Дерандомизация задачи OneMax с помощью бинарного оператора

Буланова Н. С., аспирант кафедры компьютерных технологий университета ИТМО, ninasbulanova@gmail.com
Буздалов М. В., к.т.н, доцент кафедры компьютерных технологий университета ИТМО, mbuzdalov@gmail.com

Аннотация

В теории случайных поисковых эвристик важнейшую роль играет понятие объективной сложности черного ящика (black-box complexity), который в свою очередь определяется как алгоритм, использующий несмещенные операторы (unbiased operations). Известно множество работ, в которых операторы принимают k аргументов и производят лишь одного потомка. В данной работе изучаются бинарные или $(2 \to 2)$ несмещенные операторы, и то как они помогают дерандомизировать поисковые эвристики на примере задачи OneMax.

Введение

Эволюционные алгоритмы, как и большинство случайных поисковых эвристик (Randomized Search Heuristics), обычно получают информацию о задаче только с помощью запроса *функции приспособленности*. В теоретических исследованиях количество обращений для подсчета функции приспособленности является первичным показателем эффективности случайной поисковой эвристики.

Говоря в терминах генетических алгоритмов, хорошими случайные поисковые эвристики можно назвать, когда они не предпочитают один экземпляр задачи другому. Это означает, что «хороший» RSH должен одинаково относиться к двум различным генам или двум аллелям одного и того же гена. В некотором смысле такой алгоритм инвариантен относительно некоторых преобразований пространства поиска задачи.

Часто, из-за простоты изучения, в black-box complexity используются унарные операторы. Тем не менее, было показано, что использование более одного аргумента может улучшить эффективность алгоритма [3], что, в некотором смысле, является мотивацией для использования кроссоверов, поскольку необходимость в кроссоверах была давней проблемой в эволюционных вычислениях [4]. Фактически, даже для простых задач, таких как OneMax, было показано, что кроссовер

ускоряет оптимизацию с помощью постоянного коэффициента [6], и совсем недавно был предложен алгоритм, который дает асимптотическое ускорение [2, 1].

Однако во всем теоретическом анализе случайных поисковых эвристик учитывались только операторы, берущие несколько точек поиска в качестве аргументов и возвращающие *единственную* новую точку поиска. В каком-то смысле это противоречит тому, что мы видим в реальной жизни, где мейоз приводит к двум гаплоидным клеткам, которые обе впоследствии участвуют в размножении. Одной из возможных причин такого противоречия может быть то, что например при рассмотрении задачи OneMax, где фитнес измеряется как количество бит равных единице, второй потомок кроссовера может рассматриваться как ненужная трата запросов функции приспособленности, так как его фитнес можно вычислить непосредственно из значений приспособленности родителей A и B и первого потомка C, как $A + B - C$.

В этой статье приводится опровержение бесполезности второго потомка кроссовера на примере задачи OneMax с помощью дерандомизированного алгоритма с $(2 \to 2)$-арным несмещенным оператором.

$(k \to 1)$-арные и $(k \to m)$-арные операторы

Понятие *несмещенного вариационного оператора* введенно в [5] для псевдобулевых задач, в которых пространство поиска состоит из всех битовых строк фиксированной длины n. Такие операторы являются инвариантными относительно изменения значений конкретных битов, то есть после поразрядного применения исключающего "или"(XOR) к аргументам, результат подвергается такому же преобразованию. Так же применение перестановок к аргументам приводит к соответствующему изменению результата.

Таким образом, если мы имеем дело с операторами преобразующими k аргументов в одну результирующую особь, то вышеизложенные условия можно формально представить следующим образом:

$$P(y \mid x_1, \ldots, x_k) = P(y \oplus z \mid x_1 \oplus z, \ldots, x_k \oplus z), \tag{1}$$

$$P(y \mid x_1, \ldots, x_k) = P(\pi(y) \mid \pi(x_1), \ldots, \pi(x_k)), \tag{2}$$

где $a \oplus b$ это операция побитового XOR, и $\pi(a)$ это применение перестановки π к битовой строке a.

Для того чтобы описать подобное условие для $(k \to m)$-арные операторы, нужно всего лишь заменить в выражении единственный вывод

y на вывод множества $y_1, \ldots, y_m$, применив к нему соответсвенно перестановку или операцию XOR.

$$P(y_1, \ldots, y_m \mid x_1, \ldots, x_k) = P(y_1 \oplus z, \ldots, y_m \oplus z \mid x_1 \oplus z, \ldots, x_k \oplus z), \quad (3)$$

$$P(y_1, \ldots, y_m \mid x_1, \ldots, x_k) = P(\pi(y_1), \ldots, \pi(y_m) \mid \pi(x_1), \ldots, \pi(x_k)). \quad (4)$$

$(2 \to 2)$-арные несмещенные операторы

В данном исследовании в первую очередь изучались $(2 \to 2)$-арные несмещенные операторы. Простой пример такого оператора - однородный кроссовер, который производит потомство, обменивая биты родителей с вероятностью p, одинаковой для всех битов, и в результате возвращает обоих потомков. В случае $p \neq 0.5$ имеет значение порядок потомков.

