электронный научно-технический журнал
ИНЖЕНЕРНЫЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51036. ISSN 2307-0595

Мета-оптимизация методом настройки параметров. Обзор

 # 11, ноябрь 2015

Агасиев Т. А. оглы[1,*], профессор, д.ф.-м.н. Карпенко А. П.[1]

УДК: 519.6

[1]Россия, МГТУ им. Н.Э. Баумана
[*]taaalex@mail.ru

Введение

В связи со стремительным развитием технологий производства и ростом сложности выпускаемой продукции, усложняются и методы анализа проектных решений (конструкций, структурных схем, расписаний и т.п.). Успех производителя во многом зависит от качества изделия и сроков его разработки. Таким образом, задача поиска оптимальных проектных решений является более чем актуальной. Для ее решения все чаще прибегают к использованию автоматизированных систем, поддерживающих интеллектуальные методы синтеза проектных решений.

Процесс оптимизации требует огромных вычислительных ресурсов из-за сложности современных инженерных изделий. Высокие темпы развития технологий подразумевают разработку и использование наиболее современных и сложных алгоритмов оптимизации. Все менее очевидным становится вопрос выбора того или иного алгоритма. Эффективность решения конкретной задачи также зависит от правильной настройки используемого алгоритма и требует понимания процесса оптимизации со стороны пользователя.

Автоматизированная настройка алгоритмов позволяет инженерам использовать весь потенциал современных методов оптимизации без помощи специалистов в области математики. Задача оптимальной настройки алгоритма называется задачей мета-оптимизации.

1. Постановка задачи мета-оптимизации

Большинство современных алгоритмов оптимизации имеют свободные параметры, позволяющие настраивать эти алгоритмы. Совокупность свободных параметров алгоритма называют его *стратегией*. Правильный подбор значений параметров алгоритма может значительно улучшить его эффективность при решении той или иной задачи. Возникает задача поиска оптимальной стратегии алгоритма, процесс решения которой называют *мета-оптимизацией*. При этом настраиваемый алгоритм оптимизации называют *базовым*, а настраивающий алгоритм - *мета-алгоритмом*.

В работе [1] приведена постановка задачи мета-оптимизации. Обозначим некоторый класс задач многомерной безусловной однокритериальной оптимизации $\Phi = (\phi_i, i = 1, 2, \ldots)$, где $\phi_i = \phi$ одна из задач рассматриваемого класса. Пусть $a(B)$ - базовый алгоритм, решающий задачи класса Ф, где $B = (b_1, b_2, \ldots, b_{|B|})$ - вектор параметров этого алгоритма. Положим, что определено множество допустимых стратегий алгоритма

$$D_B = \{b_i \mid b_i^- \le b_i \le b_i^+, i \in [1{:}\,|B|]\},$$

где b_i^-, b_i^+- интервалы допустимых значений компоненты b_i вектора B. Множество D_B определяет совокупность алгоритмов

$$A = \{a(B) | B \in D_B\}.$$

Таким образом, можно рассматривать две постановки задачи мета-оптимизации. В первом случае нужно найти такую стратегию B^* для функции $\phi \in \Phi$, которая обеспечит минимальное значение целевой *мета-функции* $\mu(\phi, B)$:

$$\min_{B \in D_B} \mu(\phi, B) = \mu(\phi, B^*).$$

Вторая постановка задачи мета-оптимизации имеет следующий вид. Для класса задач Ф найти стратегию B^*, которая обеспечивает минимальное значение той же целевой мета-функции:

$$\min_{B \in D_B} \mu(\Phi, B) = \mu(\Phi, B^*).$$

Стратегию алгоритма можно разделить на две составляющие. *Параметрическая* стратегия представляет собой вектор параметров алгоритма, имеющих интервалы допустимых значений. Набор параметров алгоритма, определяющих его структуру (принцип действия), называют *структурной* стратегией алгоритма. Поиск оптимальной параметрической стратегии называют параметрической мета-оптимизацией, а поиск оптимальной структурной стратегии - структурной мета-оптимизацией.

Описание различных показателей эффективности стратегий и способы оценки устойчивости базового алгоритма можно найти в главе 1.2 работы [2].

2. Классификация методов мета-оптимизации

Существует много различных способов классификации методов мета-оптимизации. Возьмем за основу классификацию, предложенную Эйбеном (A.E. Eiben) в работе [3]. В соответствии с ней на верхнем уровне выделяют две группы методов - методы *настройки* параметров и методы *управления* параметрами. При настройке параметров стратегия алгоритма фиксирована во время работы алгоритма. Управление параметрами предполагает варьирование параметров стратегии на каждой, или некоторых итераций базового алгоритма.

2.1. Настройка параметров

Методы настройки параметров предполагают нахождение оптимальной стратегии алгоритма для ряда задач некоторого класса. После этого найденные эффективные стратегии базового алгоритма используют для решения других задач рассматриваемого класса. Выделяют методы *однократной* и *перманентной* настройки.

Методы однократной настройки параметров представляют собой классические методы мета-оптимизации. Идея метода состоит в том, что настраиваемый алгоритм многократно тестируют с различными стратегиями на большом числе оптимизационных задач, которые принадлежат рассматриваемому классу. Данный подход не эффективен из-за ошибочности предположения, что параметры базового алгоритма устойчивы к вариациям задач оптимизации. Экспоненциальный рост вычислительных затрат при увеличении числа параметров в стратегии не позволяет провести исчерпывающее исследование.

В методах перманентной настройки, мета-оптимизатор в процессе работы накапливает информацию об эффективности тех или иных стратегий базового алгоритма при решении различных оптимизационных задач. На основании собранной информации варьируются параметры базового алгоритма с целью найти их оптимальные значения. При этом полученные результаты могут быть использованы другими исследователями и практиками без необходимости изменения их собственных реализаций базового алгоритма.

Схема классификации Эйбена приведена на рисунке 1.

Рис. 1. Классификация Эйбена

2.2. Управление параметрами

Методы управления параметрами решают задачу мета-оптимизации в процессе работы базового алгоритма. Оптимальная стратегия подбирается под конкретную задачу. Такой подход часто требует модификации базового алгоритма, что увеличивает его вычислительную сложность. В соответствии с классификацией [3], методы управления параметрами можно разделить на методы *адаптации* и *самоадаптации*.

При адаптивном управлении параметрами алгоритм мета-оптимизации надстраивается над базовым алгоритмом. В процессе решения задачи оптимизации, на основании информации о ходе решения, меняется стратегия базового алгоритма. В роли метода мета-оптимизации часто может выступать какое-либо эвристическое правило.

Методы самоадаптации подразумевают включение параметров стратегии в пространство поиска базового алгоритма, что увеличивает размерность исходной задачи. Также модифицируется целевая функция исходной задачи и базовый алгоритм начинает выполнять функции мета-оптимизатора.

Основным преимуществом методов управления параметрами перед методами настройки является простота синтеза динамических стратегий. В случае настройки параметров базовый алгоритм не требует доработки и рассматривается как "черный ящик". Более подробное сравнение методов настройки и управления параметрами можно найти в работе [4].

2.3. Альтернативная классификация методов настройки

Можно выделить две основные цели настройки параметров [2]. Первой является нахождение оптимальной стратегии алгоритма, при которой его производительность будет максимальна для рассматриваемого класса задач. За это отвечает *интенсификационная* составляющая поиска. Вторая цель - получение информации о поведении алгоритма для более глубокого его понимания, изучения устойчивости стратегий алгоритма к вариациям базовой задачи. Для этого требуется обширное исследование пространства поиска мета-задачи оптимизации. В таком случае говорят о *диверсификационной* составляющей поиска.

Таким образом, методы мета-оптимизации различаются соотношением интенсификации и диверсификации. На основании вышесказанного, в работе [2] выделили четыре основных подхода к настройке параметров:

- *мета-стохастические поисковые методы* (*meta-randomized search methods, meta-RSM*) "заточены" под быстрое отыскание эффективных стратегий,
- *выборочные методы* (*sampling methods*) позволяют провести обширное исследование пространства поиска мета-оптимизатора,
- *скрининговые методы (screening methods),*
- методы, которые используют математические модели эффективности алгоритма *(model-based methods).*

Последние два подхода комбинируют свойства первых двух. В рамках терминологии работы [2], методы настройки бывают *итеративными* и *не итеративными*. При итеративной настройке параметров набор тестируемых стратегий не фиксируется, новые стратегии генерируются в процессе настройки - на каждой или некоторых итерациях мета-оптимизации.

Методы однократной настройки параметров могут быть итеративными и не итеративными, в то время как методы перманентной настройки строго итеративные. Наоборот, итеративные методы можно использовать как для однократной, так и для перманентной настройки параметров алгоритма. Не итеративные методы подойдут скорее для однократной настройки.

