Selection of Auxiliary Objectives with Reinforcement Learning:
Overview of Theoretical Results

Arina Buzdalova  Maxim Buzdalov

ITMO UNIVERSITY

Saint-Petersburg, Russia

Dagstuhl Seminar 15211
Theory of Evolutionary Algorithms

May 21, 2015
Why do we need auxiliary objectives?
[A.B., M.B., Parfenov, 2013]

- Automated generation of tests for programming challenges
- Goal: maximize running time $T$ of an algorithm that solves a contest problem
- $T$ – the **target** objective
  - noisy, quantified and platform-dependent
- Auxiliary objectives — counters in the source code: $\{I, L, Q\}$
- May help to optimize $T$
- Source code example:

  Read the input data
  $I \leftarrow 0$, $L \leftarrow 0$, $last \leftarrow 0$
  while (solution not found) do
    Randomly shuffle ships and havens
    last $\leftarrow 0$
    Call the recursive ship arranging procedure
    For each call to this procedure, $last \leftarrow last + 1$
    if (solution is found) then
      Write the answer
    else
      $I \leftarrow I + 1$, $L \leftarrow L + last$, $last \leftarrow 0$
    end if
  end while
  $Q \leftarrow 10^9 \cdot I + last$
Related work

- Multi-objectivization — transformation of a single objective problem to a multi-objective problem [Knowles, Watson, Corne, 2001]
- Ways of applying auxiliary objectives
  - Optimize all of them simultaneously (often done after decomposition of the target objective) [Knowles et al., 2001]
  - Dynamically select random objective to be optimized [Jensen, 2004]
  - For Job-Shop problem: select objectives according to a special rule [Lochtefeld, Ciarallo, 2011]
- Theoretical analysis of complexity of EAs with multi-objectivization:
  - Scharnow, Tinnefeld, Wegener, 2005
  - Neumann, Wegener, 2006
  - Handl, Lovell, Knowles, 2008
  - Brockhoff, Friedrich, Hebbinghaus, Klein, Neumann, Zitzler, 2009

- EA+RL was proposed for the case when different objectives are efficient at different stages of optimization.

Action: agent selects objective from target and auxiliary objectives

Reward: difference between target objective values in two consecutive generations

State: in theoretical analysis, usually target value

Single-objective evolutionary algorithm is used
Problem: OneMax with ZeroMax obstructive objective

Algorithms: RLS (flip-one-bit mutation), Q-learning with no exploration

Proof technique: reducing a Markov chain to fitness levels

Result: $T_{EA+RL} < 2T_{EA} = \Theta(n \log n)$

Summary: EA+RL is able to ignore bad objective
Ignoring Inefficient Objective: solution with exploration
[M.B., A.B., CEC 2015]

- Problem: OneMax with ZeroMax obstructive objective
- Algorithms: RLS, Q-learning with $\varepsilon$-greedy exploration strategy
- Proof technique: analysis of a transition tree
- Results:
  - EA+RL is efficient when $\varepsilon = 0$ (the previous case)
  - When $\varepsilon > 0$, EA+RL performs in exponential time:

$$\Omega \left( \left( 1 + \frac{1}{1 - \varepsilon/2} \right)^N \right)$$

- Summary: it may be hard to ignore bad objective when exploration is turned on
Ignoring Inefficient Objective: \((1+1)\) solution
[Antipov, M.B., Doerr, 2015]

- Problem: OneMax with ZeroMax obstructive objective
- Algorithms: \((1+1)\) EA, Q-learning with no exploration
  - in \((1+1)\) EA each bit is flipped with probability \(1/n\)
- Proof technique: drift analysis
  - Potential function relies on both fitness (addend 1) and what has been learned (addend 2):
    \[
    \Phi(i, L) = \sum_{t=i}^{N-1} \frac{N}{N-t} + C \cdot N \sum_{t=0}^{N-1} \frac{1 - L(t)}{N-t}
    \]
- Result: running time is at most \(3.12eN\log N\)
- Summary: EA+RL is able to ignore bad objective also in the case of standard mutation
Taking Advantage of Efficient Objective
[M.B., A.B., 2014]

- Problem: \( \text{XdivK} = \left\lfloor \frac{\text{OneMax}}{k} \right\rfloor \) with OneMax aux. objective
- Algorithms: RLS, Q-learning with no exploration
- Proof technique: reducing a Markov chain to fitness levels
- Results: \( \frac{T_{EA}}{T_{EA+RL}} \geq 2^{k-2}(1 - o(1)) \)
- Summary: EA+RL is able to decrease running time asymptotically
No Asymptotic Advantages from Efficient Objective
[M.B., A.B., CEC 2015]

- Problem: LeadingOnes with OneMax auxiliary objective
- Algorithms: RLS, Q-learning with no exploration
- Proof technique: derived from [Böttcher, Doerr, Neumann, 2010]
- Results:
  - $\frac{N^2}{3} + o(N^2)$ for random objective selection (or when RL state is equal to LeadingOnes fitness)
  - $\frac{N^2}{4} + o(N^2)$ for a single RL state
  - compare: $\frac{N^2}{2}$ for RLS on LeadingOnes
- Summary: Auxiliary function that do not provide immediate growth of target function is not useful for simple EA+RL
Switch-and-Restart Algorithm: Alternative Method  
[M.B., 2014]

- Consider EAs with different objectives as different algorithms
- Run meta-iterations until an optimum is found, the i-th meta-iteration (i ≥ 0):
  - Run each algorithm G for $A_i$ iterations and then forget it
  - $\frac{A_{i+1}}{A_i} = 2$ is optimal
- Running time is at most $4K \min_G T_G$
- SaRA vs EA+RL
  - SaRA is more efficient than EA+RL when a single auxiliary objective may be used to reach the optimum of the target objective
  - SaRA is not able to use different objectives at different stages of optimization
Conclusion

▶ Summary: runtime analysis of simple EA+RL on some simple problems
  ▶ Optimization of OneMax is not affected by ZeroMax if there is no exploration
  ▶ Optimization of XdivK is sped up using OneMax
  ▶ It is hard to optimize LeadingOnes faster using OneMax

▶ Practical implications of EA+RL analysis
  ▶ Try no exploration — it may work good (supported by experiments)
  ▶ Construct auxiliary objectives that give immediate reward or
    use more sophisticated reinforcement learning (i.e. Delayed Q-learning)

▶ Future work:
  ▶ Theoretically analyzing the case when different objectives are useful at different stages of optimization
  ▶ Construction of an algorithm that combines advantages of SaRA and EA+RL
Thank you! Questions?

▶ Contacts
  ◀ abuzdalova@gmail.com, http://rain.ifmo.ru/~afanasyeva/
  ◀ mbuzdalov@gmail.com, http://rain.ifmo.ru/~buzdalov/

▶ Acknowledgments
  ◀ This work was partially financially supported by the Government of Russian Federation, Grant 074-U01