Примечательным фактом является то, что $(2 \to 2)$-арные операторы не обязательно должны быть симметричными, то есть они не ограничиваются равномерным кроссовером. Например, такой оператор может контролировать, оставаясь несмещенным, обменивать или не обменивать биты, по сравнению с первым аргументом x_1, в первом и втором потомках отдельно. Таким образом получается не две, а четыре группы с различным поведением в каждом наборе битов. Это дает шесть степеней свободы в $(2 \to 2)$-арных несмещенных операторах, по сравнению только с двумя степенями свободы в обычных бинарных несмещенных операторах, которые можно обозначить как $(2 \to 1)$-арные операторы.

Дерандомизация задачи OneMax

В источнике [3] предлагается беспристрастный алгоритм решения OneMax с ожидаемым временем оптимизации $2n + o(n)$и следующей гарантией выполнения: время выполнения превышает $2n(1+\varepsilon)$ с вероятностью не более $\exp(-\varepsilon^2 n/2(1+\varepsilon))$.

Использование же несмещенного оператора $(2 \to 2)$ позволяет создать более сильный, дерандомизированный алгоритм, который гарантированно решает поставленную задачу за $2n$ шагов. Соответствующий алгоритм изложен в Листинг 1.

Листинг 1 $(2 \to 2)$-арный несмещенный алгоритм для OneMax

```
procedure Main(n, f ∈ OneMax)
    x_1 ← UniformRandom({0,1}^n)
    x_2 ← Inverse(x_1)
    Query(x_1)
    Query(x_2)
    for i ∈ [1..n] do
        (y_1, y_2) ← SwapOneWhereDifferent(x_1, x_2)
        Query(y_1)
        Query(y_2)
        if f(y_1) > f(x_1) then
            x_1 ← y_1
        else
            x_2 ← y_2
        end if
    end for
end procedure
```

Визуализация процесса дерандомизации представлена на Рис. 1. Суть данного алгоритма в том, что изначально случайно сгенерированная особь x_1 и обратная ей Inverse(x_1) различаются во всех битах (желтая область на рисунке). Какие из бит этих особей установлены верно, то есть способствуют увеличению функции приспособленности, нам не известно. В последствии, обменивая местами различающиеся биты в особях (зеленая область на рисунке) с помощью SwapOneWhereDifferent (SOWD на рисунке), и отбирая особи у которых приспособленность лучше, чем у исходных, за $2n$ запросов можно добиться согласованности в $n-1$ битах. В этом случае одна из особей точно является оптимальной.

Доказательство корректности предложенного алгоритма можно произвести как и в [3] с помощью инварианта алгоритма, согласно ко-

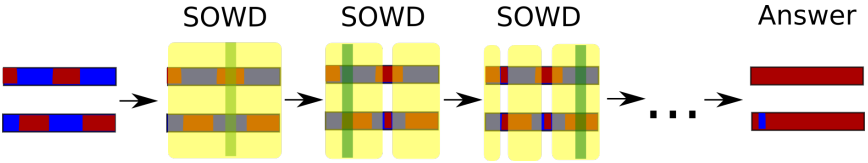

Рис. 1: Дерандомизация задачи OneMax

торому биты, совпадающие в x_1 и x_2, считаются правильными. Каждая пара запросов в цикле for помещает правильное значение для бита, находящегося по определенному индексу, выбранному для операции обмена SwapOneWhereDifferent в x_1 и x_2. Таким образом, после $2n$ запросов x_1 и x_2 согласуются в $n-1$ битах и одна из получившихся особей является решением задачи.

Заключение

В работе был представлен $(k \to m)$-арный несмещенный оператор, производящий из k аргументов m потомков. Проведенное исследование показало, что даже простой $(2 \to 2)$-арный оператор может быть достаточно эффективным, что создаёт позитивное направление дальнейшего исследования подобных операторов.

Литература

[1] B. Doerr and C. Doerr. Optimal parameter choices through self-adjustment: Applying the 1/5-th rule in discrete settings. // Proceedings of Genetic and Evolutionary Computation Conference, pages 1335–1342, 2015.

[2] B. Doerr, C. Doerr, and F. Ebel. From black-box complexity to designing new genetic algorithms. // Theoretical Computer Science, 567:87–104, 2015.

[3] B. Doerr, D. Johannsen, T. Kötzing, P. K. Lehre, M. Wagner, and C. Winzen. Faster black-box algorithms through higher arity operators. // Proceedings of Foundations of Genetic Algorithms, pages 163–172, 2011.

[4] T. Jansen and I. Wegener. The analysis of evolutionary algorithms–a proof that crossover really can help. // Algorithmica, 34:47–66, 2002.

[5] P. K. Lehre and C. Witt. Black-box search by unbiased variation. // Algorithmica, 64:623–642, 2012.

[6] D. Sudholt. Crossover speeds up building-block assembly. // Proceedings of Genetic and Evolutionary Computation Conference, pages 689–696, 2012.