На рисунке 2 представлена схема для сравнения подходов к настройке параметров. Каждый блок представляет набор стратегий алгоритма. Эффективные стратегии обозначаются зеленым цветом, не эффективные - красным. Стрелками обозначаются потоки векторов параметров или информации об их эффективности. К примеру, отличительной особенностью методов, использующих математические модели, является обратная связь от тестирования стратегий к генерации новых. Таким образом, эти методы относятся к ите-

ративным. В скрининговых методах обратная связь находится внутри этапа тестирования. Это говорит о том, что только лучшие стратегии (зеленые) тестируются повторно в следующих итерациях.

Следует отметить, что большинство современных методов настройки параметров тем или иным образом комбинируют стратегии диверсификации и интенсификации.

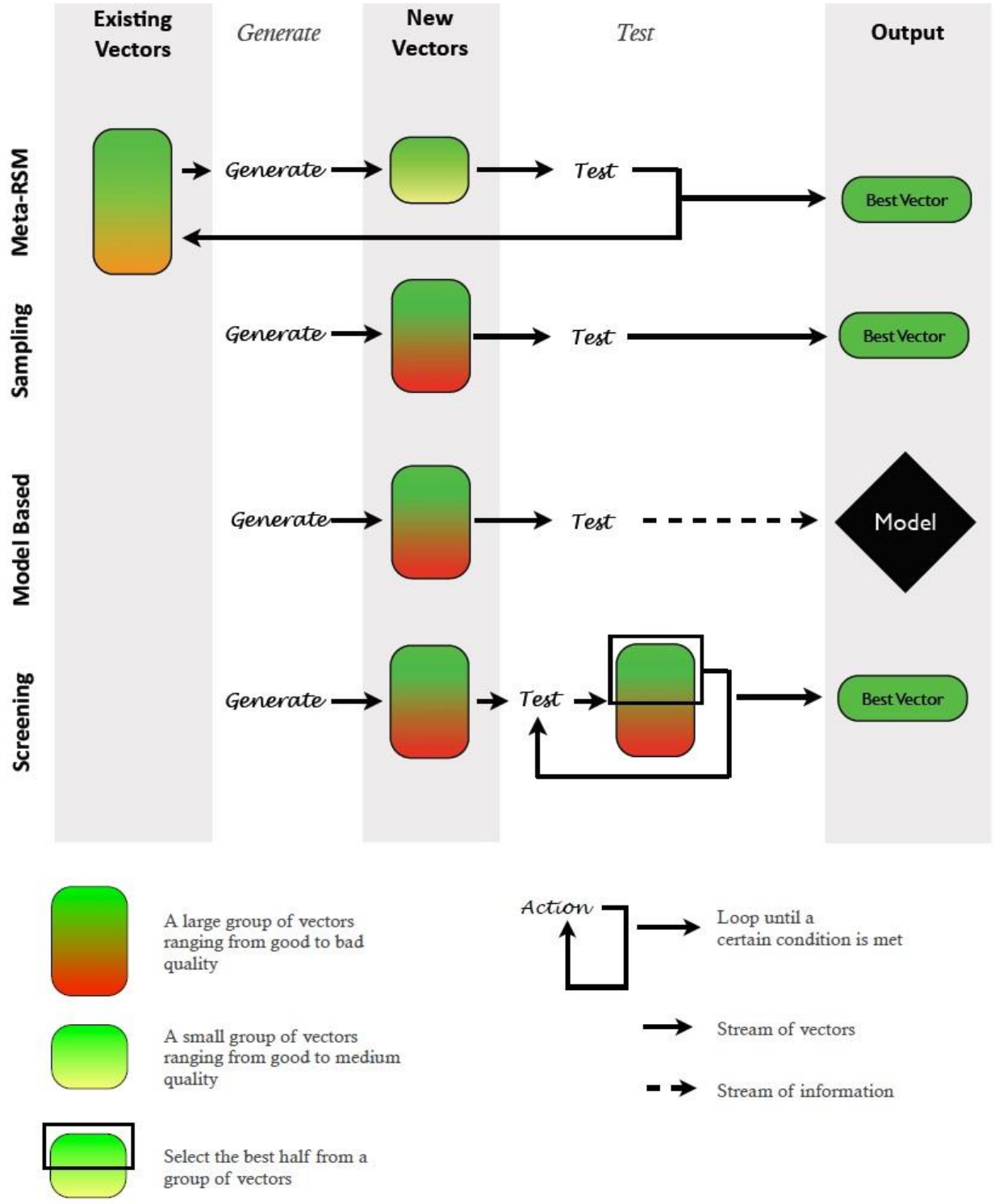


Рис. 2. Четыре подхода к настройке параметров [2]

3. Методы настройки параметров

3.1. Выборочные методы

Выборочные методы сокращают вычислительные затраты на мета-оптимизацию, уменьшая общее число тестируемых стратегий. Два наиболее ярких метода, реализующих данный подход: латинский квадрат (*Latin-Square*) [5] и метод Тагути (*Taguchi Orthogonal Arrays*) [6].

Результаты работы выборочных методов должны быть дополнительно проанализированы для определения наиболее устойчивых стратегий и стратегий, при которых алгоритм максимально эффективен для решения задач рассматриваемого класса. Поэтому данные методы скорее используют на стадии инициализации (например, для *Model-Based* методов), нежели в качестве самостоятельных мета-оптимизаторов. Отсутствие интенсификационной составляющей поиска приводит к нахождению стратегий низкого качества.

Имеются работы, посвященные итеративным выборочным методам таким, как *CALIBRA* [7] и *Empirical Modelling of Genetic Algorithms* [5]. В них область поиска новых стратегий базового алгоритма определяется на каждой итерации мета-уровня, чего не происходит в не итеративных выборочных методах. Таким образом, итеративные выборочные методы могут использоваться в качестве самостоятельных мета-оптимизаторов.

Метод, описанный в работе [5], содержит в себе два этапа. Сначала пространство поиска мета-алгоритма представляется в виде греко-латинского квадрата. Затем происходит перебор всех возможных комбинаций значений компонентов стратегии базового алгоритма в уже более узких диапазонах.

Метод *CALIBRA* начинает с полного перебора первого и третьего квантилей диапазона значений каждого компонента стратегии. На основании полученной информации, векторы параметров для следующей итерации генерируются в суженном диапазоне. Синтез новых стратегий осуществляется при помощи *Taguchi Orthogonal Array* [6]. Процедура повторяется до превышения максимального числа итераций.

Выборочные методы, в большинстве своем, являются простыми и не требуют значительных вычислительных затрат. Мысль о том, что мета-оптимизация должны быть простой и понятной, изложена в работе [8]. В ней предложено использовать два простых алгоритма для настройки параметров: на основе паттернов (*Pattern Search*) и локальной унимодальной выборки (*Local Unimodal Sampling*). Оба алгоритма являются итеративными. Исследования не показали заметного падения эффективности найденных оптимальных стратегий по сравнению с эволюционными алгоритмами. При этом вычислительная сложность данных методов минимальна.

3.2. Настройка с использованием мета-моделей

Мета-модели или *суррогатные* модели широко используются при решении оптимизационных задач высокой вычислительной сложности [9, 10]. Если говорить о настройке параметров, то в этом случае строится модель для аппроксимации функции эффективности стратегий базового алгоритма. Наличие такой модели позволяет уменьшить число до-

рогостоящих тестов стратегий алгоритма. Часто для определения эффективности не тестированных стратегий используют метод регрессии [11-13].

Моделью функции эффективности алгоритма называют формулу, отображающую зависимость некоторого показателя эффективности алгоритма от значений компонентов его стратегии. Пример модели алгоритма с двумя настраиваемыми параметрами и показателем эффективности u имеет вид

$$u(p_1, p_2) = a_0 + a_1 p_1 + a_2 p_1^2 + a_3 p_2 + a_4 p_2^2 + a_5 p_1 p_2,$$

где коэффициенты a_i определяют значимость соответствующего параметра алгоритма. Чем выше значение коэффициента a_i, тем больше будет разброс значений показателя эффективности алгоритма при варьировании соответствующего компонента стратегии. Если алгоритм настраивается под широкий спектр оптимизационных задач, то большие колебания параметров a_i могут говорить о низкой устойчивости алгоритма к вариациям решаемых задач.

Очевидно, что при использовании не итеративных мета-оптимизаторов данного класса точность модели функции эффективности алгоритма будет относительно низкой, как и качество найденных оптимальных стратегий алгоритма. Итеративные методы, использующие мета-модели, позволяют провести более точное исследование пространства параметров алгоритма.

