Synergies between Evolutionary Algorithms and Reinforcement Learning

Madalina M. Drugan

Vrije Universiteit Brussel

Brussels, Belgium

madalina.drugan@gmail.com

http://www.sigevo.org/gecco-2015/

GECCO'15 Companion, July 11–15, 2015, Madrid, Spain.
ACM 978-1-4503-3488-4/15/07.
http://dx.doi.org/10.1145/2739482.2756582

1

Instructors

❖*Madalina M. Drugan* received the Diploma Engineer degree in Computer Science from the Technical University of Cluj-Napoca, Romania. In 2006, she received the PhD degree from the Computer Science Department, University of Utrecht, The Netherlands. She has carried out research in Machine Learning and Optimisation and related topics like Evolutionary Computation, Bioinformatics, Multi-objective optimisation, Meta-heuristics, Operational Research, and Reinforcement Learning. She has experience with research grants, reviewing services and a strong publication record in international peer-reviewed journals and conferences. During the last three years, she initiated and organised several special sessions, tutorials and workshops on Reinforcement Learning and Optimisation at IEEE WCCI, IEEE SSCI, PPSN, ESANN, CEC, IJCNN, SIMCO, EVOLVE.

2

Course agenda

- Reinforcement learning: short introduction
  - Multi-armed bandits
- Evolutionary Computation in Reinforcement Learning
  - Motivation
  - Multi-objective reinforcement learning
  - Multi-criteria decision making and reinforcement learning
- Reinforcement Learning in Evolutionary Computation
  - Online adaptive operator selection
  - Schemata bandits
- Examples
- Discussion
- Questions

3

Reinforcement Learning (RL): short introduction

[Sutton and Barto, 1998] [Wiering and van Otterlo, 2012]

- The most general on-line/off-line learning technique that includes a long-term versus a short term reward trade-off.
- Applications: game theory, robot control, control theory, operations research, etc.
- RL solves environments modelled as Markov decision processes (MDP) by rewarding good actions and punishing bad actions
- The best actions are identified by trying them out and evaluating their consequences including long term consequences which might only be apparent after a large number of other actions have been taken
- The exploration / exploitation dilemma is crucial for online RL
  - exploration refers to trying out actions of which the outcome is still uncertain
  - exploitation refers to selecting actions which have shown to be good in the past

4

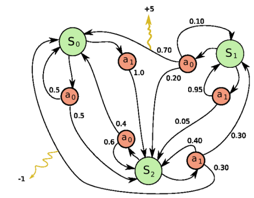
Markov decision processes (MDPs)

- *Markov decision processes* (MDP) a popular formalism to study decision making under uncertainty with the goal of maximising the long term reward intake
- An MDP is characterised with a tuple $< S, A, T, R >$
  - the *state – search space* $S = \{s_1, s_2, \ldots, s_N\}$, where $s_t \in S$
  - a set of actions $A = \{a_1, a_2, \ldots, a_N\}$ available to the agent in each state.
  - a *transition probability* $T(s, a, s')$ mapping state action pairs to a probability distribution over successor states
  - a *reward function* $R : S \times A \times S \rightarrow \mathbb{R}$ to denote the expected reward when the agent makes the transition from state s to state s' using action a.
  - r_t is the *immediate scalar reward* obtained at time t
- This process is Markovian → the distribution over the next states is independent on the past given the current state and action.

5

Markov decision processes (MDP)

- An example



- Policies can perform deterministic or stochastic action selection
- **Goal of MDPs**: to find the best policy which is the policy that receives the most rewards

6

MDPs paradigms: Dynamic programming (DP)

- Bellman equation (1957) states that the *expected value* of a state $V^{\pi}(s)$ is defined in terms of the immediate reward and the expected future rewards
  - the value of a state s under an optimal policy π^* is

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s') \left(R(s, a, s') + \gamma V^*(s') \right)$$

- Dynamic programming methods assume that
  - the transition model *P* is known → strong assumption
  - the reward function R is known → mild assumption
- DP breaks the problem up in smaller problems
- ***Model based techniques***
  - policy iteration
  - value iteration

7

Value vs policy iteration for solving MDPs

- Value iteration algorithms
  - Each iteration, the value function is updated and then a new optimal policy given the new value function is computed
  - A policy is optimal if it maximises the expected cumulative reward for any initial state
- Policy iteration algorithms
  - First fully evaluate a policy
  - Then improve the policy

8

Fundamental DP algorithms: policy iteration

- The model of the MDP is given meaning that it can be solved with dynamic programming
- Two phases:
  - policy evaluation: evaluate the value function V^{π} of a fixed policy π
    - Bellman equation is now an update rule by extending the planning horizon with one step
    - The current value function $V_k^{\pi}(s)$ is updated to $V_{k+1}^{\pi}(s)$

$$V_{k+1}^{\pi}(s) = \sum_{s' \in S} T(s, \pi(s), s') \left(R(s, \pi(s), s') + \gamma V_k^{\pi}(s)\right)$$

    - The sequence of $V_k^{\pi}(s)$ converges as k goes to infinity
  - policy improvement:
    - improves the quality of the policy π

$$\pi'(s) = argmax_{a \in A} \sum_{s' \in S} T(s, a, s') \left(R(s, a, s') + \gamma V^{\pi}(s')\right)$$

    - policy iteration

$$\pi_0 \to V^{\pi_0} \to \pi_1 \to V^{\pi_1} \to \ldots \to \pi_k \to V^{\pi_k} \to \ldots \to \pi^*$$

9

Fundamental DP algorithms: value iteration

- A simplified version of policy iteration that merges the policy evaluation and policy improvement in a single step
- The value function is updated on the fly

$$V_{k+1}(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s') \left(R(s, a, s') + \gamma V_k(s)\right)$$

- V_k converges to the optimal value V^*

$$V_0 \to V_1 \to \ldots \to V_k \to V_{k+1} \ldots \to V^*$$

- Very popular due its simplicity

Partially observable Markov decision processes (POMDPs)

- Standard MDPs → completely observable MDPs (CO-MDPs)
- POMDPs → do not have direct access to the current state

10

Online Reinforcement Learning

- ***Model free*** reinforcement learning algorithms
  - The transition model and the reward functions are not known a-priori
- Rewards classification
  - Immediate rewards returned by environment
    - Scalar values that can be stochastic
    - Reward good actions → positive rewards
    - Punish bad actions → negative rewards
  - Value function denotes cumulative rewards
  - The goal of RL is to optimise the long term reward intake $Q(s, a)$
  - Not known long-term reward → a large number of actions have to be taken in order to approximate its value since there are an infinite number of futures

11

Online RL paradigms: Q-learning

- Very popular model free RL algorithm [Watkins, 1989]
- Off-policy learning algorithm meaning that it learns a policy π while searching for the optimal policy π^*
- Q-values are estimated based on immediate rewards and taken actions
- Incrementally decreases the learning rate such that Q-values converge to the true values, exploring all actions an infinite number of times is necessary
- **ε** -greedy policy selects often the best action and with small probability another action
- Q-learning algorithm's inner loop
  - select an action a based on Q-values and an exploration strategy
  - perform action a
  - observe the new state s' and receive reward r_t

$$Q(s, a) \leftarrow Q(s, a) + \alpha \cdot \left(r_t + \gamma \cdot \max_{a' \in A(s')} Q(s', a') - Q(s, a)\right)$$

