Optimality in MDPs

Theorem

For any additive infinite-horizon single-objective MDP, there exists a deterministic stationary optimal policy [Howard 1960]

- All optimal policies share the same optimal value function:
  \[ V^\pi(s) = \max_a V^\pi(s, a) \]
  \[ V^\pi(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^\pi(s')] \]
- Extract the optimal policy using local action selection:
  \[ \pi^*(s) = \arg \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^\pi(s')] \]

Multi-Objective MDP (MOMDP)

- Vector-valued reward and value:
  \[ R : S \times A \times S \rightarrow \mathbb{R}^n \]
  \[ V^\pi = E[\sum_{k=1}^{\infty} \gamma^k r_{k+1} | \pi] \]
  \[ V^\pi(s) = E[\sum_{k=1}^{\infty} \gamma^k r_{k+1} | \pi, s_0 = s] \]
- $V^\pi(s)$ imposes only a partial ordering, e.g.,
  \[ V_i^\pi(s) > V_j^\pi(s) \text{ but } V_i^\gamma(s) < V_j^\gamma(s). \]
- Definition of optimality no longer clear

Part 1: Motivation & Concepts

- Multi-Objective Motivation
- MDPs & MOMDPs
- Problem Taxonomy
- Solution Concepts

Axiomatic vs. Utility-Based Approach

- Axiomatic approach: define optimal solution set to be Pareto front

Axiomatic vs. Utility-Based Approach

- Axiomatic approach: define optimal solution set to be Pareto front
- Utility-based approach:
  - Execution phase: select one policy maximizing scalar utility $V^w_i$, where $w$ may be hidden or implicit
Axiomatic vs. Utility-Based Approach

- **Axiomatic approach**: define optimal solution set to be Pareto front
- **Utility-based approach**:
  - **Execution phase**: select one policy maximizing scalar utility $V_w^*$, where $w$ may be hidden or implicit
  - **Planning phase**: find set of policies containing optimal solution for each possible $w$; if $w$ unknown, size of set generally $> 1$

Three Factors

- Multi-objective scenario
  - Known weights $\rightarrow$ single policy
  - Unknown weights or decision support $\rightarrow$ multiple policies
- Properties of scalarization function
  - Linear
  - Monotonically increasing
- Allowable policies
  - Deterministic
  - Stochastic

Problem Taxonomy

<table>
<thead>
<tr>
<th>Single policy (known weights)</th>
<th>Multiple policies (unknown weights or decision support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear deterministic stationary policy</td>
<td>Convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>Monotonically increasing deterministic stationary policy</td>
<td>Pareto coverage set of deterministic non-stationary policies</td>
</tr>
<tr>
<td>Deterministic non-stationary policy</td>
<td>Convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>Stochastic deterministic stationary policy</td>
<td>Deterministic non-stationary policies</td>
</tr>
<tr>
<td>One mixture policy of two or more deterministic stationary policies</td>
<td>One mixture policy of two or more deterministic non-stationary policies</td>
</tr>
</tbody>
</table>

Part 1: Motivation & Concepts

- Multi-Objective Motivation
- MDPs & MOMDPs
- Problem Taxonomy
- Solution Concepts
Linear Scalarization Functions
- Computes inner product of \( w \) and \( V^\pi \):
  \[
  V^\pi = \sum_{i=1}^n w_i V_i^\pi = w \cdot V^\pi, \; w \in \mathbb{R}^n
  \]
- \( w_i \) quantifies importance of \( i \)-th objective
- Simple and intuitive, e.g., when utility translates to money:
  \[
  \text{revenue} = \# \text{cans} \times \text{ppc} + \# \text{bottles} \times \text{ppb}
  \]
- \( \text{revenue} \) typically constrained to be a convex combination:
  \[
  \forall i \; w_i \geq 0, \quad \sum_i w_i = 1 \]