Процедура Коя [14] является одной из базовых расширений методов настройки с использованием математических моделей. Идея процедуры состоит в использовании локального поиска для уточнения значений параметров. На первом этапе пространство параметров дискретизируется, и в узлах полученной сетки вычисляются показатели эффективности стратегий. Затем на основании полученных данных строится линейная регрессионная модель функции эффективности алгоритма, по которой определяются области поиска оптимальных стратегий. На втором этапе выбранные стратегии тестируются до стагнации показателей эффективности. Так как на этом этапе не предусмотрено обновление модели, качество найденных стратегий напрямую зависит от корректности модели, построенной на первом этапе.

В отличие от двухступенчатой процедуры Коя, метод *SPO* (*Sequential Parameter Optimization*) постоянно обновляет модель функции эффективности алгоритма [15, 16]. Каждая итерация начинается с генерации набора тестируемых векторов параметров и прогнозирования их эффективности с использованием модели. Тестируются только самые перспективные стратегии, после чего модель обновляется. При этом число тестов каждого вектора параметров может варьироваться в зависимости от желаемой точности оценки эффективности алгоритма. После превышения допустимого числа итераций метод завершает свою работу. Результатом является модель функции эффективности алгоритма, уточненная в наиболее перспективных областях.

Качество найденных оптимальных стратегий во многом зависит от типа используемой модели. Авторы рассмотренных работ предпочли использовать модель Кригинга. Исследования различных типов суррогатных моделей, способов их построения и использования в целях настройки параметров приведены в работе [17].

3.3. Скрининговые методы

Идеей скрининговых методов является выявление лучшей стратегии с помощью минимального числа запусков базового алгоритма. Достигается это путем тестирования только перспективных векторов параметров, заслуживающих дальнейшего исследования. Таким образом, выявление лучших стратегий скрининговыми методами требует меньших вычислительных затрат, чем при использовании выборочных методов. Это позволяет получить более эффективные оптимальные стратегии и больше информации об устойчивости алгоритма при одинаковых вычислительных затратах.

Скрининговые методы сформировались под влиянием методов выбора лучшей из имеющихся систем путем имитационного моделирования, целью которых является уменьшение необходимого для выбора числа стохастических имитаций [18]. При этом аналогом стратегии алгоритма можно назвать одну из моделируемых систем, запуск базового алгоритма - стохастическая имитация, а методы сравнения и выбора моделирующих систем - мета-оптимизаторы.

Четыре основных работы в данном направлении рассмотрены в публикации [19]: *Interactive Analysis* [20], *Ranking and Selection* [21], *Multiple Comparison Procedures* [22] и *Fully Sequential Indifference-zone Selection procedure* [23]. Методы отличаются только гарантиями выбора лучшей системы и необходимым числом итераций. Такие скрининговые методы основаны на допущении, что результаты моделирования распределяются по нормальному закону.

Такого рода предположения, хоть и являются слабо обоснованными, могут быть достигнуты при помощи ранжирования [19]. Преимуществом скрининговых методов можно назвать определенные гарантии локализации наилучшей стратегии в некоторой области пространства параметров (*Ranking and Selection, Fully Sequential Indifference-zone Selection procedure*). В методе *Multiple Comparison Procedures* оптимальная стратегия будет иметь некоторый уровень доверия. Более подробную информацию о данных методах можно получить в [18, 19].

Применительно к настройке параметров часто используется понятие *гонки*, которое описано в работе [24]. В сущности, данный подход не значительно отличается от методов выбора лучшей системы путем моделирования, за исключением возможности рассмотрения огромного числа претендентов. Преимуществом гонок является то, что они не требуют каких-либо допущений о распределении значений эффективности настраиваемого алгоритма (*Hoeffding Races* [24], *F-RACE* [25]).

Ярким представителем итеративных скрининговых методов можно назвать метод *I/F-RACE* [26] (расширенный *F-RACE*). Этот метод сочетает в себе техники скрининга и локального уточняющего поиска. В начале вся область допустимых значений параметров алгоритма покрывается крупной сеткой с относительно небольшим числом узлов. Затем при помощи гонок число тестируемых стратегий уменьшается до определенного уровня. При помощи выживших стратегий строится многомерное нормальное распределение, которое затем используется в качестве функции плотности вероятности. Таким образом, для

последующего разбиения выбираются наиболее перспективные области пространства параметров. Так как метод *I/F-RACE* делает свой выбор, опираясь на полученное распределение, его можно отнести к итеративным методам, использующим мета-модели.

3.4. Мета-стохастические поисковые методы

Первым представителем методов данного класса является мета-эволюционный алгоритм (*meta-EA*) [27], предложенный в 1978-м году. В данной работе для настройки параметров использовался эволюционный алгоритм, но исследования были очень ограниченными из-за высокой вычислительной сложности мета-алгоритма.

Более обширные исследования провели над мета-генетическим алгоритмом (*meta-GA*) [28], который показал эффективность случайного поиска при настройке параметров. Стратегии базового алгоритма соответствовали особям мета-алгоритма, а показатель эффективности стратегий определял фитнес-функцию. Такая интерпретация задачи мета-оптимизации позволяет использовать для настройки параметров любой эволюционный алгоритм. Настройка параметров отличается зашумленностью функции эффективности базового алгоритма (фитнес-функции) и высокой вычислительной сложностью итераций. В связи с этим, были разработаны специализированные методы случайного поиска, использующие скрининговые техники.

Примером таких методов является метод *FocusedILS* [29], который сочетает эволюционную стратегию с локальным уточняющим поиском. В данной работе скрининг используется для сравнения эффективности различных стратегий. Похожее решение предложено в работе [30], автор которой добавил гонки в алгоритм *meta-GA*, что дало возможность проведения структурной и параметрической настройки алгоритма. Обе работы демонстрируют целесообразность такого рода модификаций, а именно: большее число рассмотренных стратегий, ускорение мета-оптимизации и рост эффективности найденных стратегий. Недостатком этого метода можно назвать усложнение мета-алгоритма и рост вычислительных затрат на мета-оптимизацию.

Существует такое понятие как *многозадачная* настройка алгоритма, целью которой является отыскание эффективной стратегии, максимально устойчивой к вариациям оптимизационных задач базового уровня. Этот подход отличается тем, что "надежная" стратегия алгоритма скорее всего не будет оптимальной для решения ни одной из рассматриваемых задач. Следует отличать многозадачную настройку от поиска оптимального вектора параметров для класса задач. В первом случае задачи могут значительно отличаться друг от друга, что делает нецелесообразным использование средних значений эффективности алгоритма.

Метод *SMAC* (*Sequential Model-based Algorithm Configuration*) [31] представляет собой модифицированный метод *FocusedILS*, сочетающий скрининг с построением мета-моделей. Метод *SMAC* предлагает использование методов локального уточняющего поиска для генерации наиболее перспективных векторов параметров. Перспективность стратегии определяется на основе модели случайного леса (*Random Forest Model*) [32].

Кроме того, метод *SMAC* направлен на многозадачную настройку. Пользователем определяется функция эффективности алгоритма, именуемая *expected positive improvement*, которая учитывает ожидаемый разброс решаемых задач при определении эффективности алгоритма. Нечто похожее используется в методе *Local Unimodal Sampling* [33], в котором эффективность стратегии определяется как средняя эффективность при решении рассматриваемого набора задач. Так как речь идет о многозадачной настройке алгоритма, мета-модель также собирает информацию об особенностях решаемых задач. Таким образом, точность мета-модели напрямую зависит от возможности предоставления данного рода информации.

Определение особенностей решаемых задач является достаточно сложной проблемой, что ограничивало возможности использования методов многозадачной настройки. Относительно недавнее исследование показало эффективность так называемого *диверсификационного ландшафтного анализа* (*exploratory landscape analysis*) для выявления особенностей оптимизируемых задач в методе *BBOB* [34].

3.5. Многокритериальные методы

Использование общего показателя эффективности для разных типов задач имеет явный недостаток. Рассматриваемые задачи могут значительно отличаться в плане сложности целевых функций. Мета-оптимизатор отдаст предпочтение стратегиям, которые эффективны при решении оптимизационных задач большей вычислительной сложности, поскольку они дают больший вклад в показатель эффективности стратегий. Величину смещения оптимальных стратегий в пользу более сложных задач можно уменьшить, используя средние геометрические значения показателей эффективности.

Ситуацию может исправить рассмотрение многозадачной настройки алгоритма как задачи многокритериальной мета-оптимизации. Такая точка зрения вполне оправдана, так как каждый показатель эффективности алгоритма и тип решаемой задачи можно выделить в отдельные критерии оптимальности мета-алгоритма. При помощи многокритериальных методов оптимизации можно построить Парето-фронт стратегий алгоритма, что позволит изучить устойчивость различных показателей эффективности стратегии к изменениям решаемых задач [35].