  - $s \leftarrow s'$
- Converges to the optimal policy π^*

12

Model free RL

- **SARSA**
  - On-policy algorithm that optimises the policy that is executed
    $$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\left(r_t + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)\right)$$
  - max operator in Q-learning is replaced with the action estimate according to the policy
  - converges in infinite time to the optimal policy when all states and actions were tried infinitely often and exploration decreases over time
- **QV learning**
  - On policy algorithm that uses also states values to speed up learning
    $$Q_{t+1}(s_t, a_t) = r_t + \gamma V_t(s_{t+1})$$
    $$V_{t+1}(s_t) = r_t + \gamma V_t(s_{t+1})$$
- **Temporal difference learning (TD)**
  - On policy method that learns the values of states based on estimates of other values, or *bootstrapping*

13

RL paradigms: Multi-armed bandits (MAB)

- Popular mathematical formalism used to study the convergence properties of RL with a single state
- A machine learning paradigm used to study and analyse resource allocation in stochastic and noisy environments.
- An example: a gambler faces a row of slot machines and decides
  - which machines to play,
  - how many times to play each machine
  - in which order to play them

- When played, each machine provides a reward generated from an unknown distribution specific to a machine.
- The goal of the gambler is to *maximise the sum of rewards* earned through a sequence of lever pulls.

14

Multi-armed bandits (MAB) algorithms

- Intuition on the MAB algorithms
  - An agent must choose between *N*-arms (= actions) such that the expected reward over time is maximised.
  - The algorithm starts by fairly exploring the *N*-arms, gradually focusing on the arm with the best performance.
  - The distribution of the stochastic payoff of the different arms is assumed to be unknown to the agent.
- ***Exploration / exploitation trade-off***
  - Explore the sub-optimal arms that might have been unlucky
  - Exploit the optimal arm as much as possible
- **Performance measures**
  - Cumulative regret is a measure of how much reward a strategy loses by playing the suboptimal arms

15

Multi-armed bandits: type of algorithms

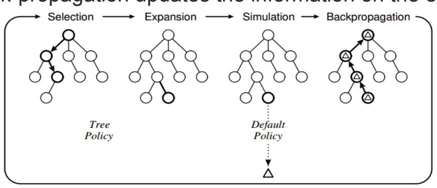
[Bubeck & Cesa-Bianchi, 2015]

- Stochastic multi-armed bandits
  - Online selection of the arm with the maximum expected mean (i.e., the arm with higher expected reward)
    - The best arm can change over time
  - Best arm identification algorithms
    - Fixed confidence vs fixed budget
    - Multiple best arm identification
- Adversarial multi-armed bandits
  - a game is played between a forecaster and an environment assuming that the adversarial process controls the rewards
- Contextual multi-armed bandits
  - uses the context to adapt the multi-armed bandit long term behaviour, or regret
- Bayesian multi-armed bandits: Thompson sampling
- Continuous multi-armed bandits: X-armed bandits
- Monte Carlo tree search: Upper confidence tree search

16

Monte Carlo Tree Search (MTCS)

- [Browne et al, 2012]
- A heuristic used to solve intractable problems, i.e. huge search spaces, like playing computer Go
- MCTS → builds a search tree using a search policy selecting the most probable nodes to expand
- A top down approach, i.e. root to leaves, with the following steps
  - Selection of the most promising children
  - Expansion creates new nodes using a tree policy
  - Simulation plays at random from the current node to the end of the game
  - Back-propagation updates the information on the explored path



17

Exploration / exploitation trade-off

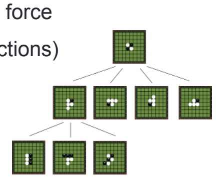
- Model free RL
  - exploration means try different actions to see their results
  - exploitation means to exploit the knowledge about good actions
- Monte Carlo Tree Search
  - exploration means generate new branches in the tree
  - exploitation means to focus the search on the tree branches that returned good rewards
- Multi-armed bandits
  - exploration of suboptimal arms
  - exploitation means to pull often optimal or close to optimal arms
- Evolutionary multi-objective optimisation
  - exploration of unknown regions of the search space
  - exploitation of good parts of the solution space

18

Real-world applications for reinforcement learning

- Playing online games:
  - Atari games (Pacman)
    - generative RL using Bayesian rules
    - QV - learning
    - Monte Carlo Tree Search

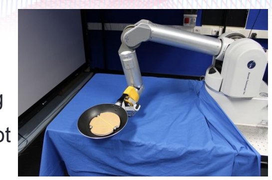
  - Go games --> Monte Carlo Tree Search using Multi-armed bandits
  - (Chess can be efficiently played with brute force
  - MinMax methods and good evaluation functions)



  - Othello, Pacman
  - Mazes --> Q-learning

Real-world applications for RL

- Robotics
  - Vision with model-based RL
  - Control of a robot arm with Q-learning
  - POMDPs for navigating a mobile robot



- Traffic light control
  - Model based RL and game theory
  - Multi-agent systems in
    - traffic distribution in a network

- *RL is considered slow and resource consuming compared with other heuristics but can learn online under changing environmental conditions*

20

Multi-criteria decision making (MCDM)

- Assumes the involvement of a decision maker (DM) → the main difference with multi-objective optimisation
- Classification of MCDM methods
  - Pareto-based ranking → DM is indifferent
    - Dominance relation → a partial order relation where two solutions can be better in one objective and worse in another objective
  - Scalarized relation → DM prefers a region of the search space
    - A function that transforms the value vector of a solution in a scalar value
  - Preference ranking → DM expresses the preference for some objectives
    - A utility function or a lexicographic order to rank the objectives
  - Interactive methods → DM permanently interacts with the environment
    - Games

Pareto dominance relation

- *A reward vector can be better than another reward vector in one objective and worse in another objective*
- The natural order relationship for multi-objective search spaces
- Examples of relationships between reward vectors

relationship	notation	relationships
μ_1 dominates μ_2	$\mu_2 \prec \mu_1$	$\exists j, \mu_2^j < \mu_1^j$ and $\forall o, j \neq o, \mu_2^o \leq \mu_1^o$
μ_1 weakly domin μ_2	$\mu_2 \preceq \mu_1$	$\forall j, \mu_2^j \leq \mu_1^j$
μ_1 is incomp with μ_2	$\mu_2 \| \mu_1$	$\mu_2 \nprec \mu_1$ and $\mu_1 \nprec \mu_2$
μ_1 is non-domin by μ_2	$\mu_2 \nprec \mu_1$	$\mu_2 \prec \mu_1$ or $\mu_2 \| \mu_1$

- *The Pareto front is* the set of expected reward vectors that are *non-dominated* by the other expected reward vectors
- All the solutions in the Pareto front are considered equally important

22

Lp scalarization function

- *Goal*: Lp transforms the multi-objective search space into a single objective space using a scalarization function
- Weighted power p sums of reward values, where a set of predefined weights is considered