Linear Scalarization & Single Policy
- No special methods required: just apply \( f \) to each reward vector
- Inner product distributes over addition yielding a normal MDP:
  \[
  V^\pi = w \cdot V^\pi = w \cdot \left[ E \left[ \sum_{k=0}^\infty r_{t+k+1} \right] \right] = E \left[ \sum_{k=0}^\infty (w \cdot r_{t+k+1}) \right]
  \]
- Apply standard methods to an MDP with:
  \[
  R(s, a, s') = w \cdot R(s, a, s'), \quad (2)
  \]
  yielding a single deterministic stationary policy

Problem Taxonomy
- single policy (known weights)
  deterministic stochastic
deterministic stochastic
- linear scalarization
  one deterministic stationary policy
  convex coverage set of deterministic stationary policies
- monotonically increasing scalarization
  one deterministic non-stationary policy
  one mixture policy of two or more deterministic stationary policies
  Pareto coverage set of deterministic non-stationary policies
  convex coverage set of deterministic stationary policies

Example: collecting bottles and cans

Note: only call in taxonomy that does not require multi-objective methods
Multiple Policies

- Unknown weights or decision support → multiple policies
- During planning $w$ is unknown
- Size of solution set is generally $> 1$
- Set should not contain policies suboptimal for all $w$

Undominated & Coverage Sets

Definition

The undominated set $U(\Pi)$, is the subset of all possible policies $\Pi$ for which there exists a $w$ for which the scalarized value is maximal,

$$U(\Pi) = \{ \pi : \pi \in \Pi \land \exists w(v'(\pi \in \Pi) V^w_\pi \geq V^w_{\pi'} \}$$

Definition

A coverage set $CS(\Pi)$ is a subset of $U(\Pi)$ that, for every $w$, contains a policy with maximal scalarized value, i.e.,

$$CS(\Pi) \subseteq U(\Pi) \land (\forall w) (\exists \pi) \left( \pi \in CS(\Pi) \land \forall v'(\pi \in \Pi) V^w_\pi \geq V^w_{\pi'} \right)$$

Example

<table>
<thead>
<tr>
<th>$V^w_\pi$</th>
<th>$w = \text{true}$</th>
<th>$w = \text{false}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi = \pi_1$</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>$\pi = \pi_2$</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>$\pi = \pi_3$</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>$\pi = \pi_4$</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

- One binary weight feature: only two possible weights
- Weights are not objectives but two possible scalarizations

Execution Phase

- Single policy selected from $CS(\Pi)$ and executed
- **Unknown weights**: weights revealed and maximizing policy selected:

  $$\pi^* = \arg \max_{\pi \in CS(\Pi)} V^w_\pi$$

- **Decision support**: $CS(\Pi)$ is manually inspected by the user
Linear Scalarization & Multiple Policies

**Definition**
The convex hull \( CH(\Pi) \) is the subset of \( \Pi \) for which there exists a \( w \) that maximizes the linearly scalarized value:

\[
CH(\Pi) = \{ \pi : \pi \in \Pi \land \exists w(\pi' \in \Pi) \ w \cdot V^{\pi'} \geq V^{\pi} \}
\]

**Problem Taxonomy**

- **Linear Scalarization & Multiple Policies**
- **Visualization**
  - Objective Space
  - Weight Space
  - \( V_w = w_0 V_0 + w_1 V_1 \), \( w_0 = 1 - w_1 \)

**Monotonically Increasing Scalarization Functions**
- Mining example: \( V^{\pi_1} = (1, 0) \), \( V^{\pi_2} = (0, 3) \), \( V^{\pi_3} = (1, 1) \)
- Choosing \( V^{\pi_3} \) implies nonlinear scalarization function

**Example:** mining gold and silver
Monotonically Increasing Scalarization Functions

Definition
A scalarization function is strictly monotonically increasing if changing a policy such that its value increases in one or more objectives, without decreasing in any other objectives, also increases the scalarized value:
\[(\forall i \ V_i^\pi \geq V_i^\pi' \wedge \exists i \ V_i^\pi > V_i^\pi') \Rightarrow (\forall w \ V_w^\pi > V_w^\pi')\]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 19 / 104

