On Binary Unbiased Operators Returning Multiple Offspring

Nina Bulanova
ITMO University
49 Kronverkskiy ave.
Saint-Petersburg, Russia 197101
ninokfox@gmail.com

Maxim Buzdalov
ITMO University
49 Kronverkskiy ave.
Saint-Petersburg, Russia 197101
mbuzdalov@gmail.com

ABSTRACT

The notion of unbiased black-box complexity plays an important role in theory of randomized search heuristics. A black-box algorithm is usually defined as an algorithm which uses unbiased variation operations. In all known papers, the analysed variation operators take k arguments and produce one offspring. On the other hand, many practitioners use crossovers which produce two offspring, and in many living organisms a diploid cell produces two distinct haploid genotypes.

We investigate how the binary-to-binary, or $(2 \to 2)$, unbiased variation operators look like, and how they can be used to improve randomized search heuristics. We show that the $(2 \to 2)$ unbiased black-box complexity of Needle coincides with its unrestricted black-box complexity. We also show that it can be used to put strong worst-case guarantees for solving OneMax.

CCS CONCEPTS

•Theory of computation → Theory of randomized search heuristics;

KEYWORDS

Black-box complexity, crossovers, Needle, OneMax.

ACM Reference format:
Nina Bulanova and Maxim Buzdalov. 2017. On Binary Unbiased Operators Returning Multiple Offspring. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 4 pages.
DOI: http://dx.doi.org/10.1145/3067695.3082505

1 INTRODUCTION

Evolutionary algorithms, as well as most other randomized search heuristics (RSHs), are generally seen as problem-agnostic solvers which gain information about the problem by the sole means of querying the so-called *fitness function.* In theoretical research, the number of calls to the fitness function is a primary performance measure of a randomized search heuristic. A huge body of research is dedicated to runtime analysis of various algorithms on various problem classes, which produces statements such as "the running time of a $(1 + 1)$ evolutionary algorithm on the OneMax problem is $\Theta(n \log n)$, where n is the problem size" [1]. A counterpart to this is measuring the complexity of a problem with respect to a family of algorithms: the *black-box complexity* of a problem is, loosely speaking, the expected running time of a best possible algorithm on this problem [5].

As RSHs are problem-agnostic, they shall not prefer one instance of a problem to another one. This means that a "good" general-purpose RSH, speaking in terms of genetic algorithms, shall treat equally two different genes or two alleles of the same gene. In a sense, if such an algorithm is invariant under certain transformations of the problem search space, we can prove its properties for the entire problem, or even a class of problems, by analysing a single problem instance. However, this does not come for free, as specialized problem solvers, even those which receive information about the problem just by using the fitness function, may make use of the problem properties which are known for them, and this gives them an advantage over RSHs. This, in particular, results in ridiculuously small black-box complexities of certain problems [5]. Together, this gave rise to the notion of the *unbiased black-box complexity*, where the analysed algorithms are only allowed to operate in such ways which remain symmetric under certain transformations of the search space [7].

Lehre and Witt [7] introduced the notion of a *unbiased variation operator* for pseudo-Boolean problems (which have the search space consisting of all bit strings of a fixed length n). An unbiased variation operator is invariant with respect to changing of meanings of the particular bit values (that is, if all its arguments undergo a bitwise exclusive-or operation with a certain bit string x, then its result also undergoes the same transformation), and also with respect to changing bit positions (that is, if a certain permutation π is applied to all arguments, then the result also changes accordingly). Most often, the mutation-only, or *unary*, unbiased black-box complexity is studied, due to its simplicity. However, it was shown that using more than one argument can provably yield better algorithms [4], which is, in a sense, a motivation for using crossovers, as whether crossovers are needed was a long-standing issue in evolutionary computation [6]. In fact, even for the simple problems, such as OneMax, crossover was shown to speed up optimization by a constant factor [8], and, very recently, an algorithm was proposed which yields an asymptotic speed-up [2, 3].

However, in the entire body of theoretical analysis of the RSHs, only the operators which take $K \geq 1$ search points as arguments and return *one* new search point were considered. In the case of $K = 2$, this contradicts with the common usage of crossovers in the practice of evolutionary computation, when both offsping, resulting in swapping several genes between the parents, are evaluated and participate in the selection. In a sense, it contradicts with what we

GECCO '17 Companion, Berlin, Germany

978-1-4503-4939-0/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3067695.3082505

see in real life, where meiosis results in two haploid cells, which both subsequently participate in reproduction.