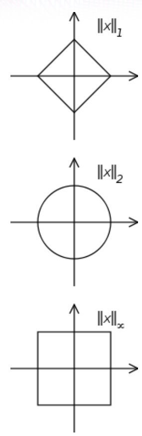
$$f_p(\mu_i) = \sqrt[p]{\sum_{j=1}^{D} \omega^j \cdot (\mu_i^j - z^j)^p}$$

L_p function can find all solutions of any shape, i.e. non-convex

The reference point $\mathbf{z} = (z^1, \ldots, z^D)$ is an extra parameter

L_1 function is *a linear scalarization* function

L_∞ function is *a Chebyshev scalarization* function



23

Evolutionary Computation into RL

- Evolutionary Multi-objective Optimisation
- Multi-criteria decision making (MCDM)
- Learning in games
- Evolutionary Algorithms for Reinforcement Learning
- Multi-objective dynamic programming (MODP)
- Multi-objective reinforcement learning (MORL)
- Online learning in multi-objective games
- Multi-objective multi-armed bandits

24

Evolutionary Algorithms for RL [Moriarty et al, 1999]

- EARL evolves policies with EAs
  - Generate new policies using EA operators
  - Associate each policy a fitness value
  - Select the best policies to produce the next generation
- Policy representation
  - Rule-based policy representation
    - each gene is a condition-action rule that maps a set of states to an action
    - distributed rule-based representation of a policy over several EAs evolved separately for learning classifier systems (LCS)
  - Parameter representation for evolving neural networks
    - each gene is a weight in a neural network
    - distributed network based policies constructs different parts of a neural network that optimise different tasks using EAs

25

Evolutionary Algorithms for RL

- Fitness and credit assignment
  - the agent interacts with an environment
  - the fitness values are averaged over time
  - what is the effect of current action vs past actions on the current reward for sparse pay-offs like reaching the goal with a robot
  - subpolicy credit assignment for distributed policies
- Selection
  - roulette selection proportional with fitness values
- Genetic operators are specific for each policy representation
  - triggered operators for learning classifier systems (LCS)
  - real coded operators for strings of weights for neural networks

26

Evolutionary Algorithms for RL

- Strengths of EARL
  - scaling up to large state spaces
    - policy generalisation is grouping together states for which the same action is required
      - may vary considerable with the rules they encode
      - level of abstraction is higher than for a normal policy
    - policy selectivity means the knowledge about bad decisions is not represented
      - the search is reduced by focusing on promising actions
  - non-stationary environments
    - tracking in non-stationary environments
    - a statistical model of agent uncertainty
  - incomplete state information
    - reward policies that avoid the ambiguous states
    - model hidden states

27

Evolutionary Algorithms for RL

Many variants of EARL algorithms improve state of the art RLs

- CMA-ES strategies are used to select RL policies [Heidrich-Meisner & Igel, 2009] or to evolve neural networks
- Neuroevolution is used to evolve neural networks [Whiteson & Stone, 2006]
- Representation of phenotype - genotype space in EAs is used to describe populations of single weights for evolving neural network controllers [Gomez et al, 2009]

28

Multi-objective reinforcement learning (MORL)

- Early research (80s): multi-objective dynamic programming and multi-objective MDPs
- Main differences with MDPs
  - Immediate reward values are replaced with reward vectors
  - A set of optimal policies of incomparable quality in all objectives
  - Computationally intractable even for simple small environments
- Techniques from MOO and MCDC are incorporated in RL with reward vectors
- Goal: to identify one or a set of Pareto optimal policies
- When compared with RL, MORL has:
  - extra computational challenges
  - different exploration / exploitation trade-offs
  - more complicated experimental setups

29

Evolutionary multi-objective optimisation (EMO) in RL

Multi-objective Markov Decision Processes (MOMDPs)

- compute all Pareto optimal policies
- tuples of rewards instead of a single reward
- stationary and non-stationary deterministic environments

Multi-objective Reinforcement Learning (MORL)

- important differences with single objective reinforcement learning
  - several actions can be considered to be the best according to their reward tuples.
  - techniques from EMO should be used in the multi-objective RL framework to improve the exploration/exploitation trade-off
  - complex and large multi-objective environments.

Multi-objective multi-armed bandits (MOMABs)

- single state reinforcement learning algorithm
- various variants of multi-armed bandits extended to reward vectors

30

Multi-objective Markov decision processes (MOMDPs)

[White, 1982] [Wiering and De Jong, 2007][Lizotte et al, 2012][Roijers et al, 2013][Wiering et al, 2014][Parisi et al, 2014]

- Multi-objective dynamic programming (MODP)
  - A reward vector $r = (r_1, r_2, \ldots, r_m)$ with m dimensions
  - A reward function $R(s, a, s') = (R_1(s, a, s'), \ldots, R_m(s, a, s'))$
  - A set of Pareto optimal policies
  - An agent will select one or more Pareto optimal policies
  - Value or policy iteration multi-objective dynamic programming
- Pareto dominance relation or scalarization functions are used to compute and track the Pareto optimal policies
- For both non-stationary and deterministic environments
- The corresponding algorithms converge to the unique Pareto optimal set of policies

31

Value iteration multi-objective dynamic programming

- [Wiering and de Jong, 2007][Wiering et al 2014]
- The value function is a vector $V^i(s) = (V_1^i(s), V_2^i(s), \ldots, V_m^i(s))$
- The set of non-dominated value functions is denoted with

$$V^O(s) = \{V^i(s) \mid V^i(s) \text{ isn't dominated by a policy in } s\}$$

- The set of non-dominated Q-values

$$Q^O(s,a) = \{Q^i(s,a) \mid Q^i(s,a) \text{ isn't dominated in } s,\ a\}$$

- The non-dominated operator tells whether a policy i is non-dominated in the state s by any value function in $V^D(s)$
- $$ND(V^i(s), V^D(s)) \leftrightarrow \nexists V^j(s) \in V^D(s);\ \ V^j(s) \succ V^i(s)$$
- The Pareto optimal operator

$$PO(Q^D(s,a)) = \{Q^i(s,a) \mid Q^i(s,a) \in Q^D(s,a) \wedge ND(Q^i(s,a), Q^D(s,a))\}$$

- The dynamic programming operator for deterministic environments

$$DP(Q^D(s,a)) = \{R(s,a,s') \oplus \gamma V^O(s') \mid P(s,a,s') = 1.0\}$$

32

Value iteration multi-objective dynamic programming

- The optimal set of Q – values vectors Q^{O^*}

$$Q^{O^*} = (PO(DP(Q_0)))^*$$

- where * means the operator is repeated an infinite number of times
- This dynamic programming algorithm *converges* to the maximal Pareto optimal set of policies because it eliminates all dominated policies
- *The set of Pareto optimal policies is independent of the usage of the Pareto optimal operator inside the dynamic programming operator*

$$PO(DP'(Q_0))^* = (PO(DP(Q_0)))^*$$

- where DP' does not make use of the PO operator at all
- $DP'(Q^D(s,a)) = \{R(s,a,s') \oplus \gamma V^D(s') \mid P(s,a,s') = 1.0\}$