Monotonically Increasing Scalarization Functions

Definition
A scalarization function is strictly monotonically increasing if changing a policy such that its value increases in one or more objectives, without decreasing in any other objectives, also increases the scalarized value:
\[(\forall i \ V_i^\pi \geq V_i^\pi' \wedge \exists i \ V_i^\pi > V_i^\pi') \Rightarrow (\forall w \ V_w^\pi > V_w^\pi')\]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 19 / 104

Nonlinear Scalarization Can Destroy Additivity

- Nonlinear scalarization and expectation do not commute:
  \[V_w^\pi = f(E(V^\pi, w)) = f(E[\sum_{k=1}^{\infty} r_{t+k-1}w]) \neq E[\sum_{k=1}^{\infty} f(r_{t+k-1}, w)]\]
- Bellman-based methods not applicable
- Local action selection no longer yields an optimal policy:
  \[\pi^*(s) \neq \arg \max V^*(s)\]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 19 / 104

Deterministic vs. Stochastic Policies

- Stochastic policies are fine in most settings
- Sometimes inappropriate, e.g., medical treatment
- In MDPs, requiring deterministic policies is not restrictive
- Optimal value attainable with deterministic stationary policy:
  \[\pi^*(s) = \arg \max a \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]\]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 19 / 104

Deterministic vs. Stochastic Policies

- Stochastic policies are fine in most settings
- Sometimes inappropriate, e.g., medical treatment
- In MDPs, requiring deterministic policies is not restrictive
- Optimal value attainable with deterministic stationary policy:
  \[\pi^*(s) = \arg \max a \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]\]

Whiteson & Roijers (UvA) Multi-Objective Decision Making July 25, 2015 19 / 104

White’s Example (1982)

- 3 actions: \(R(a_1) = (3.0), R(a_2) = (0.3), R(a_3) = (1.1)\)
Problem Taxonomy

**White’s Example (1982)**
- 3 actions: \( R(a_1) = (3, 0), R(a_2) = (0, 3), R(a_3) = (1, 1) \)
- 3 deterministic stationary policies, all Pareto-optimal:
  \[
  V^1 = \left( \frac{3}{1-\gamma}, 0 \right),
  V^2 = \left( 0, \frac{3}{1-\gamma} \right),
  V^3 = \left( \frac{1}{1-\gamma}, \frac{1}{1-\gamma} \right)
  \]
- \( \pi_{ns} \) alternates between \( a_1 \) and \( a_2 \), starting with \( a_2 \):
  \[
  V^{\pi_{ns}} = \left( \frac{3}{1-\gamma}, \frac{3\gamma}{1-\gamma} \right)
  \]

Thus \( \pi_{ns} \gtrless \pi_\gamma \) when \( \gamma \geq 0.5 \), e.g., \( \gamma = 0.5 \) and \( f(V^1) = V^1_1V^1_2 \):
\[
V^1 = V^3 = 0, V^2 = 4, V^{\pi_{ns}} = 8
\]

**Problem Taxonomy**

<table>
<thead>
<tr>
<th>Linear Scalarization</th>
<th>Monotonically Increasing Scalarization</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Deterministic Stationary Policy</td>
<td>One Deterministic Non-Stationary Policy</td>
</tr>
<tr>
<td>Convex Coverage Set of Deterministic Stationary Policies</td>
<td>Pareto Coverage Set of Deterministic Non-Stationary Policies</td>
</tr>
</tbody>
</table>