One possible reason for such a difference may be seen as follows: when considering the OneMax problem, where fitness is measured as the number of bits equal to 1, and keeping in mind the conventional crossovers, which only exchange genes, the second offspring can be seen as an unnecessary waste of the fitness query budget, as if the OneMax values of the parents were A and B, and the value of the first offspring is C, then the value of the second offspring is $A + B - C$, that is, no query is needed to find its fitness.

In this paper, we suggest the scientific community to revisit this area. In particular, we performed some initial explorations of what we call $(2 \to 2)$*-ary unbiased operators* and find that $(2 \to 2)$ unbiased operators make it possible to solve Needle in optimal time, and they also can derandomize a binary unbiased algorithm to solve OneMax in at most $2n$ queries.

2 UNARY AND BINARY OPERATORS

In the following, we consider search spaces consisting of bit strings of length n, that is, $\{0, 1\}^n$. By $[1..n]$ we denote a set of integers from 1 to n inclusively.

A *k-ary variation operator* $\mathcal{X}$ produces a search point y from the given k search points $x_1, \ldots, x_k$ with probability $P_{\mathcal{X}}(y \mid x_1, \ldots, x_k)$. The operator $\mathcal{X}$ is *unbiased* if the following relations hold for all search points $x_1, \ldots, x_k, y, z$ and all permutations π over $[1..n]$:

$$P_{\mathcal{X}}(y \mid x_1, \ldots, x_k) = P_{\mathcal{X}}(y \oplus z \mid x_1 \oplus z, \ldots, x_k \oplus z), \quad (1)$$
$$P_{\mathcal{X}}(y \mid x_1, \ldots, x_k) = P_{\mathcal{X}}(\pi(y) \mid \pi(x_1), \ldots, \pi(x_k)), \quad (2)$$

where $a \oplus b$ is the bitwise exclusive-or operation applied to two bit strings a and b of the same length, and $\pi(a)$ is an application of permutation π to a bit string a. In simple words, Eq. 1 declares that $\mathcal{X}$ is invariant under flipping the i-th bits, for any i, in both arguments and the result, and Eq. 2 declares that $\mathcal{X}$ is invariant under permuting bits in the same way in arguments and the result.

The single-bit-flip mutation operator and the standard bit mutation operator, with any bit flipping probability, are certainly unary unbiased operators. An example of a binary unbiased operator is a homogeneous crossover operator, with any bit swapping probability, which returns either of the offspring.

All unary unbiased operators can be characterized as follows.

Theorem 2.1. *The following statements are equivalent for an unary operator $y = \mathcal{X}(x)$ defined on bit strings of length n:*

(1a) *$\mathcal{X}$ is unbiased.*
(1b) *$\mathcal{X}$ can be represented using the following algorithm:*
  - *first, a mutation size m is chosen in an arbitrary way from $\{0\} \cup [1..n]$;*
  - *second, a subset $S \subseteq [1..n]$ with size m is chosen uniformly at random, and y is obtained from x by flipping bits with indices from S.*

Proof. (1a)→(1b): As proven in [7, Proposition 1], every unbiased unary operator is Hamming-invariant, that is, for every $d \in \{0\} \cup [1..n]$ it samples all points at the Hamming distance d from the argument with equal probability p_d. Thus, $\mathcal{X}$ can be expressed as (1b) by choosing m with probability $p_d \cdot \binom{n}{d}$.

(1b)→(1a): First, (1b) either leaves a bit intact or flips it, thus it is invariant under flipping. Second, all subsets of a certain size are chosen equiprobably, thus (1b) is invariant under permutations. □

Description of all binary unbiased operators is slightly more complicated. We are unaware of any such description available in the literature, although it may belong to the common sense. Nevertheless we give and prove it.