33

Multi-objective RL (MORL) algorithms

- Pareto Q-learning algorithm applies a Pareto operator for sets to keep track of Pareto optimal policies [van Moffaert et al, 2014]
- Scalarization based RL [Vamplew et al, 2010][Brys et al, 2014] [van Moffaert et al, 2013a]
  - Weighted linear scalarization functions (most algorithms)
  - Goal: identifying all Pareto optimal policies on a convex front
- Hyper-volume unary indicator is used to measure the quality of solutions [Wang & Sebag, 2013] [van Moffaert et al, 2013b]
  - Another function that transforms the multi-objective search space
- Preference based dominance relations are used with multi-criteria RL [Gabor et al, 1998]
  - Only a region on the Pareto front is required
  - Interaction with the user to select the preferred solutions

34

Multi-objective Q-learning

- In MORL, *Q*- values are extended to ***Q***-vectors, one value for each objective

$$\mathbf{Q}(s,a) = (Q^1(s,a), \ldots, Q^M(s,a))$$

- A set of linear scalarization functions to identify convex fronts
- L_∞ norm identifies solutions on any Pareto front
- Q-values are updated separately in each objective → convergence properties inherited from Q-learning
- The selection of next actions depends on the used scalarization function
- Adaptive scalarization functions adapt the weights of the linear scalarization functions
- The main differences with single objective RL
  - ***Pareto front*** includes several reward values that are equally good (incomparable) and a set of ***Q***-vectors with incomparable best values.
  - ***Updating rules*** for the ***Q***-vectors and the selection of the best next action when compared with single objective MDPs.

35

Scalarized multi-objective Q-learning

- Multi-objective MDPs → immediate reward vectors

$$R(s,a) = (R^1(s,a), R^2(s,a), \ldots, R^m(s,a))$$

- Q values

$$Q(s,a) = (Q^1(s,a), Q^2(s,a), \ldots, Q^m(s,a))$$

- Scalarized Q-values

$$SQ(s,a) = \sum_{o=1}^{m} w^o Q^o(s,a)$$

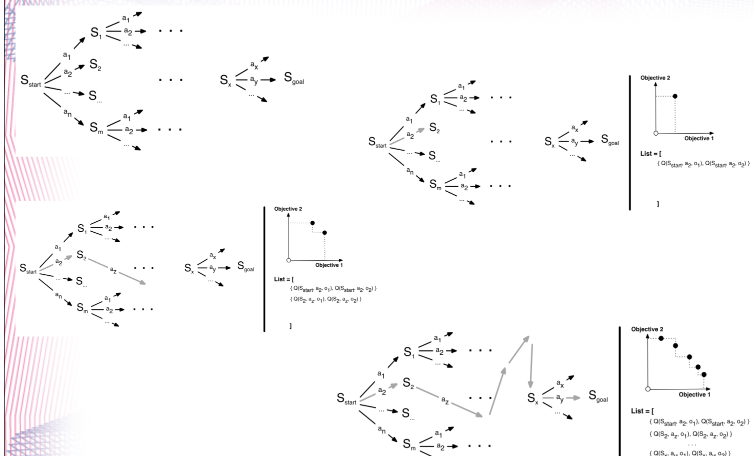
- ε -greedy selects the action a with the largest SQ value with a high probability
- Try action a
- For each objective o

$$Q(s,a,o) \leftarrow Q(s,a,o) + \alpha \cdot [r(s,a,o) + \gamma Q(s', max_{a'}, o) - Q(s,a,o)]$$

- s ← s'

36

A scalarized Q-learning algorithm

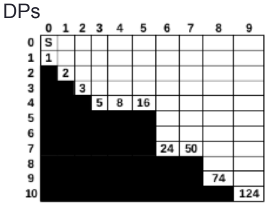


37

Model based multi-objective reinforcement learning

[Wiering et al, 2014]

- Most MORL are model free algorithms that use scalarization functions to transform the reward vectors into reward values
- Model based MORL
  - estimate the model of the environment
    - using frequencies stored in a table
  - solve this model
    - based on value iteration multi-objective DPs
- Exploration strategies
  - least visited exploration
    - counts the times each action was taken
  - random exploration
    - actions are selected randomly



38

Hypervolume based MORL

- [Wang & Sebag, 2013]
- Multi-objective Monte Carlo tree Search (MOMCTS)
- Hypervolume unary indicator is used to select a node in MOMCTS
- MOMCTS has inherited computational problems for hypervolume unary indicator
- MOMCTS is also combined with Pareto dominance to improve the diversity of solutions
- MOMCTS performs better than scalarized MORL on the bi-objective Deep Sea environment

- [van Moffaert et al, 2013b]
- Multi-objective Q-learning uses hypervolume indicator to evaluate the quality of the Pareto front
- Action selection mechanism that maximises the hypervolume indicator
- Compared with scalarization based MORL, hypervolume based MORL has
  - better performance
  - narrower Pareto front

39

Real-world applications for MORL

Many real-world problems are inherently multi-objective and were tackled with single objective techniques and predefined linear scalarization problems

- Traffic light control --> linear scalarized multi-objective Q-learning
- Control problems (like wet clutch) with two or more objectives --> adaptive linear scalarized MORL
- A benchmark or RL problems that were transformed into multi-objective environments
  - Mounting car problems
  - Maze like problems, i.e. Deep Sea World
  - Simple multi-objective MDPs --> these problems were approached with both scalarized and Pareto MORL
- Preference based MORL that requires user interactions --> scalarized MORL

40

Multi-objective multi-armed bandits (MOMABs)

- [Drugan & Nowe, 2013]
- Multi-armed bandits use reward vectors
- Evolutionary Computation (EC) techniques are used to design computationally efficient MOMABs
- The exploration / exploitation trade-off is common for both multi-armed bandits (MABs) and EC for multi-objective optimisation
  - In EC, exploration means evaluation of new solutions in a very large search space where states cannot be enumerated
  - In MAB, exploration means to pull arms that have suboptimal mean reward values
  - In EC, exploitation means to focus the search in promising regions where the global optimum could be located
  - In MAB, exploitation means to pull the currently identified best arm(s)
- MOMABs with a finite set of arms and reward vectors generated from stochastic distributions

41

Multi-objective multi-armed bandits (MOMABs)

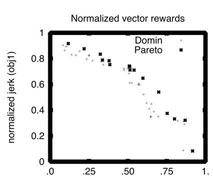
- The goal of MOMABs is to maximise the returned reward; or to minimise the regret of pulling suboptimal arms
- We assume that all Pareto optimal arms are equally important and need to be identified
- Performance measures
  - Pareto regret → sum of the distances between each suboptimal arm and the Pareto front
  - Variance regret → variance in using the Pareto optimal arms
  - KL divergence measure
- Theoretical analysis
  - Upper and lower bounds on expected cumulative regret
- Challenges
  - Large and complex *stochastic* multi-objective search spaces
  - Non-convex Pareto fronts

42

The bi-objective transmission problem of wet clutch

- *Wet clutch:* an application from control theory
- *Goal:* optimise the functionality of the clutch:
  - the optimal *current profile* of the electro-hydraulic valve that controls the pressure of the oil to the clutch
  - the *engagement time*.