**Example:** Radiation vs. chemotherapy

**White’s Example (1982)**
- 3 actions: \( R(a_1) = (3, 0), R(a_2) = (0, 3), R(a_3) = (1, 1) \)
- 3 deterministic stationary policies, all Pareto-optimal:
  \[
  V^1 = \left( \frac{3}{1-\gamma}, 0 \right),
  V^2 = \left( 0, \frac{3}{1-\gamma} \right),
  V^3 = \left( \frac{1}{1-\gamma}, \frac{1}{1-\gamma} \right)
  \]
- \( \pi_{ns} \) alternates between \( a_1 \) and \( a_2 \), starting with \( a_2 \):
  \[
  V^{\pi_{ns}} = \left( \frac{3}{1-\gamma}, \frac{3\gamma}{1-\gamma} \right)
  \]

**Mixture Policies**
- A mixture \( \pi_m \) selects \( i \)-th policy from set of \( N \) deterministic policies with probability \( p_i \), where \( \sum_{i=1}^{N} p_i = 1 \)
- Values are convex combination of values of constituent policies
- In White’s example, replace \( \pi_{ns} \) by \( \pi_m \):
  \[
  V^{\pi_m} = p_1V^1 + (1-p_1)V^2 = \left( \frac{3}{1-\gamma}, \frac{3(1-p_1)}{1-\gamma} \right)
  \]
**Problem Taxonomy**

<table>
<thead>
<tr>
<th>single policy (known weights)</th>
<th>multiple policies (unknown weights or decision support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>deterministic</td>
<td>stochastic</td>
</tr>
</tbody>
</table>

**Linear scalarization**
- one deterministic stationary policy

**Monotonically increasing scalarization**
- one deterministic non-stationary policy
- one mixture policy of two or more deterministic stationary policies

**Example:** studying vs. networking

**Pareto Sets**

**Definition**

The Pareto front is the set of all policies that are not Pareto dominated:

\[
PF(\Pi) = \{ \pi : \pi \in \Pi \land \exists \pi' \in \Pi, V^\pi' \succ_p V^\pi \}
\]

**Definition**

A Pareto coverage set is a subset of \( PF(\Pi) \) such that, for every \( \pi' \in \Pi \), it contains a policy that either dominates \( \pi' \) or has equal value to \( \pi' \):

\[
PCS(\Pi) \subseteq PF(\Pi) \land \forall (\pi' \in \Pi) (\exists \pi \in PCS(\Pi) \land (V^\pi \succ_p V^\pi' \lor V^\pi = V^\pi'))
\]

**Visualization**

![Objective Space](image1)

![Weight Space](image2)
Problem Taxonomy

<table>
<thead>
<tr>
<th>single policy (known weights)</th>
<th>multiple policies (unknown weights or decision support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>deterministic</td>
<td>stochastic</td>
</tr>
<tr>
<td>deterministic</td>
<td>stochastic</td>
</tr>
<tr>
<td>linear scalarization</td>
<td></td>
</tr>
<tr>
<td>one deterministic stationary policy</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>deterministic stationary policy</td>
<td></td>
</tr>
<tr>
<td>monotonically increasing scalarization</td>
<td></td>
</tr>
<tr>
<td>one deterministic non-stationary policy</td>
<td>one mixture policy of two or more deterministic stationary policies, Pareto coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>deterministic non-stationary policy</td>
<td></td>
</tr>
<tr>
<td>convex coverage set of deterministic stationary policies</td>
<td></td>
</tr>
</tbody>
</table>

Example: radiation vs. chemotherapy (again)

Mixture Policies

A PCS(\Pi) can be constructed by mixing policies in CCS(\Pi_{CS})

Example: radiation vs. chemotherapy (again)

Note: only setting that case requires a Pareto front!