Theorem 2.2. *The following statements are equivalent for a binary operator $y = \mathcal{X}(x_1, x_2)$ defined on bit strings of length n:*

(2a) *$\mathcal{X}$ is unbiased.*
(2b) *$\mathcal{X}$ can be represented using the following algorithm:*
  - *first, the subset $Q \in [1..n]$ is determined such that $i \in Q$ if and only if the i-th bits in x_1 and x_2 coincide;*
  - *second, based on $|Q|$ and n only, two integers s and d are chosen, such that $0 \leq s \leq |Q|$, $0 \leq d \leq n - |Q|$;*
  - *third, a subset $S \subseteq Q$ with size s, and a subset $D \subseteq [1..n] \setminus Q$ with size d, are chosen uniformly at random, and y is obtained from x_1 by flipping bits with indices from $S \cup D$.*

Proof. In simple words, we need to prove that a binary operator is unbiased if and only if it decides how many coinciding bits, and how many differing bits, need to be flipped in the first argument, and then chooses these bits at random and flips them.

(2b)→(2a): For any given strings x_1 and x_2, the subset Q does not change if they are replaced with $x_1 \oplus z$ and $x_2 \oplus z$ correspondingly, for any bit string z. Thus the probability of any subset of bit indices to be flipped simultaneously does not change. This means that exactly the same bits of the new result y', which are different from y, are different in x_1 and $x_1 \oplus z$, thus $y' = y \oplus (x_1 \oplus (x_1 \oplus z)) = y \oplus z$ and $\mathcal{X}$ is invariant under flipping.

If a permutation π is applied to both x_1 and x_2, then the new same-bits subset Q' will be $\pi(Q)$ as well. As $|\pi(Q)| = |Q|$, the choice of s and d is not affected. The probability of any subset $S \subseteq Q$ to be flipped in x_1 is thus equal to the probability of $\pi(S) \subseteq \pi(Q)$ to be flipped in x_1, the similar statement is true for any subset $D \subseteq [1..n] \setminus Q$, which means that the probability of sampling $y' = \pi(y)$ after applying the permutation is the same as the probability of sampling y in the original setup.

(2a)→(2b): Assume x_1 and x_2 have at least t equal bits, and T is the set of indices of such bits. We need to show that, for any two subsets $T_1 \subseteq T$ and $T_2 \subseteq T$, both of size t, $y = \mathcal{X}(x_1, x_2)$ has the same probability of:

- being y_1: different from x_1 in bits from T_1 and coincide with it in bits from $T \setminus T_1$;
- being y_2: different from x_1 in bits from T_2 and coincide with it in bits from $T \setminus T_2$.

As $|T_1 = T_2|$, one can construct a permutation π such that $T_2 \setminus T_1 = \pi(T_1 \setminus T_2)$ and $T_1 \setminus T_2 = \pi(T_2 \setminus T_1)$ where set equality is assumed, while all other indices are not changed. Let also $z = x_1 \oplus \pi(x_1)$. The composition of these two transformations $z \circ \pi$ transforms x_1 into x_1, as well as x_2 into x_2, and y_1 into y_2. This means that, as $\mathcal{X}$ is unbiased, the probabilities of achieving y_1 and y_2 are equal. As a result, processing equal bits is Hamming-based, similarly to the unary case, thus can be expressed in terms of (2b). Proving the case for differing bits is symmetric, thus completing the proof. □

3 EXTENDED UNBIASED OPERATORS

The notion of k-ary unbiased variation operators is easily extensible to operators which result in producing more than one offspring.

3.1 $(k \to m)$-ary Operators and Complexities

We call a $(k \to m)$-*ary variation operator* $\mathcal{X}$, defined on bit strings of length n, a function, probably a non-deterministic one, which takes k bit strings as arguments and produces m bit strings as a result. The order of the resulting bit strings matters, as well as the order of the arguments.

An $(k \to m)$-ary variation operator is unbiased if the following relations hold for all search points $x_1, \ldots, x_k, y_1, \ldots, y_m, z$ and all permutations π over $[1..n]$:

$$P_{\mathcal{X}}(y_1, \ldots, y_m \mid x_1, \ldots, x_k) = P_{\mathcal{X}}(y_1 \oplus z, \ldots, y_m \oplus z \mid x_1 \oplus z, \ldots, x_k \oplus z), \quad (3)$$

$$P_{\mathcal{X}}(y_1, \ldots, y_m \mid x_1, \ldots, x_k) = P_{\mathcal{X}}(\pi(y_1), \ldots, \pi(y_m) \mid \pi(x_1), \ldots, \pi(x_k)). \quad (4)$$

We define the $(k \to m)$-*ary unbiased black-box complexity* of a problem as the infimum of the expected runtime for all unbiased black-box algorithms which use only $(\alpha \to \beta)$-ary variation operators such that $0 \le \alpha \le k$ and $1 \le \beta \le m$.