- *Stochastic output data* —> some external factors, such as the surrounding temperature, cannot be exactly controlled.
- *Goal: optimise the parameters* —> that minimise the clutch's profile and the engagement time in varying environmental conditions.



43

Stochastic discrete MOMAB problems

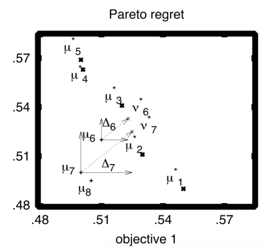
- *K*-armed bandit, $K \geq 2$, with independent arms
- The reward vectors have *D* –objectives, where *D* fixed
- An arm *i* is played at time steps $t_{1,i}, t_{2,i}, \ldots$
- The corresponding *reward vectors* $X_{i,t_1}, X_{i,t_2}, \ldots$ are independently and identically distributed according to an unknown law with unknown expectation vectors
- *The goal of MOMAB:*
  - Identify the set of best arms by simultaneously maximising rewards in all objectives
  - The arms in the Pareto front are considered equally important and should be pulled the same number of times.
  - Minimise the regret (or the loss) of not selecting the arms in the Pareto front

44

Performance metric: Pareto regret

- We denote with $\Delta_i = \|\nu_i^* - \mu_i\|_2$ the empirical distance between an arm and the Pareto front
- Let ν_i^* be the *virtual reward vector* of the arm i such that μ_i has the minimum distance to ν_i^*,
  - $\nu_i^* = \|\mu_i - \epsilon_i\|_2$ is incomparable with all reward vectors in the Pareto front
- The *expected Pareto regret* for a learning algorithm after n arm pulls is

$$\mathbb{E}[R_n] = \sum_{i=1}^{K} \Delta_i \cdot \mathbb{E}[T_i(n)]$$



Pareto MAB algorithms

- ***Definition:*** a multi-objective MAB algorithm that uses the *Pareto partial order relationship*
- The Pareto regret metric is used to upper bound the performance of the designed Pareto MAB algorithms
- *Challenges* in designing Pareto MAB algorithms:
  1. Pareto front identification
     1. Identification of a representative Pareto set of arms
  - The exploitation/exploration trade-off:
    - Exploration: pull suboptimal arms that might be unlucky
    - Exploitation: pull as much as possible the optimal arms
  - Optimising the performance of Pareto MABs in terms of upper and lower bounds on expected and/or immediate regret
  - Ameliorate the performance of Pareto MABs for large sets of arms

Pareto Upper Confidence Bound (PUCB1)

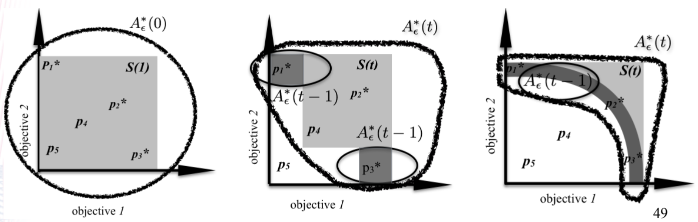
- Straightforward generalisation of UCB1
  - ✤operator selection [Fialho et al, 2009]
  - ✤learning the utility of swap operations in combinatorial optimisation [Puglierin et al, 2013]
- Maximises the reward index $\widehat{\boldsymbol{\mu}}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|\mathcal{A}^*|})}{n_i}}$
- The algorithm
  - Each iteration, a Pareto front is calculated using $\widehat{\boldsymbol{\mu}}_h + \sqrt{\frac{2\ln(n\sqrt[4]{D|\mathcal{A}^*|})}{n_h}} \succ \widehat{\boldsymbol{\mu}}_i + \sqrt{\frac{2\ln(n\sqrt[4]{D|\mathcal{A}^*|})}{n_i}}$
  - One of the arms from the Pareto front is selected
- The upper bound is $\sum_{i \notin \mathcal{A}^*} \frac{8 \cdot \log(n\sqrt[4]{D|\mathcal{A}^*|})}{\Delta_i} + (1 + \frac{\pi^2}{3}) \cdot \sum_{i \notin \mathcal{A}^*} \Delta_i$
- The worst-case performance of this algorithm is when the number of arms *K* equals the number of optimal arms
- The algorithm reduces to the standard UCB1 for *D = 1*.
- Pareto UCB1 performs *similarly* with the standard UCB1 for a small number of objectives and small Pareto optimal sets

Pareto front identification

- This policy is an extension of the *best arm identification algorithm* [Audibert et al.,2010] for a set of arms of equal quality.
- The *m*-best arm identification algorithm assumes that the *m*-best arms can be totally ordered.
- ***The algorithm***
- Let $A_1 = \{1, \ldots, K\}\,,\ \overline{\log}(K) = \frac{1}{2} + \sum_{i=2}^{K} \frac{1}{i}\,,\ n_0 = 0 \text{ and for } k \in \{1, \ldots, K-1\}$
- $n_k = \left\lceil \frac{\log(D|\mathcal{A}^*|)}{\overline{\log}(K) + \log(D|\mathcal{A}^*|)} \cdot \frac{n-K}{K+1-k} \right\rceil$
- For all rounds $k = 1, 2, \ldots, K-1$
- (1) For each arm $i \in A_k$, select it for $n_k - n_{k-1}$ rounds
- (2) Let $A_{k+1} = A_k \setminus \text{argmin}_{i \in A_k} \widehat{\mathbf{x}}_{i,n_k}$ the arm to dismiss in this round
- Let the remaining set of arms be the Pareto optimal set of arms $\mathcal{A}^*$

Annealing Pareto Knowledge gradient [Yahyaa et al, 2014]

- Knowledge gradient policy is a reinforcement learning algorithm where the reward vectors are updated using Bayesian rules
- Annealing like functions that decrease uncertainty around the arms
- The algorithm
    - At initialisation, all arms are considered
    - Iteratively, extreme arms are identified as either Pareto optimal or deleted as suboptimal arms
    - The iteration stops when there are no more arms to classify



49

Challenges in designing scalarized MOMABs

- [Drugan & Nowe, 2014] [Drugan, 2015a][Drugan, 2015b]
- Identify the entire Pareto front
    - Large Pareto fronts
    - Non-convex Pareto fronts
    - Non-uniform distributions of arms on the Pareto front
- Optimising the performance of scalarized MOMABs in terms of upper and lower regret bounds
    - The *scalarized* / Pareto regret metric
    - The Kullback-Leibler divergence regret metric
- Exploitation/exploration trade-off:
    - Exploration: *sample scalarization* functions, and pull arms that might be unluckily identified as suboptimal
    - Exploitation: pull as much as possible the Pareto optimal arms of *relevant scalarization* functions

50

Reinforcement learning in EC

- Adaptive operator selection for Evolutionary Computation
    - Online parameter selection as opposite to off-line parameter selection
    - Uses reinforcement learning to select operators
        - Adaptive pursuit →pursuit more often the operator that improves the most the results
        - Multi-armed bandits like UCB1 to adaptively select the best operator
        - SARSA
    - Applied in tuning the parameters of
        - Evolutionary Computation (Genetic algorithms, Evolution Strategies)
        - Iterated (Pareto) local search
        - Evolutionary Multi-objective optimisation
- Schemata bandits
    - Monte Carlo decision trees for schemata theory
- Monte Carlo Tree Search for learning in continuous search spaces