Part 2: Methods and Applications

- Convex Coverage Set Planning Methods
  - Inner Loop: Convex Hull Value Iteration
  - Outer Loop: Optimistic Linear Support
- Pareto Coverage Set Planning Methods
  - Inner loop (non-stationary): Pareto-Q
  - Outer loop issues
- Beyond the Taxonomy
- Applications
Background: Value Iteration

- Initial estimate value estimate $V_0(s)$
- Apply Bellman backups until convergence:
  $$V_{k+1}(s) = \max_a \sum_{s'} T(s, a, s') \left[ R(s, a, s') + \gamma V_k(s') \right]$$
  $$\Rightarrow Q_{k+1}(s, a) = \max_a \sum_{s'} T(s, a, s') \left[ R(s, a, s') + \gamma V_k(s') \right]$$
- Optimal policy is easy to retrieve from Q-table

Taxonomy

<table>
<thead>
<tr>
<th></th>
<th>single policy</th>
<th>multiple policies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>deterministic</td>
<td>stochastic</td>
</tr>
<tr>
<td>linear</td>
<td>one deterministic stationary policy</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>scalarization</td>
<td>one deterministic non-stationary policy</td>
<td>one mixture policy of two or more deterministic stationary policies</td>
</tr>
<tr>
<td>monotonically increasing scalarization</td>
<td>one deterministic non-stationary policy</td>
<td>one mixture policy of two or more deterministic stationary policies</td>
</tr>
</tbody>
</table>

- Known transition and reward functions → planning
- Unknown transition and reward functions → learning
Scalarize MOMDP + Value Iteration

- For known $w$
  $$V_w^\alpha = w \cdot E \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \right] = E \left[ \sum_{k=0}^{\infty} \gamma^k (w \cdot r_{t+k+1}) \right].$$

- Scalarize reward function of MOMDP
  $$R_w = w \cdot R$$

- Apply standard VI

Scalarized Value Iteration

- Adapt Bellman backup:
  $$w \cdot V_{k+1}(s) \leftarrow \max_a Q_{k+1}(s, a),$$
  $$Q_{k+1}(s, a) \leftarrow \sum_{s'} T(s, a, s') \left[ R(s, a, s') + \gamma V_k(s') \right]$$

- Returns multi-objective value.

Taxonomy

- single policy (known weights)
  - deterministic stationary policy
  - one deterministic stationary policy
  - convex coverage set of deterministic stationary policies

- multiple policies (unknown weights or decision support)
  - stochastic
  - mixture of two or more deterministic stationary policies
  - convex coverage set of deterministic stationary policies
  - Pareto coverage set of deterministic non-stationary policies
**Inner versus Outer Loop**

- Inner loop
  - Adapting operators of single objective method (e.g., value iteration)
  - Series of multi-objective operations (e.g., Bellman backups)

- Outer loop
  - Single objective method as subroutine
  - Series of single-objective problems

---

**Inner Loop: Convex Hull Value Iteration**

- Barrett & Narayanan (2008)
- Idea: do the backup for all \( w \) in parallel
- New backup operators must handle sets of values.
- At backup:
  - generate all value vectors for \( s, a \)-pair
  - prune away those that are not optimal for any \( w \)
- Only need deterministic stationary policies

**CHVI Example**

- Extremely simple MOMDP:
  1 state: \( s \)
  2 actions: \( a_1 \) and \( a_2 \)
- Deterministic transitions
- Deterministic rewards:
  \( R(s, a_1, s) \rightarrow (2, 0) \)
  \( R(s, a_2, s) \rightarrow (0, 2) \)
- \( \gamma = 0.5 \)
- \( V_0(s) = \{(0, 0)\} \)
CHVI Example

- Deterministic rewards:
  \( R(s_{a_1}, s) \rightarrow (2.0) \)
  \( R(s_{a_2}, s) \rightarrow (0.2) \)
- \( \gamma = 0.5 \)
- Iteration 1:
  \( V_0(s) = \{(0.0)\} \)

- Deterministic rewards:
  \( R(s_{a_1}, s) \rightarrow (2.0) \)
  \( R(s_{a_2}, s) \rightarrow (0.2) \)
- \( \gamma = 0.5 \)
- Iteration 1:
  \( V_0(s) = \{(0.0)\} \)
  \( Q_1(s_{a_1}) = \{(2.0)\} \)
  \( Q_1(s_{a_2}) = \{(0.2)\} \)
  \( V_1(s) = \text{CPrune}(\bigcup_a Q_1(s, a)) = \{(2.0), (0.2)\} \)