The previously existing unbiased operators and complexity measures fit well into this extended notation. Namely a k-ary (unbiased) variation operator is $(k \to 1)$-ary (unbiased) variation operator, and the k-ary unbiased black-box complexity is the same as the $(k \to 1)$-ary unbiased black-box complexity.

3.2 $(2 \to 2)$-ary Unbiased Operators

In this paper we concentrate primarily on $(2 \to 2)$-ary unbiased variation operators. A simple example of a $(2 \to 2)$-ary unbiased operator is the uniform crossover, which produces offspring by flipping the bits of the parents, residing at the same indices, with a probability p, which is the same for all bits, and returns both offspring as a result. Note that, when $p \neq 0.5$, the order of the offspring matters.

A notable fact is that $(2 \to 2)$-ary operators need not to be symmetric, that is, they are not limited to uniform crossovers, even if they are allowed to make distinction between equal and differing bits of the parents. For example, such an operator is able to control, while remaining unbiased, whether to flip or not to flip a bit, compared to the first argument x_1, in the first and the second offspring separately, thus having *four*, not two, groups of bits with different behavior within each sets of bits (equal or differing in parents) which binary unbiased operators are allowed to distinguish. This enables six degrees of freedom in $(2 \to 2)$-ary unbiased operators, compared to only two degrees of freedom in the conventional binary unbiased operators, which we may denote as $(2 \to 1)$-ary operators.

Similarly to Theorem 2.2, we state a theorem for the $(2 \to 2)$-ary operators. It is proven in precisely the same manner as Theorem 2.2, but we omit the proof for the sake of brevity.

Theorem 3.1. *The following statements are equivalent for a* $(2 \to 2)$-*ary operator* $(y_1, y_2) = \mathcal{X}(x_1, x_2)$ *defined on bit strings of length* n:

(3a) $\mathcal{X}$ *is unbiased.*

(3b) $\mathcal{X}$ *can be represented using the following algorithm:*

- *first, the subset* $Q \in [1..n]$ *is determined such that* $i \in Q$ *if and only if the* i*-th bits in* x_1 *and* x_2 *coincide;*
- *second, based on* $|Q|$ *and* n *only, six non-negative integers* s_{10}, s_{01}, s_{11}, d_{10}, d_{01}, d_{11} *are chosen, such that* $0 \le s_{10} + s_{01} + s_{11} \le |Q|$, $0 \le d_{10} + d_{01} + d_{11} \le n - |Q|$;
- *third, six non-intersecting subsets* $S_{10}, S_{01}, S_{11}, D_{10}, D_{01}, D_{11}$ *are chosen uniformly, such that* $|S_{10}| = s_{10}$, $|S_{01}| = s_{01}$, $|S_{11}| = s_{11}$, $|D_{10}| = d_{10}$, $|D_{01}| = d_{01}$, $|D_{11}| = d_{11}$, $S_{10} \cup S_{01} \cup S_{11} \subseteq Q$, $D_{10} \cup D_{01} \cup D_{11} \subseteq [1..n] \setminus Q$;
- *fourth,* y_1 *is created by flipping in* x_1 *bits at indices from the set* $S_{10} \cup S_{11} \cup D_{10} \cup D_{11}$, *and* y_2 *is created by flipping in* x_1 *bits at indices from the set* $S_{01} \cup S_{11} \cup D_{01} \cup D_{11}$.

Informally speaking, every $(2 \to 2)$-ary unbiased operator is free to split the bit indices where the parents x_1 and x_2 have equal bits, into four non-intersecting groups corresponding to whether the first parent's bit will be flipped in each offspring, and do the same with the remaining bit indices (where x_1 and x_2 differ) as well.

