OneMax Helps Optimizing XdivK: Theoretical Runtime Analysis for RLS and EA+RL

Maxim Buzdalov
ITMO University
49 Kronverkskiy prosp.
Saint-Petersburg, Russia
mbuzdalov@gmail.com

Arina Buzdalova
ITMO University
49 Kronverkskiy prosp.
Saint-Petersburg, Russia
abuzdalova@gmail.com

ABSTRACT

There exist optimization problems with the target objective, which is to be optimized, and several extra objectives, which can be helpful in the optimization process. The previously proposed EA+RL method is designed to adaptively select objectives during the run of an optimization algorithm in order to reduce the number of evaluations needed to reach an optimum of the target objective.

The case when the extra objective is a fine-grained version of the target one is probably the simplest case when using an extra objective actually helps. We define a coarse-grained version of OneMax called XdivK as follows: $\textsc{XdivK}(x) = \lfloor \textsc{OneMax}(x)/k \rfloor$ for a parameter k which is a divisor of n — the length of a bit vector. We also define XdivK+OneMax, which is a problem where the target objective is XdivK and a single extra objective is OneMax.

In this paper, the randomized local search (RLS) is used in the EA+RL method as an optimization algorithm. We construct exact expressions for the expected running time of RLS solving the XdivK problem and of the EA+RL method solving the XdivK+OneMax problem. It is shown that the EA+RL method makes optimization faster, and the speedup is exponential in k.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; G.1.6 [**Numerical Analysis**]: Optimization

General Terms

Algorithms, Experimentation, Performance, Theory

Keywords

helper-objectives, expected running time, multiobjectivization

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598442.

1. INTRODUCTION

Single-objective optimization can often benefit from multiple objectives [7]. Additional objectives may be introduced to escape from the plateaus [2]. Primary objective may be decomposed into several objectives [4]. Additional objectives may also arise from the problem structure [6].

Different approaches may be applied to a problem with the "original" (*target*) objective and some extra objectives. The *multi-objectivization* approach is to optimize *all* extra objectives simultaneously [4]. The *helper-objective* approach is to optimize simultaneously the target objective and one or more extra objectives, switching between them [5].

These approaches assume that extra objectives are crafted to help optimizing the target objective. In general case, extra objectives may support or obstruct optimization process. The EA+RL method was developed to cope with that [3]. Its idea is to use a single-objective evolutionary optimization algorithm and switch between the objectives. To select objectives, reinforcement learning is used.

We show that for the XdivK benchmark problem with the parameter k EA+RL with the OneMax extra objective speeds up optimization by the factor of at least 2^{k-2}. Proofs and details are available in supplementary materials [1].

2. DEFINITIONS

The well-known OneMax problem is defined as follows. Given n, the size of the problem. The search space consists of all bit vectors of length n. The fitness function of a bit vector is the number of bits set to one. One needs to maximize the fitness function.

The XdivK problem is defined in this paper in the similar way. The fitness function is the number of bits set to one divided evenly by the parameter k. The parameter k is a divisor of n. The problem XdivK+OneMax has the target objective XdivK and a single extra objective OneMax.

We consider a simplistic optimization algorithm called "randomized local search" (RLS). It stores the current candidate solution x. An iteration of this algorithm first constructs y, a copy of x with one random bit flipped, and if y is not worse than x according to the fitness function, then x is replaced by y.

3. EXPECTED RUNNING TIME

In this section, we derive the exact expressions for the expected running time for both XdivK problem solved by RLS and XdivK+OneMax problem solved by RLS under the control of reinforcement learning. We assume that the

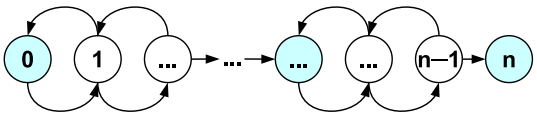

Figure 1: An overview of the Markov chain. The states correspond to OneMax fitness value, the clusters of states correspond to XdivK fitness value.

RL state is determined solely by the value of XdivK fitness function, and the reward is equal to the difference of XdivK fitness values in consecutive optimization states. There are n bits in the bit vector, and the division factor is k, such that $n \bmod k = 0$. The following fact is proven [1]:

Lemma 1. *The EA+RL algorithm never returns to any state where some reward has been obtained.*

This means that in the case of XdivK+OneMax both the XdivK and the OneMax fitness functions are always chosen with the probability of 1/2.

To compute the expected running time of the algorithms, we construct Markov chains for them. The state of a Markov chain is determined in this paper by the OneMax fitness value. The states are clustered into consecutive groups of n/k states, except for the terminal state n, which is on its own (Fig. 1). The clusters correspond to the states with the same value of XdivK fitness. The probabilities for different algorithms are given in supplementary materials [1].