Метод, представленный в работе [36], использовал в качестве критериев оптимальности различные показатели эффективности стратегий. В качестве мета-алгоритма использовался известный метод многокритериальной оптимизации *NSGA-II* [37]. Авторы работы [35] пошли дальше и разработали специальный метод многокритериальной настройки параметров *M-FETA*. Его основным преимуществом является наличие специальных операторов, понижающих число тестов каждой стратегии. Эффективность некоторых стратегий оценивается с помощью имеющихся значений эффективности соседних. В перспективных областях пространства параметров генерируется больше стратегий, что уменьшает расстояние между соседними векторами параметров и уточняет оценки эффективности. Таким образом уменьшается число дорогостоящих запусков базового алгоритма, а значит и вычислительные затраты на мета-оптимизацию.

3.6. Современные методы настройки

3.6.1. Метод *REVAC*

Метод *REVAC* [38] (*Relevance Estimation and Value Calibration of Parameters*) является итеративным мета-стохастическим поисковым методом настройки параметров. Имеется многокритериальная модификация метода. Для лучшего понимания данного метода, рассмотрим таблицу 1. Строки таблицы содержат стратегии алгоритма $\vec{x}^j$, которые состоят из k параметров алгоритма, u^j - усредненный показатель эффективности стратегии. Столбцы $D(x_i)$ таблицы отображают m допустимых значений i-го параметра.

Таблица 1. Векторы параметров базового алгоритма

	$D(x_1)$ ... $D(x_i)$ ... $D(x_k)$	$Utility$
$\vec{x}^1$	$\{x_1^1 \quad ... \quad x_i^1 \quad ... \quad x_k^1\}$	u^1
.	.	.
.	.	.
.	.	.
$\vec{x}^n$	$\{x_1^n \quad ... \quad x_i^n \quad ... \quad x_k^n\}$	u^n
.	.	.
.	.	.
.	.	.
$\vec{x}^m$	$\{x_1^m \quad ... \quad x_i^m \quad ... \quad x_k^m\}$	u^m

Метод *REVAC* работает с популяцией из m стратегий, что позволяет его отнести к эволюционным алгоритмам. Популяция постоянно обновляется путем скрещивания двух выбранных стратегий (родителей), после чего в популяцию добавляется их потомок. Родители выбираются из числа $n < m$ лучших стратегий, при этом каждый новый вектор параметров вытесняет самый "старый" вектор популяции. Итерации метода продолжаются до превышения допустимого числа испытанных стратегий.

Таким образом, после завершения настройки по столбцам таблицы можно определить оптимальные интервалы значений какого-либо из параметров. Чем больше особей находятся в некотором интервале, тем перспективнее эта область.

Метод *REVAC* имеет несколько модификаций, улучшающих его работу. Например, можно уменьшить число тестов стратегии до определения ее эффективности, а значит и общее число итераций, добавив гонки [24]. Идея данной модификации - не фиксировать число запусков базового алгоритма для определения эффективности стратегии. Изначально выполняется всего несколько тестов для каждой стратегий, постепенно отбрасываются откровенно плохие стратегии. Фаза гонок заканчивается после получения n лучших стратегий или после превышения максимального числа перезапусков тестов. Оставшиеся векторы участвуют в последующих итерациях метода *REVAC*.

Есть и другая модификация, направленная на повышение качества оценки эффективности стратегий. Метод *sharpening* (заточка) был предложен как самостоятельный метод в

работе [39], хоть и являлся частью метода *SPO* [15]. Идеей модификации является двукратное увеличение числа тестов каждого вектора параметров при определенных условиях. Изначально это число принимается очень маленьким, что позволяет быстро исследовать пространство поиска. При нахождении перспективной области, метод пытается уменьшить погрешность определения эффективности стратегии. Таким образом, чем лучше вектор, тем чаще он будет тестироваться.

Эти две модификации можно объединить следующим образом. Метод заточки будет, по мере необходимости, увеличивать максимальное число тестов для гонок. Изначально максимальное число тестов будет мало, что снизит эффективность гонок на ранних этапах. По мере роста числа тестов использование гонок позволит серьезно сэкономить ресурсы за счет отбрасывания не перспективных стратегий. Таким образом, основные усилия пойдут на тщательное исследование перспективных областей пространства параметров.

3.6.2. Метод *Bonesa*

Метод *Bonesa* [2] относится к итеративным методам настройки параметров использующим мета-модели. Схема алгоритма изображена на рисунке 3, на котором видно два рабочих цикла. Поисковый цикл на каждой итерации мета-оптимизации генерирует новые точки пространства поиска, лучшие из которых будут тестированы. Цикл обучения использует информацию об эффективности тестированных стратегий для построения модели функции эффективности базового алгоритма, при помощи которой будут оцениваться не протестированные стратегии. Предварительные оценки стратегий затем используются в цикле поиска.

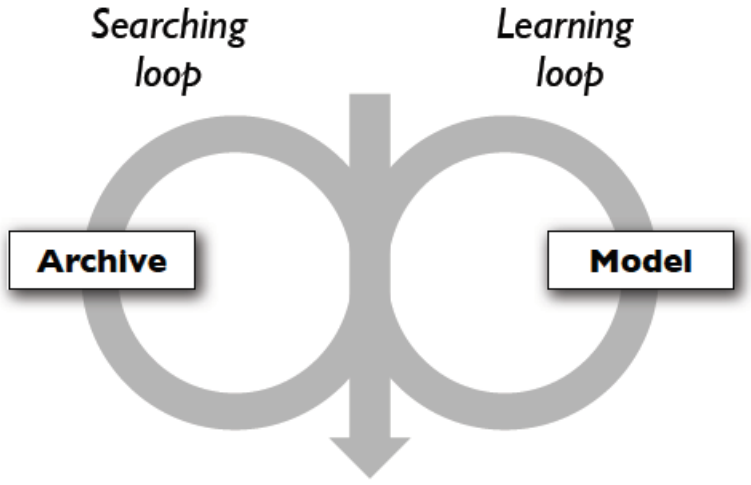


Рис. 3. Схема работы алгоритма *Bonesa*

Такой подход к настройке параметров помогает значительно сократить вычислительные затраты на мета-оптимизацию. Тестируются только перспективные стратегий, после чего мета-модель обновляется. Обновление происходит на каждой итерации мета-оптимизации. В итоге, мы получим довольно точное представление о поведении базового алгоритма.

Тем не менее, определение эффективности стратегии потребует нескольких дорогостоящих запусков базового алгоритма. Для уменьшения числа повторных запусков стратегий, авторы работы использовали фильтр Гаусса [40, 41], который уменьшает зашумленность функции эффективности. Для сравнения эффективности двух стратегий неявно используются гонки [24]: число перезапусков стратегий постепенно увеличивается до тех пор, пока не будет определена лучшая.

Метод *Bonesa* позволяет проводить многокритериальную настройку параметров для разных типов решаемых задач. В отличие от методов, использующих взвешенную сумму показателей эффективности [42], метод *Bonesa* использует подход метода *SPEA2* [43] для построения Парето-фронта. Таким образом, базовый алгоритм можно сразу настраивать под широкий спектр задач. Это означает, что результатом настройки будет набор стратегий, эффективных для разных классов решаемых задач.

Метод *Bonesa* начинает работу со случайной генерации K стратегий. Каждая из них однократно тестируется на каждом типе тестовой задачи. Если всего тестовых задач n, то будет произведено $K \cdot n$ запусков базового алгоритма на первом этапе. Затем протестированные векторы и значения их эффективности добавляются в базу данных, и на их основании строится модель функции эффективности базового алгоритма. На этом завершается обучающий цикл и запускается цикл поиска на основе метода *REVAC* [39, 44], в котором с использованием полученных из мета-модели рекомендаций, генерируются и тестируются новые векторы.

3.6.3 Метод *SEPaT*

Метод *SEPaT* (*Simple Evolutionary Parameter Tuning*) разработан в работе [45] и, скорее, является подходом к настройке параметров, поскольку в качестве мета-алгоритма может использоваться любой мета-оптимизатор (в данной работе использовались методы *Differential Evolution* [46] и *Particle Swarm Optimization* [47]).

Основными преимуществами метода *SEPaT* перед дургими методами настройки авторы называют методику кодирования и сравнения стратегий базового алгоритма, включающих и структурные параметры.

Индивидуумы мета-алгоритма представляют собой векторы параметров базового алгоритма, который тестируется T раз на наборе из F тестовых функций ($F \geq 1$). Эффективность стратегии вычисляется на основе среднего и стандартного отклонений показателя эффективности алгоритма при оптимизации всех рассматриваемых функций. Это означает, что для вычисления эффективности используется $2 \times F$ значений: F средних отклонений и F стандартных отклонений.