CHVI Example

- Deterministic rewards:
  \( R(s_{a_1}, s) \rightarrow (2.0) \)
  \( R(s_{a_2}, s) \rightarrow (0.2) \)
- \( \gamma = 0.5 \)
- Iteration 2:
  \( V_2(s) = \{(2.0), (0.2)\} \)

- Deterministic rewards:
  \( R(s_{a_1}, s) \rightarrow (2.0) \)
  \( R(s_{a_2}, s) \rightarrow (0.2) \)
- \( \gamma = 0.5 \)
- Iteration 2:
  \( V_2(s) = \{(2.0), (0.2)\} \)
  \( Q_2(s_{a_1}) = \{(3.0), (2.1)\} \)
  \( Q_2(s_{a_2}) = \{(1.2), (0.3)\} \)
  \( V_2(s) = \text{CPrune}(\{(3.0), (2.1), (1.2), (0.3)\}) \)
CHVI Example

Determinate rewards:
- $R(s, a_1, s) \rightarrow (2.0)$
- $R(s, a_2, s) \rightarrow (0.2)$
- $\gamma = 0.5$
- Iteration 2:
  - $V_1(s) = \{(2.0, 0.2)\}$
  - $Q_1(s, a_1) = \{(3.0, 2.1)\}$
  - $Q_1(s, a_2) = \{(1.2, 0.3)\}$
  - $V_2(s) = \{(2.0, 0.3)\}$

Convex Hull Value Iteration

- CPrune retains at least one optimal vector for each $w$
- Therefore, $V_w$ that would have been computed by VI is kept
- CHVI does not retain excess value vectors

Outer Loop

- Repeatedly calls a single-objective solver
- Generic multi-objective method
  - multi-objective coordination graphs
  - multi-objective (multi-agent) MDPs
  - multi-objective partially observable MDPs

Convex Hull Value Iteration

- CPrune retains at least one optimal vector for each $w$
- Therefore, $V_w$ that would have been computed by VI is kept
- CHVI does not retain excess value vectors
- CHVI generates a lot of excess value vectors
- Removal with linear programs (CPrune) is expensive

Outer Loop: Optimistic Linear Support

- Optimistic linear support (OLS) adapts and improves linear support for POMDPs (Cheng (1988))
- Solves scalarized instances for specific $w$
Outer Loop: Optimistic Linear Support

- **Optimistic linear support (OLS)** adapts and improves linear support for POMDPs (Cheng (1988))
- Solves *scalarized* instances for specific $w$
- Terminates after checking only a finite number of weights
- Returns exact CCS

Optimistic Linear Support

- Priority queue, $Q$, for corner weights
- Maximal possible improvement $\Delta$ as priority
- Stop when $\Delta < \varepsilon$
Optimistic Linear Support

- Solving scalarized instance not always possible
- $\varepsilon$-approximate solver
- Produces an $\varepsilon$-CCS

Whiteson & Roijers (UvA)  Multi-Objective Decision Making  July 25, 2015  78 / 104

Reusing Value Functions

- Observation: when $w$ are close so are optimal $V$
- Hot start with $V(s)$ from close $w$
- Optimistic Linear Support with Alpha Reuse (OLSAR)

Taxonomy

<table>
<thead>
<tr>
<th>single policy (known weights)</th>
<th>multiple policies (unknown weights or decision support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>deterministic</td>
<td>deterministic stationary policies</td>
</tr>
<tr>
<td>stochastic</td>
<td>deterministic stationary policies</td>
</tr>
<tr>
<td>linear scalarization</td>
<td>monotonically increasing scalarization</td>
</tr>
<tr>
<td>deterministic stationary policy</td>
<td>one deterministic non-stationary policy</td>
</tr>
<tr>
<td>non-stationary policy</td>
<td>one mixture policy of two or more deterministic stationary policies</td>
</tr>
<tr>
<td>convex coverage set of deterministic stationary policies</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
</tbody>
</table>