If one extends the idea of Theorem 2.2 to *ternary* unbiased operators (in the new terminology, the $(3 \to 1)$-ary unbiased operators), one can see that ternary operators, like the $(2 \to 2)$-ary, also have eight blocks of bits with different decisions. It is also not difficult to see that every $(2 \to 2)$-ary unbiased operator can be simulated with a system of one $(2 \to 1)$-ary and one $(3 \to 1)$-ary unbiased operator with no additional cost. This results in the following corollary.

Corollary 3.2. *For every problem, its* $(2 \to 2)$-*ary unbiased black-box complexity is at least as high as its* $(3 \to 1)$-*ary unbiased black-box complexity.*

4 $(2 \to 2)$-ARY UNBIASED BBC OF NEEDLE

The needle-in-the-haystack problem, or simply Needle, is one of the hardest black-box optimization problems. It provides no guidance towards the optimum through answers to queries, as all these answers are equal to zero in all points except for the optimum itself [5]. More formally, Needle_n is the set of problem instances $\text{Needle}_{n,z}$ defined on bit strings of length n as follows:

$$\text{Needle}_{n,z} : \{0,1\}^n \to \mathbf{Z}; x \mapsto 1 \text{ if } x = z, 0, \text{ otherwise},$$

where $z \in \{0,1\}^n$ is the optimum.

The query complexity of Needle is $(2^n + 1)/2$, which corresponds to shuffling all search points, without repetitions, in a random order and querying them in this order [5]. On the other hand, just querying random search points brings the expected runtime of 2^n, which is only a factor of 2 apart from the best possible algorithm.

Typical evolutionary algorithms are even worse: for instance, for a constant $0 < \eta < n/2$ there exists a constant $c > 0$ such that with probability $1 - 2^{-\Omega(n)}$ the $(1 + 1)$ evolutionary algorithm creates search points with the distance to the optimum at least ηn in 2^{cn} steps [1, Theorem 2.13].

We prove that $(2 \to 2)$-ary unbiased operators are powerful enough to reach the best possible performance on Needle.

Theorem 4.1. *The* $(2 \to 2)$-*ary unbiased black-box complexity of Needle is* $(2^n + 1)/2$.

Algorithm 1 The $(2 \to 2)$-ary unbiased algorithm for NEEDLE

```
procedure TRAVERSE(x1, x2, n, ℓ)
    if ℓ + 1 < n then
        (y1, y2) ← SWAPONEWHEREDIFFERENT(x1, x2)
        QUERY(y1)
        QUERY(y2)
        TRAVERSE(x1, y2, n, ℓ + 1)
        TRAVERSE(y1, x2, n, ℓ + 1)
    end if
end procedure
procedure MAIN(n)
    x1 ← UNIFORMRANDOM({0, 1}^n)
    x2 ← INVERSE(x1)
    QUERY(x1)
    QUERY(x2)
    TRAVERSE(x1, x2, n, 0)
end procedure
```

Algorithm 2 The $(2 \to 2)$-ary unbiased algorithm for ONEMAX

```
procedure MAIN(n, f ∈ ONEMAX)
    x1 ← UNIFORMRANDOM({0, 1}^n)
    x2 ← INVERSE(x1)
    QUERY(x1)
    QUERY(x2)
    for i ∈ [1..n] do
        (y1, y2) ← SWAPONEWHEREDIFFERENT(x1, x2)
        QUERY(y1)
        QUERY(y2)
        if f(y1) > f(x1) then
            x1 ← y1
        else
            x2 ← y2
        end if
    end for
end procedure
```

PROOF. We will use the algorithm outlined as Algorithm 1. It uses one unary unbiased variation operator called INVERSE, which performs inversion of the argument, and one $(2 \to 2)$-ary unbiased variation operator, SWAPONEWHEREDIFFERENT, which chooses one random bit index among the indices of bits which are different in parents and swaps the bits on this index to achieve two offspring.

To prove the theorem, it is enough to prove that this algorithm queries every search point exactly once, as the random first query will also randomize the position of the optimum in the list of queries.

The procedure TRAVERSE(x_1, x_2, n, ℓ) is called on two search points x_1 and x_2 which have exactly ℓ equal bits. We prove that this procedure will query all search points, which have the same ℓ bits equal to the ones in x_1 and x_2, except x_1 and x_2 themselves. As the MAIN procedure calls TRAVERSE with $\ell = 0$ on the complementary x_1 and x_2, which are already queried by that time, this will prove the entire theorem.