Полученные значения используются для сравнения стратегий по следующему принципу. Данный вектор параметров лучше другого вектора, если он показывает лучшую эффективность базового алгоритма при решении тестовых задач. В случае ничьей, победитель выбирается при помощи теста Велча (*Welch's t-test* [48]) по формуле

$$R = \sum_{i=1}^{F} \frac{\bar{X}_a^i - \bar{X}_b^i}{\sqrt{\frac{(\bar{s}_a^i)^2 - (\bar{s}_b^i)^2}{T}}},$$

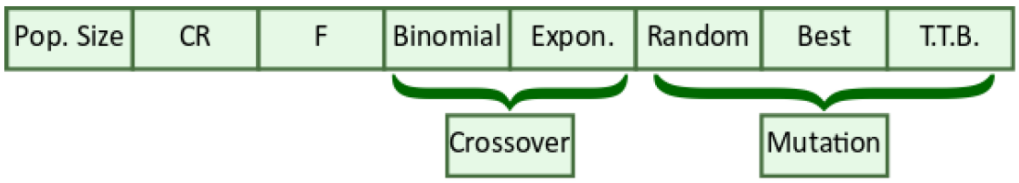
где a и b - два сравниваемых вектора параметров, $\bar{X}^i$ и s^i - среднее и стандартное отклонения показателя эффективности базового алгоритма при решении i задачи T раз. В случае минимизации, стратегия a лучше стратегии b, если $R < 0$.

При необходимости сравнения большего числа стратегий, каждая из них сравнивается с каждой, тем самым определяется число побед и поражений всех стратегий. Стратегия с наименьшим числом поражений признается лучшей, а в случае равенства числа проигрышей, сравнивается число выигрышей. Таким образом, найденная лучшая стратегия будет в среднем эффективной для всех рассматриваемых функций, что не гарантирует максимальную эффективность базового алгоритма при решении каждой из тестовых задач. Больше информации о данном методе можно найти в исследованиях [49, 50], где он сравнивается с методами гонок [24] и *ParamILS* [51].

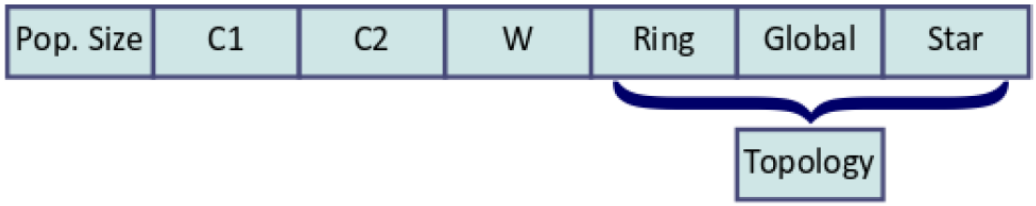
Метод *EMOPaT* (*Evolutionary Multi-Objective*), предложенный в работе [45], является многокритериальной модификацией метода *SEPaT*. Метод *EMOPaT* разработан с целью поиска лучших стратегий базового алгоритма с учетом особенностей решаемых задач и нескольких показателей эффективности стратегии. Такой подход позволяется больше узнать о поведении базового алгоритма, семантике его параметров и их влиянии на эффективность решения той или иной задачи путем анализа Парето-фронта. В качестве мета-алгоритма использовался *NSGA-II* [37]. Анализ по Парето-фронту зависимости показателей эффективности базового алгоритма от его параметров производится с помощью простого и эффективного подхода, который называется *innovization (innovation through optimization)* [52].

Поскольку метод *EMOPaT* является многокритериальным методом настройки, этот метод позволяется получить множество стратегий алгоритма, эффективных в тех или иных ситуациях по разным показателям эффективности. В работе [39] показано, что хотя *meta-EA* алгоритмы являются более эффективными по сравнению с *SPO* и *REVAC*, они не дают представления о пространстве параметров и поведении базового алгоритма. Метод *EMOPaT* позволяет исправить этот недочет. Больше информации об алгоритме и результаты исследований можно найти в работе [53].

В работе [45] также предложена методика кодирования структурных параметров алгоритма для дальнейшей настройки. Для примера рассмотрим метод *PSO*, в котором следует выбрать топологию соседства частиц. В векторе параметров алгоритма под каждую топологию соседства частиц выделяется отдельная переменная, которая варьируется мета-алгоритмом, как и численные параметры алгоритма. На рисунке 4 представлены закодированные стратегии алгоритмов *DE* и *PSO*. Отметим, что все структурные параметры базового алгоритма нормированы в диапазоне [0; 1]. Перед использованием того или иного значения структурного параметра выполняется дискретизация (линейное преобразование) величины структурного параметра.



а) метод DE



б) метод PSO

Рис. 4. Кодирование параметров

При использовании метода *EMOPaT* для выявления влияния различных параметров на показатели эффективности базового алгоритма, векторы параметров кодируются по схеме, представленной на рисунке 5.

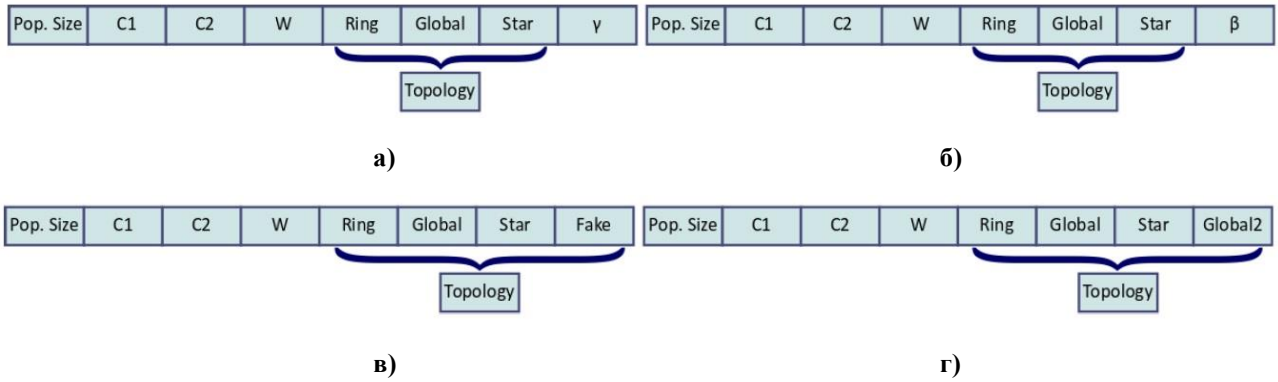


Рис. 5. Кодирование параметров PSO для выявления: "бесполезного" численного параметра (а); "вредного" численного параметра (б); худшего структурного параметра (в); эквивалентных структурных параметров (г).

В ходе эксперимента варьирование численного параметра γ не оказало никакого влияния на эффективность базового алгоритма *PSO*. "Вредным" численным параметром оказался β, увлечение которого приводило к ухудшению эффективности базового алгоритма. В ходе эксперимента намеренно добавили топологию *fake,* при которой сильно снижалась эффективность алгоритма. В результате новая топология была исключена уже через две итерации мета-оптимизации. Использование эквивалентных структурных параметров никак не отразится на производительности алгоритма. Для проведения эксперимента в вектор параметров намеренно была добавлена топология *global2,* которая показывала очень схожие результаты эффективности с топологией *global.* Метод *EMOPaT* успешно выявил все требуемые параметры алгоритма.

Более подробные результаты исследований можно найти в главе 4 работы [45]. Ана-

логичный подход был предложен в работе [54], авторы которой показали возможность определения худшего параметра алгоритма с использованием методов *ParamILS* и *REVAC*.

4. Настройка с учетом конфигурации вычислительной системы

4.1. Использование мультипроцессорных систем

В работе [55] предлагается методика настройки параметров с использованием многопроцессорных системы в режиме реального времени. Метод *ReACT* (*Real-time Algorithm Configuration through Tournamens*) позволяет одновременно тестировать несколько стратегий алгоритма при помощи техники гонок.

При запуске мета-алгоритма задаются следующие величины: число доступных процессоров n, число тестируемых стратегий, минимальное число побед для исключения стратегии из гонок. Имеется возможность задать начальный набор векторов параметров алгоритма ("теплый" старт), а также сгенерировать их случайным образом ("холодный старт"). Хоть "холодный" старт и может привести к снижению производительности в начале поиска, но позволяет избежать потенциального смещения в сторону заданных пользователем начальных стратегий.