51

Adaptive operator selection

- Motivation:
    - the performance of EAs depends on the used parameters
    - the performance of a genetic operator depends on the landscape
    - an operator can have different performance in different regions of the landscape
- Tuning genetic operators
    - Selection of parameters
    - Mutation rates / Recombination exchange rates
    - Population size
    - Variable neighbourhood size (local search)
- Online learning strategy
    - The algorithms should learn relatively fast the best operator
    - There are several operators that perform similarly

52

Adaptive pursuit strategy (AP) [Thierens, 2005]

- Each operator i has associated a probability value $P_i^{(t)}$ of selection and an estimated reward value $Q_i^{(t)}$
- Online operator selection algorithm with fixed target probabilities is a step like distribution $D = \{p_M, p_m, \ldots, p_m\}$
  - p_M has a large probability value to select often the best operator
  - p_m has a small non zero probability to select any suboptimal operator
- The iterative algorithm
  - Pursuit with probability $P_v^{(t)}$ the operator *v* with the maximal estimated reward $Q_v^{(t)}$
  - Get reward vector $R_v^{(t)}$ for the operator *v*
  - Update reward value $Q_v^{(t)}$ using the immediate reward $R_v^{(t)}$
  - High rank the estimated reward distribution $Q_v^{(t)}$ and set the values in vector r
  - For each operator i, update the selection probabilities

$$P_i^{(t+1)} \leftarrow P_i^{(t)} + \beta \cdot (D_{r[i]} - P_i^{(t)})$$

53

UCB1 for online operator selection [Fialho at al, 2010]

- Each operator is considered an arm with unknown probability of getting a reward
- The reward function for operator i contains
  - the estimated value for the operator $\widehat{\mu}_i$
  - the exploitation coefficient $C\sqrt{\frac{2\log\sum_j n_j}{n_i}}$

where n_i is the number of times the operator *i* was selected and *C* the exploration constant

- Remarks
  - Originally, UCB1 has positive sub-unitary values
  - Setting up *C* is important for any fitness landscape
  - UCB1 detects changes in the environment but will react quite slow to them
  - UCB1 is combined with other optimisation techniques to improve the performance of the online operator selection algorithm

54

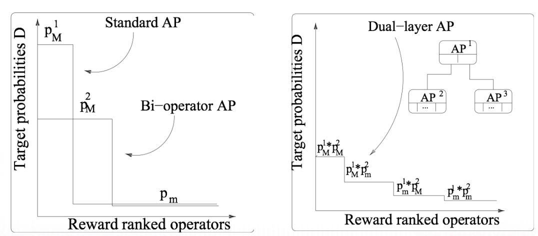
UCB1 for online operator selection

- Performance of operator selection depends on
  - the improvement measure considered like difference in fitness value and / or diversity
- Techniques to improve the performance of UCB1
  - Detect a change in the distribution with Page-Hinkley statistical tests
  - Weigh the operators using their frequency in applying it
  - Area under curve is also used as a measure of improvement in UCB1
  - Extreme values operator selection focuses on extremes to encourage exploration
- Hyper-parameter tuning, or tuning the tuner
- Off-line parameter tuning with F-race
- UCB1 is used to select solutions that adapt the CMA-ES matrix in continuous MO-CMA-ES [Loshchilov et al, 2011]

55

Generalised adaptive pursuit [Drugan & Thierens, 2011]

- An umbrella technique that could describe the main online adaptive selection algorithms
- Any target operator probabilities D can be considered, i.e. static or time dependent
- We assume that exploiting a set of related operators is beneficial for the performance of the algorithm
- We can consider two high value probabilities instead of one high value probability in D
- We consider multiple layers of adaptive pursuit algorithms



56

Online multi-operator selection [Drugan & Talbi, 2014]

- Optimise the usage of two or more operators simultaneously
- Motivated by the quadratic assignment problem:
  - Exploring large variable neighbourhoods is expensive
  - Iterated local search is efficient for QAPs
- ***Probability distribution*** of the mutation and the neighbourhood operators

$$\mathbf{P}_{\mathcal{N}}^{(t)} = \{\mathbf{P}_{11}^{(t)}, \ldots, \mathbf{P}_{1K}^{(t)}\}, \quad \mathbf{P}_{\mathcal{M}}^{(t)} = \{\mathbf{P}_{21}^{(t)}, \ldots, \mathbf{P}_{2P}^{(t)}\}$$

$$\mathbf{Q}_{\mathcal{N}}^{(t)} = \{\mathbf{Q}_{11}^{(t)}, \ldots, \mathbf{Q}_{1K}^{(t)}\}, \quad \mathbf{Q}_{\mathcal{M}}^{(t)} = \{\mathbf{Q}_{21}^{(t)}, \ldots, \mathbf{Q}_{2P}^{(t)}\}$$

- ***Quality distribution*** of the mutation and the neighbourhood operators

$$\mathcal{Q}_{\mathcal{N}}^{(t)} = \frac{\#\ improv\ of\ v_{\mathcal{N}}}{\#\ trials\ of\ v_{\mathcal{N}}}, \quad \mathcal{Q}_{\mathcal{M}}^{(t)} = \frac{\#\ improv\ of\ v_{\mathcal{M}}}{\#\ trials\ of\ v_{\mathcal{M}}}$$

- ***Update reward vectors***: an improvement in the cost of the candidate solution when compared with the current solution

$$\mathcal{P}_{\mathcal{N}i}^{(t+1)} \leftarrow \mathcal{P}_{\mathcal{N}i}^{(t)} + \beta \cdot (\mathcal{D}_{r_{\mathcal{N}i}} - \mathcal{P}_{\mathcal{N}i}^{(t)}), \quad \forall 1 \leq i \leq P$$

$$\mathcal{P}_{\mathcal{M}i}^{(t+1)} \leftarrow \mathcal{P}_{\mathcal{M}j}^{(t)} + \beta \cdot (\mathcal{D}_{r_{\mathcal{M}j}} - \mathcal{P}_{\mathcal{M}j}^{(t)}), \quad \forall 1 \leq j \leq K$$

- ***Update probabilities***: the probability distributions are independently updated

57

Generic Parameter Control with RL [Karafotias et al, 2014]

- On fly parameter tuning of parameters of EA algorithms based on
  - Diversity (phenotypic and genotypic diversity)
  - Improvement (fitness variation and improvement and stagnation counter)
- SARSA is used by the parameter control algorithm
  - States are represented in a binary decision tree
  - Actions set each parameter to a certain value
  - Each pair state-action has associated a value to determine how much impact this pair has on the estimated reward
- Evolution Strategy applied on
  - population size
  - the generation gap (the ratio of offsprings)
  - mutation step size
  - tournament size for survivor selection
- CMA-ES

58

Schemata bandits [Drugan et al, 2014]