We use induction by ℓ. The base case, $\ell + 1 = n$, is obvious: the search points x_1 and x_2 differ in exactly one bit, so no more queries are needed, and TRAVERSE indeed does not do anything. Consider any ℓ such that $0 \le \ell < n-1$. The newly produced points, y_1 and y_2, share the same bit index, in which y_1 differs from x_1 and y_2 differs from x_2. This means that the remaining points to be queried agree either with y_1, in which case they will be queried in the recursive call TRAVERSE(y_1, x_2, n, $\ell + 1$) by induction, or with y_2, which is covered by TRAVERSE(x_1, y_2, n, $\ell + 1$). □

Note that this theorem does not work when two independent $(2 \to 1)$-ary operators are used to create y_1 and y_2, since these operators cannot guarantee that y_1 and y_2 will differ from their corresponding parents in the same bit, and thus they cannot split the search space into the necessary halves.

5 DERANDOMIZATION OF ONEMAX

A simple binary unbiased algorithm for solving ONEMAX was proposed in [4, Algorithm 4] with the expected running time of $2n+o(n)$ and the following runtime guarantee: the running time exceeds $2n(1 + \varepsilon)$ with probability at most $\exp(-\varepsilon^2 n/2(1 + \varepsilon))$.

We show that using a $(2 \to 2)$-ary unbiased variation operator yields a stronger, "more deterministic", runtime guarantee of $2n$ with probability one. The corresponding algorithm is outlined as Algorithm 2. As in [4, Algorithm 4], the invariant of the algorithm is as follows: the bits coinciding in x_1 and x_2 are guessed correctly. Every pair of queries in the for-loop puts the right value for the bit residing at the index chosen for swapping by the SWAPONEWHEREDIFFERENT operator in both x_1 and x_2, thus after $2n$ queries x_1 and x_2 will agree in $n - 1$ bits which are set correctly. In this case, exactly one of them is the optimum.

6 CONCLUSION

We introduced an extension of k-ary unbiased variation operators, the $(k \to m)$-ary unbiased variation operators which take k arguments and produce m offspring. Our initial explorations showed that even $(2 \to 2)$-ary operators can be quite powerful. By this research, we hope to show one more dimension to the hierarchy of black-box complexities, which may in turn produce one more way towards better algorithms.

REFERENCES

[1] Anne Auger and Benjamin Doerr. 2011. *Theory of Randomized Search Heuristics: Foundations and Recent Developments.* World Scientific Publishing Co., Inc., River Edge, NJ, USA.
[2] Benjamin Doerr and Carola Doerr. 2015. Optimal Parameter Choices Through Self-Adjustment: Applying the 1/5-th Rule in Discrete Settings. In *Proceedings of Genetic and Evolutionary Computation Conference.* 1335–1342.
[3] Benjamin Doerr, Carola Doerr, and Franziska Ebel. 2015. From black-box complexity to designing new genetic algorithms. *Theoretical Computer Science* 567 (2015), 87–104.
[4] Benjamin Doerr, Daniel Johannsen, Timo Kötzing, Per Kristian Lehre, Markus Wagner, and Carola Winzen. 2011. Faster black-box algorithms through higher arity operators. In *Proceedings of Foundations of Genetic Algorithms.* 163–172.
[5] Stefan Droste, Thomas Jansen, and Ingo Wegener. 2006. Upper and Lower Bounds for Randomized Search Heuristics in Black-Box Optimization. *Theory of Computing Systems* 39, 4 (2006), 525–544.
[6] Thomas Jansen and Ingo Wegener. 2002. The Analysis of Evolutionary Algorithms–A Proof that Crossover Really Can Help. *Algorithmica* 34 (2002), 47–66.
[7] Per Kristian Lehre and Carsten Witt. 2012. Black-box Search by Unbiased Variation. *Algorithmica* 64 (2012), 623–642.
[8] Dirk Sudholt. 2012. Crossover speeds up building-block assembly. In *Proceedings of Genetic and Evolutionary Computation Conference.* 689–696.