Inner Loop: Pareto-Q

- Similar to CHVI
- Different pruning operator
- Pairwise comparisons: $V(s) \succ_P V'(s)$
- Comparisons cheaper but much more vectors
- Converges to correct Pareto coverage set (White (1982))
- Executing a policy is no longer trivial (Van Moffaert & Nowé (2014))
**Inner Loop: Pareto-Q**

- Compute all possible vectors
  \[ Q_{k+1}(s, a) = \bigoplus_{s'} T(s, a, s') [R(s, a, s') + \gamma V_k(s')] \]

  where \( u + V = \{ u + v : v \in V \} \),
  \( U \oplus V = \{ u + v : u \in U \land v \in V \} \)

- Take the union across \( a \)

- Prune Pareto-dominated vectors
  \[ V_{k+1}(s) \leftarrow \text{PPrune} \left( \bigcup_{a} Q_{k+1}(s, a) \right) \]

---

**Pareto-Q Example**

- Extremely simple MOMDP:
  - 1 state: \( s \)
  - 2 actions: \( a_1 \) and \( a_2 \)
- Deterministic rewards:
  - \( R(s, a_1, s) \rightarrow (2.0) \)
  - \( R(s, a_2, s) \rightarrow (0.2) \)
  - \( \gamma = 0.5 \)
  - \( V_0(s) = \{(0.0)\} \)

---

**Pareto-Q Example**

- Deterministic rewards:
  - \( R(s, a_1, s) \rightarrow (2.0) \)
  - \( R(s, a_2, s) \rightarrow (0.2) \)
  - \( \gamma = 0.5 \)
- Iteration 1:
  - \( V_0(s) = \{(0.0)\} \)
  - \( Q_1(s, a_1) = \{(2.0)\} \)
  - \( Q_1(s, a_2) = \{(0.2)\} \)
  - \( V_1(s) = \text{PPrune} \left( \bigcup_{a} Q_1(s, a) \right) = \{(2.0), (0.2)\} \)

---

**Pareto-Q Example**

- Deterministic rewards:
  - \( R(s, a_1, s) \rightarrow (2.0) \)
  - \( R(s, a_2, s) \rightarrow (0.2) \)
  - \( \gamma = 0.5 \)
- Iteration 2:
  - \( V_2(s) = \{(2.0), (0.2)\} \)
  - \( Q_2(s, a_1) = \{(3.0), (2.1)\} \)
  - \( Q_2(s, a_2) = \{(1.2), (0.3)\} \)
  - \( V_2(s) = \text{PPrune} \left( \{(3.0), (2.1), (1.2), (0.3)\} \right) \)
Inner Loop: Pareto-Q

- PCS size can explode
- No longer deterministic
- Cannot read policy from Q-table
- Except for first action
- “Track” a policy during execution (Van Moaert & Nowé (2014))
  - For deterministic transitions: $ s, a \rightarrow s' $
  - From $ Q_{t+1}(s, a) $ subtract $ R(s, a) $
  - Correct for discount factor $ \rightarrow V_{t+1}(s') $
  - Find $ V_{t+1}(s') $ in Q-tables for $ s' $
- For stochastic transitions, see Kristof van Moffaert’s PhD thesis

Outer Loop?

- Outer loop very difficult:
  $$ V^w = E \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | w \right] \neq E \left[ \sum_{k=0}^{\infty} \gamma^k f(r_{t+k+1}, w) \right] $$
- Maximization does not do the trick!
- Heuristic with non-linear $ f $ (Van Moffaert, Drugan, Nowé (2013))
- Not guaranteed to find optimal policy, or converge

Taxonomy

<table>
<thead>
<tr>
<th>single policy (known weights)</th>
<th>multiple policies (unknown weights or decision support)</th>
</tr>
</thead>
<tbody>
<tr>
<td>deterministic stationary policy</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>one deterministic stationary policy</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
<tr>
<td>one deterministic non-stationary policy</td>
<td>Pareto coverage set of deterministic non-stationary policies</td>
</tr>
<tr>
<td>one mixture policy of two or more deterministic stationary policies</td>
<td>convex coverage set of deterministic stationary policies</td>
</tr>
</tbody>
</table>