При необходимости решения базовой задачи, на каждом из n доступных процессоров запускается экземпляр базового алгоритма. Это значит, что в каждый момент времени тестируется n стратегий алгоритма из общего набора. По завершении какого-либо из базовых алгоритмов, он отсылает остальным время решения поставленной задачи. Таким образом, после завершения всех запущенных базовых алгоритмов определяется стратегия-победитель. Имеется возможность ограничить время решения базовой задачи. В таком случае, если по истечении указанного времени ни один из базовых алгоритмов не завершится, победителем будет выбрана стратегия с лучшим достигнутым значением базовой целевой функции.

После завершения итераций мета-алгоритма обновляется число выигрышей победившей стратегии, худшие стратегии исключаются из дальнейших состязаний. К исключению приводит превышение стратегией заданного числа проигрышей.

Метод *ReACT* больше склоняется к диверсификационному поиску. Случайная генерация новых стратегий не позволит мета-алгоритму "упасть" в локальный оптимум. При этом в памяти метода всегда будет находится несколько эффективных стратегий алгоритма.

Метод *ReACT* является простым методом, направленным на эффективное использование многопроцессорных систем. Очевидно, что выбор стратегий для гонок можно производить различными способами, например, строить мета-модели или использовать эволюционные техники, усилив таким образом интенсификационную составляющую поиска.

4.2. Настройка с ограничением вычислительных ресурсов

Методы мета-оптимизации, как правило, ищут оптимальную стратегию алгоритма для конкретной конфигурации вычислительной системы, на которой работает настраиваемый

алгоритм. При изменении доступных вычислительных ресурсов, оптимальные параметры целесообразно подобрать заново. Так, например, использование ЭВМ малой мощности требует оптимального использования вычислительных ресурсов, в то время как мощные вычислительные системы позволяют сосредоточиться на обширных исследованиях.

Если об имеющихся вычислительных ресурсах ничего не известно, то мета-оптимизация может быть проведена несколько раз - для различных конфигураций. Очевидно, что это займет много времени. Аналогичная ситуация возникает, если подбор оптимальных параметров должен вестись с учетом различных возможных временных и вычислительных ограничений.

В работе [56] представлен новый подход, позволяющий получить оптимальную стратегию базового алгоритма для вычислительных систем различной мощности. В качестве алгоритма настройки используется эволюционный алгоритм. Каждый индивидуум в популяции мета-уровня определяется конкретным набором параметров эволюционного алгоритма базового уровня. Значение фитнес-функции определяется эффективностью базового алгоритма при выбранной стратегии.

Основная идея метода состоит в поиске множества векторов параметров, пригодных для различных конфигураций ЭВМ. Достигается это путем изменения правил выбора на мета-уровне для выявления стратегий, оптимальных для разных вычислительных систем, вне зависимости от текущей конфигурации. Идея была взята из метода *NSGA-II* [37], в котором предложено ранжирование стратегий базового алгоритма. В работе [56] данный подход назван *Flexible Budget Method* . Подразумевается следующий порядок действий.

- Задаем произвольную популяцию векторов параметров.
- Оцениваем каждый вектор параметров, применив его к алгоритму базового уровня для заранее определенной вычислительной мощности n_{max}(предпочтительна максимально возможная). Прогоняем базовый алгоритм k раз для учета стохастичности алгоритма.
- Запоминаем полученное для каждого вектора параметров наилучшее значение показателя эффективности (усредненное).
- Для каждой возможной конфигурации ЭВМ (конфигурация определяет ограничение числа итераций), находим эффективную стратегию и присваиваем ей первый ранг ($r = 1$).
- Удаляем из популяции особи, соответствующие найденным стратегиям и повторяем действия. Полученные векторы параметров относим к векторам второго ранга ($r = 2$).
- Повторяем действия пока не останется особей в популяции векторов параметров.

В отличие от классической мета-оптимизации, где векторы параметров выбирают по лучшему достигнутому значению эффективности базового алгоритма, в данном случае наиболее предпочтительными считаются векторы с меньшими рангами. На рисунке 7 можно видеть, что предпочтительным является вектор параметров, обозначенный сплош-

ной линией. Ему присваивается первый ранг. Вектор параметров, обозначенный пунктирной линией, показывает лучшие значения целевой функции после ста шестидесяти итераций. Ему тоже присваиваем первый ранг.

Теперь, имея ограничения по вычислительной мощности, например, по числу испытаний, можно сделать вывод об оптимальности того или иного вектора параметров. Из рисунка 6 видим, что при ограничении в сто итераций, выгодно использовать вектор параметров, обозначенный сплошной линией.

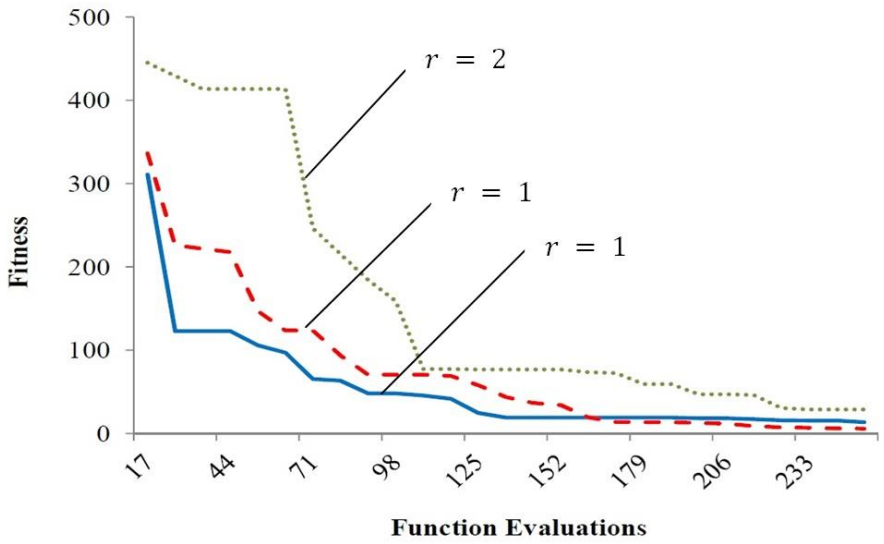


Рис.6. Поведение алгоритма при различных стратегиях: r – ранг вектора параметров

Заключение

В данной работе проведен обзор методов настройки параметров алгоритмов оптимизации. В отличие от управления параметрами, настройка дает возможность накопления и повторного использования информации об эффективности базового алгоритма. Такой подход позволяет не только определять максимально эффективные стратегии, но и получить представление о поведении настраиваемого алгоритма, об устойчивости стратегий к вариации задач, решаемых базовым алгоритмом.

В работе рассмотрено несколько классификаций методов мета-оптимизации. Классификация Эйбена определяет два подхода к настройке параметров: однократная настройка и перманентная настройка. В иных работах методы настройки делятся на итеративные и не итеративные. Структура работы основана на классификации методов настройки по принципу действия и цели настройки. Таким образом, выделено четыре основных вида методов настройки: выборочные, мета-стохастические, скрининговые и методы, использующие мета-модели. Каждый подход к настройке параметров имеет свои преимущества и недостатки.

Выборочные методы являются простыми и не требуют значительных вычислительных затрат. Малая точность данных методов не позволяет подробно изучить настраиваемый алгоритм. Поскольку задача мета-оптимизации является второстепенной и нацелена на

увеличение эффективности базового алгоритма оптимизации, малая вычислительная сложность методов выборки в некоторых случаях оправдывает относительно низкую точность настройки базового алгоритма.

Мета-стохастические методы настройки хорошо проявили себя при необходимости максимально быстрого поиска эффективных стратегий базового алгоритма. Диверсификационная составляющая поиска у данных методов слабая, что привело к появлению большого числа их модификаций. Большинство современных методов настройки являются гибридными и сочетают в себе несколько различных подходов, что увеличивает их вычислительную сложность.

Скрининговые методы позволяют уменьшить число запусков стратегий при помощи определения наиболее перспективных стратегий для тестирования. Разработаны методы, позволяющие сравнивать огромное число стратегий в режиме "соревнования". Предусмотрена возможность управления числом тестов каждой стратегии по мере решения задачи мета-оптимизации. Скрининговые подходы часто используются для модификации мета-алгоритмов других классов.

Для настройки параметров целесообразным оказалось использование мета-моделей показателей эффективности базового алгоритма. Эти модели позволяют уменьшить общее число запусков базового алгоритма для настройки путем использования полученных из мета-модели значений эффективности алгоритма. При этом мета-модель уточняется с каждой итерацией мета-оптимизации, что при большом числе итераций даст пользователю довольно точную модель функции эффективности базового алгоритма.

В связи с тем, что настройка может проводится с учетом различных показателей эффективности алгоритма для разных групп решаемых задач, используют также многокритериальные методы настройки. С их помощью можно оценить устойчивость базового алгоритма к вариации решаемых задач и сразу настраивать алгоритм под различные типы решаемых задач. В качестве мета-алгоритмов предложены как многокритериальные методы оптимизации, так и специально разработанные для настройки параметров.