- A hierarchical bandit where each arm is a schemata
- A synergy between Schemata Theory and Monte Carlo tree Search
- Genetic algorithms that do not use the genetic operators to generate new individuals
- Current version is computationally challenging
- The schemata with the maximal estimated mean fitness is selected the most often using an UCB1 algorithm
- A schemata is a L – dimensional hypercube, 2L binary strings

0011111001

0111011001 *0001111010*

0101101010 **0**1**10**** *0101011001*

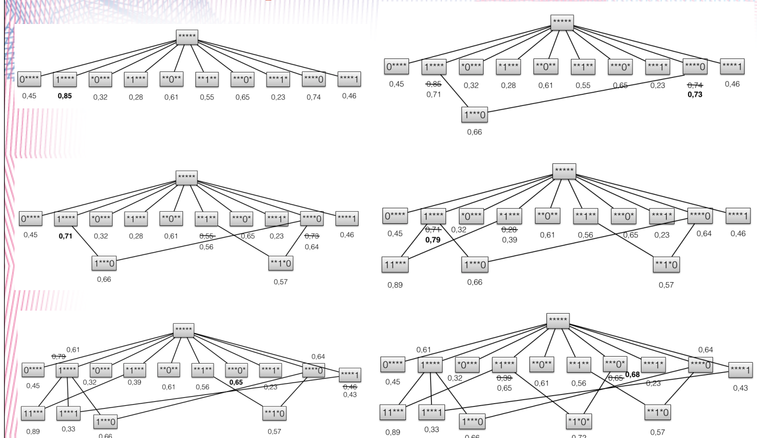
0111111011 *0101001000*

0011101001 *0001001010*

0001001001

59

An example of schemata net



60

Schemata bandits

- Schemata are generated from good solutions
- Random solutions are generated from a schemata node
- The mean of solutions represents the value of the schemata

- Schemata net structure
    - 3L schemata of the form $H \in \{0, 1, *\}$
    - o(H) – the order of schemata is the number of 0s or 1s
    - d(H) – the dimension of the schema H (number of * symbol)
  - *root* - the most general schema **…*,
  - *the leaves* – each * is replaced with 0 or 1
  - Each *node has*:
    - a value that is the mean fitness of the individuals belonging to the schema
    - 2 * d(H) children replace a * symbol with 0 or 1
    - o(H) parents replace of 0 or 1 with * symbol

61

A baseline schemata algorithm

- Initialise n random individuals
- Repeat
  - Select the root schemata
  - Select the most promising child of the current schemata using UCB1

$$argmax_i \overline{f}(H_i) + C \cdot \sqrt{\frac{2\log(t)}{t_i}}$$

  - Update counters
  - Back-propagate the information to update the value of schemata nodes

  - Parameter free optimisation algorithm
  - Schemata net is densely connected
    - computationally infeasible for large L
    - expand only a part of the schemata net
  - hybrid between the two approaches

62

Bandit trees for real coded optimisation

Monte Carlo tree search variants are used in optimisation of real-coded multi-dimensional functions

- The search space is partitioned in subdomains
- Each node in MCTS contains a multi-dimensional domain
- The search focuses on the most promising partitions, i.e. that contain the best solutions
- The other regions are explored with small probability

- Simultaneous optimistic optimisation (SOO) [Preux et al, 2014] is successfully applied on many dimensional test problems from the CEC'2014 competition on single objective real-parameter numerical optimisation.
- Hierarchical CMA-ES solver [Drugan, 2015c] uses CMA-ES solvers in each node of MCTS

63

Concluding remarks on MORL algorithms

- Multi-objective reinforcement learning
  - Follows closely the latest developments in RL and MOO, but also MCDM
- Multi-objective multi-armed bandits
  - New theoretical tools needed to study the performance of MORL algorithms
- Open research questions
  - Computationally efficient exploitation / exploration trade-off
  - Adequate performance measures for MORL and MOMABs
  - Advanced MOO and MCDM techniques to improve the performance of MORL and MOMAB algorithms
  - Challenging real world problems to motivate MORL and MOMABs paradigms

64

Concluding remarks on EC using RL

- Most EC algorithms use model free RL or multi-armed bandits techniques for parameter tuning
- Schemata theorem as initially used in association with multi-armed bandits by [Holland, 1975]
- Learning Classifier Systems [Holland, 1975] is used with RL to optimise and learn in complex non-stationary environments
- Variants of Monte Carlo Tree Search are used to optimise multi-dimensional real world problems
- Hyper-heuristics use reinforcement learning to select the best heuristic for a given task [Ozcan et al, 2010]
- Pareto Local search is used in combination with RL for optimising in multi-objective search spaces [Inja et al, 2014]

65

Conclusions

- Hybrid algorithms between reinforcement learning and evolutionary computation
  - perform better than many standard settings of both algorithms
  - can represent realistic models of problems in, for example, engineering and management
    - incomplete observations
    - large stochastic and changing environments
  - new methodological and theoretical challenges
    - in reinforcement learning, the convergence proofs need to take in account the multiple dimensions and the possible interactions between them
    - in evolutionary computation, some performance metrics to measure the adaptability of the algorithm needs to be considered
  - potential to develop new algorithms for automatic parameter tuning

66

References

[Sutton & Barto, 1998] Richard S. Sutton, Andrew G. Barto: Reinforcement Learning: An Introduction. IEEE Transactions on Neural Networks 9(5): 1054-1054 (1998)

[Wiering & van Otterlo, 2012] Marco A. Wiering, Martijn van Otterlo: Reinforcement Learning: State-of-the-Art, Springer, 2012

[Watkins & Dayan, 1992] Christopher J. C. H. Watkins, Peter Dayan: Technical Note Q-Learning. Machine Learning 8: 279-292 (1992)

[Wiering & Hasselt, 2009] Marco A. Wiering, Hado van Hasselt: The QV family compared to other reinforcement learning algorithms. ADPRL 2009: 101-108

[Auer et al, 2002] Peter Auer, Nicolò Cesa-Bianchi, Paul Fischer: Finite-time Analysis of the Multiarmed Bandit Problem. Machine Learning 47(2-3): 235-256 (2002)

[Bubeck & Cesa-Bianchi, 2012] Sébastien Bubeck, Nicolò Cesa-Bianchi: Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. Foundations and Trends in Machine Learning 5(1): 1-122 (2012)

[Browne et al, 2012] Cameron Browne, Edward J. Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, Simon Colton: A Survey of Monte Carlo Tree Search Methods. IEEE Trans. Comput. Intellig. and AI in Games 4(1): 1-43 (2012)

[Moriarty et al, 1999] David E. Moriarty, Alan C. Schultz, John J. Grefenstette: Evolutionary Algorithms for Reinforcement Learning. J. Artif. Intell. Res. (JAIR) 11: 241-276 1999

[Heidrich-Meisner & Igel, 2009] Verena Heidrich-Meisner, Christian Igel: Uncertainty handling CMA-ES for reinforcement learning. GECCO 2009: 1211-1218

[Whiteson & Stone, 2006] Shimon Whiteson, Peter Stone: On-line evolutionary computation for reinforcement learning in stochastic domains. GECCO 2006: 1577-1584

[Gomez et al, 2009] Faustino J. Gomez, Julian Togelius, Jürgen Schmidhuber: Measuring and Optimizing Behavioral Complexity for Evolutionary Reinforcement Learning. ICANN (2) 2009: 765-774

67

[White, 1982] D. J. White: The set of efficient solutions for multiple objective shortest path problems. Computers & OR 9(2): 101-107 (1982)

[Wiering & de Jong, 2007] Marco Wiering and Edwin D. de Jong. Computing Optimal Stationary Policies for Multi-objective Markov Decision Processes. IEEE ADPRL 2007, pp. 158-165.