Part 2: Methods and Applications

- Convex Coverage Set Planning Methods
  - Inner Loop: Convex Hull Value Iteration
  - Outer Loop: Optimistic Linear Support
- Pareto Coverage Set Planning Methods
  - Inner loop (non-stationary): Pareto-Q
  - Outer loop issues

Learning Scenarios

- Model-based learning (Wiering, Withagen & Drugan (2014))
Learning Methods

- **Q-Learning schema:**
  \[
  Q_{\text{new}}(s, a) = (1 - \alpha)Q_{\text{old}}(s, a) + \alpha (R(s, a, s') + \gamma V_{\text{old}}(s'))
  \]

  - CCS: Parallel Q-learning (Hiraoka, Yoshida & Mashima (2009))
  - Det. Stat. PCS: schema not applicable

- Other Decision Problems
  - Taxonomy applies to cooperative decision problems
  - Multi-objective coordination graphs / constraint optimization problems (Rollón (2008); Roijers, Whiteson & Oliehoek (2015a); Wilson, Abdul, Marinescu (2015))
  - Multi-objective POMDPs (Soh & Demeris (2011); Roijers, Whiteson & Oliehoek (2015b); Wray & Zilberstein (2015))
Other Decision Problems

- Taxonomy applies to cooperative decision problems
- Multi-objective coordination graphs / constraint optimization problems (Rollen (2008); Roijers, Whiteson & Oliehoek (2015a); Wilson, Abdul, Marinescu (2015))
- Multi-objective POMDPs (Soh & Demiris (2011); Roijers, Whiteson & Oliehoek (2015b); Wray & Zilberstein (2015))
- Multi-objective bandit problems (Drugan & Nowe (2013))

Part 2: Methods and Applications

- Convex Coverage Set Planning Methods
  - Inner Loop: Convex Hull Value Iteration
  - Outer Loop: Optimistic Linear Support
- Pareto Coverage Set Planning Methods
  - Inner loop (non-stationary): Pareto-Q
  - Outer loop issues
- Beyond the Taxonomy

Medical Applications

- Anthrax response (Soh & Demiris (2011))
  - Minimizing loss of life
  - Minimizing number of false alarms
  - Minimizing cost of investigation
- Partial observability (MOPOMDP)
- Finite-state controllers
- Evolutionary method
- Pareto coverage set

- Control system for wheelchairs (Soh & Demiris (2011))
  - Maximizing safety
  - Maximizing speed
  - Minimizing power consumption.
- Partial observability (MOPOMDP)
- Finite-state controllers
- Evolutionary method
- Pareto coverage set

3 papers at this conference!
Environmental Applications

- Environmental control of a building (Kwak et al. (2012))
  - Minimizing energy consumption
  - Maximizing comfort
- Reduce energy consumption by 30% w.r.t. old system
- While maintaining comfort
- Groups of people with different preferences
- Smart grids

Broader Application

- “Probabilistic Planning is Multi-objective” — Bryce et al. (2007)
  - The expected return is not enough
  - Cost of a plan
  - Probability of success of a plan
  - Non-goal terminal states
- Risk sensitive
  - variance
  - expected value

Broader Application

- Fairness, e.g. in traffic
  - Centralized planner in multi-agent problem
  - Each agent as a reward function
  - Pareto-optimal
  - “Robin Hood transfers” improve utility, e.g.,
    
    \[(3, 3) \succ_{L} (6, 0)\]
  - Lorenz optimality (Perny et al. (2013))
- Repeated cooperation

Closing

- Consider multiple objectives
  - most problems have them
  - a priori scalarization can be bad
- Derive your solution set
  - Pareto front often not necessary
- Promising applications
References