Имеются также работы, посвященные настройке параметров с учетом конфигурации вычислительной системы. Прежде всего, настройка может производится с учетом доступных вычислительных ресурсов, которых, например, может не хватать на подробное исследование базового алгоритма. Тогда все вычислительные ресурсы будут направлены на наискорейшее нахождение эффективной стратегии базового алгоритма. В связи с тем, что широкое распространение получили многопроцессорные вычислительные системы, в работе также рассмотрен метод эффективного использования многопроцессорных систем для настройки параметров.

Список литературы

[1]. Карпенко А.П. Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой: учебное пособие / А.П. Карпенко. М.: Издательство МГТУ им. Н.Э. Баумана. 2014. 446 с.

[2]. Smit S.K. Parameter tuning and scientific testing in evolutionary algorithms. Amsterdam: Vrije Universiteit. 2012. 195 с. Режим доступа: http://dspace.ubvu.vu.nl/bitstream/handle/1871/38404/dissertation.pdf (дата обращения 01.10.2015)

[3]. Eiben A. E., Michalewicz Z., Schoenauer M., Smith J. E. Parameter Control in Evolutionary Algorithms // Parameter Setting in Evolutionary Algorithms. Springer Verlag. 2007. Vol.54. P. 19–46. DOI: 10.1007/978-3-540-69432-8_2

[4]. Pedersen M. E. H., Chipperfield A. J. Tuning differential evolution for artificial neural networks // Hvass Laboratories Technical Report HL0803. 2008. 19 p. Режим доступа: http://www.hvass-labs.org/people/magnus/publications/pedersen08tuning.pdf (дата обращения 01.10.2015)

[5]. Myers R., Hancock E.R. Empirical modelling of genetic algorithms // Evolutionary computation. 2001. Vol. 9. No. 4. P. 461-493. DOI: 10.1162/10636560152642878

[6]. Taguchi G., Yokoyama Y. Taguchi methods: design of experiments. (Taguchi Methods Series). Book. 4. Amer Supplier Inst. 1993.

[7]. Adenso-Diaz B., Laguna M. Fine-tuning of algorithms using fractional experimental designs and local search // Operations Research. 2006. Vol. 54. No. 1. P. 99-114. Режим доступа: http://opim.wharton.upenn.edu/~sok/papers/a/AdensoDiaz-Laguna-OR.pdf (дата обращения 01.10.2015)

[8]. Pedersen M. E. H. Tuning & simplifying heuristical optimization. Thesis for the degree of Doctor of Philosophy. University of Southampton. 2010. 204 p. Режим доступа: http://eprints.soton.ac.uk/342792/1.hasCoversheetVersion/MEH_Pedersen_PhD_Thesis_2010.pdf (дата обращения 01.10.2015)

[9]. El-Beltagy M.A., Nair P.B., Keane A.J. Metalmodeling techniques for evolutionary optimization of computationally expensive problems: promises and limitations. // Genetic algorithms and classifier systems. (In, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99), Orlando, USA, 13 - 17 Jul 1999). 1999. P. 196 – 203. Режим доступа: http://eprints.soton.ac.uk/23624/1/elbe_99c.pdf (дата обращения 01.10.2015)

[10]. Jin Y. A comprehensive survey of fitness approximation in evolutionary computation // Soft computing. 2005. Vol. 9. Is. 1. P. 3-12. DOI: 10.1007/s00500-003-0328-5

[11]. Czarn A., MacNish C., Vijayan K., Turlach B., Gupta R. Statistical exploratory analysis of genetic algorithms // Evolutionary Computation. IEEE Transactions. 2004. Vol. 8. Is. 4. P. 405-421. DOI: 10.1109/TEVC.2004.831262

[12]. Ramos I.C.O., Goldbarg M.C., Goldbarg E.F.G., Neto A.D.D. Logistic regression for parameter tuning on an evolutionary algorithm // Proceedings of the IEEE Congress on Evolutionary Computation. (CEC 2005, 2-4 September 2005, Edinburgh, UK). IEEE. Poster Session 2. 2005. Vol. 2. P. 1061-1068. DOI: 10.1109/CEC.2005.1554808

[13]. François O., Lavergne C. Design of evolutionary algorithms - A statistical perspective // IEEE Transactions on Evolutionary Computation. 2001. Vol. 5. No. 2. P. 129-148. DOI: 10.1109/4235.918434

[14]. Coy S.P., Goldenet B.L., Runger G.C., Wasil E.A. Using experimental design to find effective parameter settings for heuristics // Journal of Heuristics. 2001. Vol. 7. No. 1. P. 77-97. DOI: 10.1023/A:1026569813391

[15]. Bartz-Beielstein T., Parsopoulos K.E., Vrahatis M.N. Analysis of particle swarm optimization using computational statistics // Proceedings of the International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2004). Chalkis. Greece. Wiley-VCH Verlag GmbH & Co. Weinheim. 2004. P. 34-37. Режим доступа: http://spotseven.de/wordpress/wp-content/papercite-data/pdf/bpv04.pdf (дата обращения 01.10.2015)

[16]. Lasarczyk C.W.G. Genetische Programmierung einer algorithmischen chemie. Dissertation zur Erlangung des Grades eines Doktors der Naturwissenschaften. Technische Universiteit Dortmund. 2007. 215 p. Режим доступа: https://eldorado.tu-dortmund.de/bitstream/2003/25029/1/diss_lasarczyk_gpac.pdf (дата обращения 01.10.2015)

[17]. Hutter F., Bartz-Beielstein T., Hoos H., Leyton-Brown K., Murphy K. Sequential model-based parameter optimization: An experimental investigation of automated and interactive approaches // Experimental Methods for the Analysis of Optimization Algorithms. Springer Berlin Heidelberg. 2010. P. 363-414. DOI: 10.1007/978-3-642-02538-9_15

[18]. Goldsman D., Nelson B.L., Schmeiser B. Methods for selecting the best system // Proceedings of the 23rd conference on Winter simulation. IEEE Computer Society. Washington. DC. USA. 1991. P. 177-186. DOI: 10.1109/WSC.1991.185613

[19]. Branke J., Chick S. E., Schmidt C. New developments in ranking and selection: an empirical comparison of the three main approaches // Simulation Conference, 2005 Proceedings of the Winter. IEEE. 2005. 10 p. DOI: 10.1109/WSC.2005.1574312

[20]. Schmeiser B. Simulation experiments // Handbooks in operations research and management science. Elsevier Science Publishers (North-Holland), Amsterdam. 1990. Vol. 2. P. 295-330.

[21]. Rinott Y. On two-stage selection procedures and related probability-inequalities // Communications in Statistics-Theory and methods. 1978. Vol. 7. No. 8. P. 799-811. DOI: 10.1080/03610927808827671

[22]. Hochberg Y., Tamhane A.C. Multiple comparison procedures. / Wiley Series in Probability and Statistics. USA. New York: John Wiley & Sons. 1987. 450 p. Режим доступа: http://dl.acm.org/citation.cfm?id=39892 (дата обращения 01.10.2015)

[23]. Kim S.H., Nelson B.L. A fully sequential procedure for indifference-zone selection in simulation // ACM Transactions on Modeling and Computer Simulation (TOMACS). 2001. Vol. 11. No. 3. P. 251-273. DOI: 10.1145/502109.502111

[24]. Maron O., Moore A.W. The racing algorithm: Model selection for lazy learners // Lazy learning. Springer Netherlands. 1997. P. 193-225. DOI: 10.1007/978-94-017-2053-3_8

[25]. Birattari M. Tuning metaheuristics: a machine learning perspective. Berlin: Springer. 2009. Vol. 197. DOI: 10.1007/978-3-642-00483-4

[26]. Balaprakash P., Birattari M., Stützle T. Improvement strategies for the F-Race algorithm: Sampling design and iterative refinement. // (4th International Workshop, HM 2007, Dortmund,

Germany, October 8-9, 2007.) Proceedings. Vol. 4771 of the series Lecture Notes in Computer Science. Hybrid Metaheuristics. Springer Berlin Heidelberg. 2007. P. 108-122. DOI: 10.1007/978-3-540-75514-2_9

[27]. Mercer R.E., Sampson J.R. Adaptive search using a reproductive meta-plan // Kybernetes. 1978. Vol. 7. No. 3. P. 215-228. DOI: 10.1108/eb005486.