[Lizotte et al, 2012] D. J. Lizotte, M. Bowling, and S. A. Murphy. Linear fitted-q iteration with multiple reward functions. Journal of Machine Learning Research, 13:3253-3295, 2012.

[Roijers et al, 2013] Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, Richard Dazeley: A Survey of Multi-Objective Sequential Decision-Making. J. Artif. Intell. Res. (JAIR) 48: 67-113 (2013)

[Wiering et al, 2014] Marco A. Wiering, Maikel Withagen, Madalina M. Drugan: Model-based multi-objective reinforcement learning. ADPRL 2014: 1-6

[Parisi et al, 2014] Simone Parisi, Matteo Pirotta, Nicola Smacchia, Luca Bascetta, and Marcello Restelli. Policy gradient approaches for multi-objective sequential decision making. In IJCNN 2014, IEEE

[Vamplew et al, 2011] Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, Evan Dekker: Empirical evaluation methods for multi-objective reinforcement learning algorithms. Machine Learning 84(1-2): 51-80 (2011)

[van Moffaert et al, 2014] Van Moffaert, K., Drugan, M. M., & Nowé, A.. (2014). Learning Sets of Pareto Optimal Policies. AAMAS *-Adaptive Learning Agents Workshop (ALA)*

[Brys et al, 2014] Tim Brys, Anna Harutyunyan, Peter Vrancx, Matthew E. Taylor, Daniel Kudenko, Ann Nowé: Multi-objectivization of reinforcement learning problems by reward shaping. IJCNN 2014: 2315-2322

[Wang & Sebag, 2013] Weijia Wang, Michèle Sebag: Hypervolume indicator and dominance reward based multi-objective Monte-Carlo Tree Search. Machine Learning 92(2-3): 403-429 (2013)

[van Moffaert et al, 2013a] Kristof Van Moffaert, Madalina M. Drugan, Ann Nowé: Hypervolume-Based Multi-Objective Reinforcement Learning. EMO 2013: 352-366

68

[van Moffaert et al, 2013b] Kristof Van Moffaert, Madalina M. Drugan, Ann Nowé: Scalarized multi-objective reinforcement learning: Novel design techniques. ADPRL 2013: 191-199

[Gabor et al, 1998] Z. Gabor, Z. Kalmar, and C. Szepesvari. Multi-criteria reinforcement learning. In ICML'98, pages 197{205. Morgan Kaufmann, 1998

[Drugan & Nowe, 2013] Madalina M. Drugan, Ann Nowé: Designing multi-objective multi-armed bandits algorithms: A study. IJCNN 2013: 1-8

[Fialho et al, 2009] Álvaro Fialho, Luís Da Costa, Marc Schoenauer, Michèle Sebag: Dynamic Multi-Armed Bandits and Extreme Value-Based Rewards for Adaptive Operator Selection in Evolutionary Algorithms. LION 2009: 176-190

[Puglierin et al, 2013] Francesco Puglierin, Madalina M. Drugan, Marco Wiering: Bandit-Inspired Memetic Algorithms for solving Quadratic Assignment Problems. IEEE Congress on Evolutionary Computation 2013: 2078-2085

[Audibert et al, 2010] Jean-Yves Audibert, Sébastien Bubeck, Rémi Munos: Best Arm Identification in Multi-Armed Bandits. COLT 2010: 41-53

[Drugan & Nowe, 2014] Madalina M. Drugan, Ann Nowé: Scalarization based Pareto optimal set of arms identification algorithms. IJCNN 2014: 2690-2697

[Drugan, 2015a] Madalina M. Drugan:Linear Scalarization for Pareto Front Identification in Stochastic Environments. EMO (2) 2015: 156-171

[Drugan, 2015b] Madalina M. Drugan: Multi-objective optimization perspectives on reinforcement learning algorithms using reward vectors, ESANN 2015

[Yahyaa et al, 2014] Saba Q. Yahyaa, Madalina M. Drugan, Bernard Manderick: Annealing Pareto multi-objective multi-armed bandit algorithm. ADPRL 2014: 1-8

[Thierens, 2005] Dirk Thierens: An adaptive pursuit strategy for allocating operator probabilities. GECCO 2005: 1539-1546

[Fialho et al, 2010] Álvaro Fialho, Luís Da Costa, Marc Schoenauer, Michèle Sebag: Analyzing bandit-based adaptive operator selection mechanisms. Ann. Math. Artif. Intell. 60(1-2): 25-64 (2010)

[Drugan & Thierens, 2011] Madalina M. Drugan, Dirk Thierens: Generalized adaptive pursuit algorithm for genetic pareto local search algorithms. GECCO 2011: 1963-1970

[Loshchilov et al, 2011] Ilya Loshchilov, Marc Schoenauer, Michèle Sebag: Not All Parents Are Equal for MO-CMA-ES. EMO 2011: 31-45

[Drugan & Talbi, 2014] Madalina M Drugan, Talbi El-Ghazali: Adaptive Multi-operator MetaHeuristics for quadratic assignment problems. EVOLVE 2014, Springer

[Karafotias et al, 2014] Giorgos Karafotias, Ágoston E. Eiben, Mark Hoogendoorn: Generic parameter control with reinforcement learning. GECCO 2014: 1319-1326

[Drugan et al, 2014] Madalina M. Drugan, Pedro Isasi, Bernard Manderick: Schemata Bandits for Binary Encoded Combinatorial Optimisation Problems. SEAL 2014: 299-310

[Kocsis & Szepesvári, 2006] Levente Kocsis, Csaba Szepesvári: Bandit Based Monte-Carlo Planning. ECML 2006: 282-293

[Preux et al, 2014] Philippe Preux, Rémi Munos, Michal Valko: Bandits attack function optimization. IEEE Congress on Evolutionary Computation 2014: 2245-2252

[Drugan, 2015c] Madalina M. Drugan: Efficient real-parameter single objective optimizer using hierarchical CMA-ES solvers, EVOLVE 2015, Springer.

[Ozcan et al, 2010] Ender Özcan, Mustafa Misir, Gabriela Ochoa, Edmund K. Burke: A Reinforcement Learning - Great-Deluge Hyper-Heuristic for Examination Timetabling. Int. J. of Applied Metaheuristic Computing 1(1): 39-59 (2010)

[Inja et al, 2014] Maarten Inja, Chiel Kooijman, Maarten de Waard, Diederik M. Roijers, Shimon Whiteson: Queued Pareto Local Search for Multi-Objective Optimization. PPSN 2014: 589-599