Let $Z_X(x)$ be the expected number of steps for RLS solving XdivK problem to reach the state $x+1$ from the state x. It is shown in supplementary materials [1] that:

$$Z_X(x) = \frac{n}{n-x} \qquad \text{if } x \bmod k = 0,$$
$$Z_X(x) = \frac{n}{n-x} + Z_X(x-1)\frac{x}{n-x} \qquad \text{if } x \bmod k \neq 0.$$

The equivalent value $Z_R(x)$ for RLS under EA+RL solving XdivK+OneMax can be computed as follows:

$$Z_R(x) = \frac{n}{n-x} \qquad \text{if } x \bmod k = 0,$$
$$Z_R(x) = \frac{n}{n-x} + Z_R(x-1)\frac{x}{2(n-x)} \qquad \text{if } x \bmod k \neq 0.$$

Theorem 1. *The following holds for all x:*

$$Z_X(x) = \sum_{i=0}^{x \bmod k} \frac{\binom{n}{x-i}}{\binom{n-1}{x}}; \quad Z_R(x) = \sum_{i=0}^{x \bmod k} 2^{-i}\frac{\binom{n}{x-i}}{\binom{n-1}{x}}.$$

Running times for RLS solving XdivK problem (T_X) and for RLS using EA+RL solving XdivK+OneMax problem (T_R) starting from all-zero bit vector are:

$$T_X(n,k) = \sum_{x=0}^{n-1} Z_X(x); \quad T_R(n,k) = \sum_{x=0}^{n-1} Z_R(x).$$

Denote $\sum_{j=i}^{k-1}\sum_{m=0}^{\frac{n}{k}-1}\frac{\binom{n}{mk+j-i}}{\binom{n-1}{mk+j}}$ as $V(n,k,i)$. It is shown that:

$$T_X(n,k) = \sum_{i=0}^{k-1} V(n,k,i); \quad T_R(n,k) = \sum_{i=0}^{k-1} 2^{-i} V(n,k,i).$$

Theorem 2. *For constant k, $V(n,k,i) = \Omega(n^{i+1}) = O(n^{i+2})$.*

Theorem 3. *The complexity of both $T_X(n,k)$ and $T_R(n,k)$ for constant k is $\Omega(n^k)$ and $O(n^{k+1})$.*

Theorem 4. *For sufficiently large n and fixed k,*

$$T_X(n,k) \geq 2^{k-2}(1-o(1)) \cdot T_R(n,k).$$

4. CONCLUSION

We presented the exact expressions for $T_X(n,k)$ — the expected running time of RLS solving the XdivK problem, — and for $T_R(n,k)$ — the expected running time of the EA+RL method using RLS solving the XdivK+OneMax problem with the RL state equal to the XdivK fitness value and with the reward equal to the difference of XdivK fitness values between consecutive states.

Using these expressions, we gave theoretical evidence that $T_X(n,k) = \Omega(n^k) = O(n^{k+1})$ for constant k, $T_R(n,k) = \Omega(n^k) = O(n^{k+1})$ for constant k, and $T_X(n,k)/T_R(n,k)$ approaches at least 2^{k-2} for constant k and large n.

The values of $T_X(n,k)$ and $T_R(n,k)$ for several n and k are presented in supplementary materials [1]. The value of T_X/T_R can be seen to grow slowly with n when k is constant. The ratio is very much similar to 2^{k-1}.

For asymptotic determination, we tabulate the values of $T_X(n,k)$ and $T_X(n/2,k)$ for several n and k. The results are presented in supplementary materials [1]. One can see that the value of $T_X(n,k)/T_X(n/2,k)$ approaches 2^k as n grows, which suggests that the reality is $T_X(n,k) = \Theta(n^k)$.

5. REFERENCES

[1] Supplementary materials (proofs and tables). URL: https://github.com/mbuzdalov/papers/blob/master/2014-gecco-xdivk/xdivk-extra.pdf.

[2] D. Brockhoff, T. Friedrich, N. Hebbinghaus, C. Klein, F. Neumann, and E. Zitzler. On the Effects of Adding Objectives to Plateau Functions. *Transactions on Evolutionary Computation*, 13(3):591–603, 2009.

[3] A. Buzdalova and M. Buzdalov. Increasing Efficiency of Evolutionary Algorithms by Choosing between Auxiliary Fitness Functions with Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning and Applications*, volume 1, pages 150–155, 2012.

[4] J. Handl, S. C. Lovell, and J. D. Knowles. Multiobjectivization by Decomposition of Scalar Cost Functions. In *Parallel Problem Solving from Nature – PPSN X*, volume 5199 of *Lecture Notes in Computer Science*, pages 31–40. Springer Berlin Heidelberg, 2008.

[5] M. T. Jensen. Helper-Objectives: Using Multi-Objective Evolutionary Algorithms for Single-Objective Optimisation: Evolutionary Computation Combinatorial Optimization. *Journal of Mathematical Modelling and Algorithms*, 3(4):323–347, 2004.

[6] D. F. Lochtefeld and F. W. Ciarallo. Helper-Objective Optimization Strategies for the Job-Shop Scheduling Problem. *Applied Soft Computing*, 11(6):4161–4174, 2011.

[7] F. Neumann and I. Wegener. Can Single-Objective Optimization Profit from Multiobjective Optimization? In *Multiobjective Problem Solving from Nature*, Natural Computing Series, pages 115–130. Springer Berlin Heidelberg, 2008.