[28]. Grefenstette J.J. Optimization of control parameters for genetic algorithms // IEEE Transactions on Systems, Man, and Cybernetics. 1986. Vol. 16. No. 1. P. 122-128. DOI: 10.1109/TSMC.1986.289288

[29]. Hutter F., Hoos H.H., Stützle T. Automatic algorithm configuration based on local search // AAAI. 2007. Vol. 7. P. 1152-1157. Режим доступа: http://www.aaai.org/Papers/AAAI/2007/AAAI07-183.pdf (дата обращения 01.10.2015)

[30]. Yuan B., Gallagher M. Combining Meta-EAs and racing for difficult EA parameter tuning tasks // Parameter Setting in Evolutionary Algorithms. Vol. 54 of the series Studies in Computational Intelligence. Springer Berlin Heidelberg. 2007. P. 121-142. DOI: 10.1007/978-3-540-69432-8_6

[31]. Hutter F., Hoos H.H., Leyton-Brown K. Sequential model-based optimization for general algorithm configuration / 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers // Learning and Intelligent Optimization. Vol. 6683 of the series Lecture Notes in Computer Science. Springer Berlin Heidelberg. 2011. P. 507-523. DOI: 10.1007/978-3-642-25566-3_40

[32]. Biau G. Analysis of a random forests model // Journal of Machine Learning Research. 2012. V. 3. Vol. 13. No. 1. P. 1063-1095. Режим доступа: http://arxiv.org/pdf/1005.0208 (дата обращения 01.10.2015)

[33]. Pedersen M.E.H., Chipperfield A.J. Simplifying particle swarm optimization // Applied Soft Computing. 2010. Vol. 10. No. 2. P. 618-628. DOI: 10.1016/j.asoc.2009.08.029

[34]. Mersmann O., Bischl B., Trautmann H., Preuss M., Weihs C., Rudolph G. Exploratory landscape analysis // Proceedings of the 13th annual conference on Genetic and evolutionary computation. (GECCO'11, July 12–16, 2011, Dublin, Ireland). New York, NY, USA: ACM. 2011. P. 829-836. DOI: 10.1145/2001576.2001690

[35]. Smit S.K., Eiben A.E., Szlávik Z. An MOEA-based Method to Tune EA Parameters on Multiple Objective Functions // Proceedings of the International Conference on Evolutionary Computation ICEC 2010. (Valencia, Spain, October 24 - 26, 2010). IJCCI (ICEC). 2010. P. 261-268. Режим доступа: http://www.researchgate.net/profile/SK_Smit/publication/221616389_An_MOEA-based_Method_to_Tune_EA_Parameters_on_Multiple_Objective_Functions/links/54ddf8820cf22a26721d9498.pdf (дата обращения 01.10.2015)

[36]. Dréo J. Using performance fronts for parameter setting of stochastic metaheuristics // GECCO '09 Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers. New York, NY, USA: ACM. 2009. P. 2197-2200. DOI: 10.1145/1570256.1570301

[37]. Deb K., Pratap A., Agarwal S., Meyarivan T. A fast and elitist multiobjective genetic al-

gorithm: NSGA-II // IEEE Transactions on Evolutionary Computation. 2002. Vol. 6. No. 2. P. 182-197. DOI: 10.1109/4235.996017

[38]. Nannen V., Eiben A.E. A method for parameter calibration and relevance estimation in evolutionary algorithms // GECCO '06 Proceedings of the 8th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM. 2006. P. 183-190. DOI: 10.1145/1143997.1144029

[39]. Smit S.K., Eiben A.E. Comparing parameter tuning methods for evolutionary algorithms // CEC'09. IEEE Congress on Evolutionary Computation. 2009. P. 399-406. DOI: 10.1109/CEC.2009.4982974

[40]. Shapiro L.G., Haralick R.M. Computer and robot vision. Vol. 1. Addison Wesley. 1992. 672 p.

[41]. Branke J., Schmidt C., Schmec H. Efficient fitness estimation in noisy environments // Proceedings of genetic and evolutionary computation. 2001. 8 p. Режим доступа: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.7071&rep=rep1&type=pdf (дата обращения 01.10.2015)

[42]. Smit S.K., Eiben A.E. Parameter tuning of evolutionary algorithms: Generalist vs. specialist // Applications of evolutionary computation. Springer Berlin Heidelberg. 2010. P. 542-551. DOI: 10.1007/978-3-642-12239-2_56

[43]. Zitzler E., Laumanns M., Thiele L. Spea2: Improving the strength Pareto evolutionary algorithm. Zurich, Switzerland: Swiss Federal Institute of Technology. 2001. 21 p. Режим доступа: http://www.kddresearch.org/Courses/Spring-2007/CIS830/Handouts/P8.pdf (дата обращения 01.10.2015)

[44]. Eiben A.E., Schut M.C., de Wilde A.R. Is self-adaptation of selection pressure and population size possible? - A case study // Parallel Problem Solving from Nature-PPSN IX. Springer Berlin Heidelberg. 2006. P. 900-909. DOI: 10.1007/11844297_91

[45]. Ugolotti R. Meta-optimization of Bio-inspired Techniques for Object Recognition. Diss. Università di Parma. Dipartimento di Ingegneria dell'Informazione. 2015. 158 p. Режим доступа: http://dspace-unipr.cineca.it/bitstream/1889/2827/1/TesiDottoratoUgolottiOnline.pdf (дата обращения 01.10.2015)

[46]. Storn R., Price K. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. // Journal of Global Optimization. Kluwer Academic Publishers. 1997. Vol. 11. Is. 4. P. 341-359. DOI: 10.1023/A:1008202821328

[47]. Kennedy J., Eberhart C. A New Optimizer Using Particle Swarm Theory // Micro Machine and Human Science. MHS '95. Proceedings of the Sixth International Symposium. IEEE. 1995. P. 39 - 43 Режим доступа: http://www.ppgia.pucpr.br/~alceu/mestrado/aula3/PSO_2.pdf (дата обращения 01.10.2015).

[48]. Ruxton G.D. The unequal variance t-test is an underused alternative to Student's t-test and the Mann–Whitney U test // Behavioral Ecology. 2006. Vol. 17. No. 4. P. 688-690. DOI: 10.1093/beheco/ark016

[49]. Ugolotti R., Nashed Y.S.G., Mesejo P., Cagnoni S. Algorithm configuration using GPU-based metaheuristics // GECCO '13 Companion Proceedings of the 15th annual conference com-

panion on Genetic and evolutionary computation. New York, NY, USA: ACM. 2013. P. 221-222. DOI: 10.1145/2464576.2464682

[50]. Ugolotti R., Mesejo P., Nashed Y.S.G., Cagnoni S. GPU-Based Automatic Configuration of Differential Evolution: A Case Study // Progress in Artificial Intelligence. Vol. 8154 of the series Lecture Notes in Computer Science. Springer Berlin Heidelberg. 2013. P. 114-125. DOI: 10.1007/978-3-642-40669-0_11

[51]. Hutter F., Hoos H.H.; Leyton-Brown K.; Stützle T. ParamILS: an automatic algorithm configuration framework // Journal of Artificial Intelligence Research. 2009. Vol. 36. No. 1. P. 267-306. Режим доступа: http://www.aaai.org/Papers/JAIR/Vol36/JAIR-3606.pdf (дата обращения 01.10.2015)

[52]. Deb K., Srinivasan A. Innovization: Innovating design principles through optimization // Genetic and Evolutionary Computation of the 8th annual Conference, GECCO 2006, Proceedings, (Seattle, Washington, USA, July 8-12, 2006). New York, NY, USA: ACM. 2006. P. 1629-1636. DOI: 10.1145/1143997.1144266

[53]. Ugolotti R., Cagnoni S. Analysis of evolutionary algorithms using multi-objective parameter tuning // GECCO '14 Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation. New York, NY, USA: ACM. 2014. P. 1343-1350. DOI: 10.1145/2576768.2598226

[54]. Montero E., Riff M., Pérez-Caceres L., Coello Coello C.A. Are state-of-the-art fine-tuning algorithms able to detect a dummy parameter? // Parallel Problem Solving from Nature-PPSN XII. (12th International Conference, Taormina, Italy, September 1-5, 2012, Proceedings, Part I). Springer Berlin Heidelberg. 2012. P. 306-315. DOI: 10.1007/978-3-642-32937-1_31

[55]. Fitzgerald T., Malitsky Y., O'Sullivan B., Tierney K. ReACT: Real-Time Algorithm Configuration through Tournaments // Proceedings of the Seventh Annual Symposium on Combinatorial Search. 2014. Режим доступа: http://www.aaai.org/ocs/index.php/SOCS/SOCS14/paper/download/8910/8880 (дата обращения 01.10.2015)

[56]. Branke J., Elomari J. A. Meta-optimization for parameter tuning with a flexible computing budget // GECCO '12 Proceedings of the 14th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM. 2012. P. 1245-1252. DOI: 10.1145/2330163.2330336