Университет ИТМО

Факультет информационных технологий и программирования
Кафедра компьютерных технологий

Рост Аркадий Юрьевич

Адаптивная настройка параметров эволюционных алгоритмов с помощью обучения с подкреплением

Научный руководитель: зав. каф. ТП, д. т. н., проф. А. А. Шалыто

Санкт-Петербург
2015

Содержание

Введение

Эффективность работы эволюционного алгоритма зависит от выбора значений его параметров. Подбор параметров может осуществляться до запуска эволюционного алгоритма. Однако оптимальные значения параметров могут изменяться в ходе работы алгоритма. Поэтому необходим метод адаптивной настройки параметров в процессе оптимизации.

Значения параметров эволюционного алгоритма лежат в заданном интервале значений. Задачу выбора значений параметров дискретизируют, разделяя диапазон допустимых значений параметра на интервалы. Разбиение на интервалы может производиться до запуска алгоритма и не меняться в процессе его работы. Однако изменение разбиения во время работы способствует улучшению работы алгоритма.

Существуют алгоритмы адаптивной настройки параметров эволюционного алгоритма, в которых вероятность выбора значения параметра пропорциональна эффективности его применения. Одним из таких алгоритмов является *earpc*. В методе *earpc* интервал допустимых значений разбивается в ходе работы алгоритма. Также существует метод настройки параметров эволюционного алгоритма с помощью обучения с подкреплением. Однако в данном подходе разбиение диапазона допустимых значений производится до запуска алгоритма.

В данной работе предлагаются два метода адаптивной настройки параметров эволюционного алгоритма. Один из них является улучшением существующего метода настройки параметров с помощью обучения с подкреплением за счет разбиения диапазона допустимых значений параметра в ходе работы алгоритма при помощи алгоритма *earpc*. Второй метод основан на применении Q-обучения с адаптивным выделением множества действий агента.

Глава 1. Обзор предметной области

1.1. Эволюционные алгоритмы

Эволюционный алгоритм является методом решения задач оптимизации. Данный подход основан на идеях, заимствованных из биологической эволюции: естественный отбор, мутация, скрещивание и наследование признаков. Каждая итерация алгоритма характеризуется набором особей, называемым поколением. Начальное поколение обычно формируется случайным образом. На множестве особей вводят функции приспособленности, чтобы количественно оценивать, насколько данная особь близка к верному решению. Наиболее приспособленные особи имеют большую вероятность быть выбранными для создания нового поколения. При помощи оператора скрещивания (кроссовера) по двум особям текущего поколения создается новая особь для следующего поколения. Оператор мутации вносит в особь малые случайные изменения. Общая схема эволюционного алгоритма представлена на листинге 1.

Листинг 1 Общая схема эволюционного алгоритма

```
1: Создать начальное поколение
2: Вычислить значение функции приспособленности для каждой особи
3: while (условие останова эволюционного алгоритма не выполнено) do
4:     Выбирается подмножество особей текущего поколения
5:     Применяя операторы мутации и кроссовера к выбранным особям, создаются новые особи
6:     Вычисляется значение функции приспособленности для созданных особей
7:     Путем замены новыми особями наименее приспособленных в текущем поколении формируется
       новое поколение
8: end while
```

В качестве критерия останова часто используют следующие условия:

- найдено верное решение;
- достигнуто заданное количество поколений;
- превышено заданное время работы;

- превышено заданное число вычислений функции приспособленности;
- за заданное число поколений не произошло улучшение решения.

Эволюционные алгоритмы применяются для решения задач оптимизации, к которым точные алгоритмы не применимы. Стоит отметить, что эффективность работы эволюционного алгоритма сильно зависит от выбора значений его параметров, таких как вероятность мутации, вероятность скрещивания, число особей в поколении. Значения параметров зависят не только от эволюционного алгоритма, но и от решаемой задачи оптимизации. Подбор параметров может осуществляться до запуска эволюционного алгоритма. Однако оптимальные значения параметров могут изменяться в ходе работы алгоритма. Поэтому необходим метод адаптивной настройки параметров в процессе оптимизации.

1.2. Обзор существующих методов настройки параметров ЭА

Рассмотрим формальную постановку задачи адаптивной настройки параметров ЭА. Имеется набор $\{v_1, ..., v_n\}$ из n параметров ЭА, каждый из которых может принимать значения из некоторого дискретного набора или из непрерывного интервала значений. Целью алгоритма является выбор таких значений параметров v_i, чтобы ЭА работал наиболее эффективно.

Большинство алгоритмов адаптивной настройки параметров ЭА можно отнести к классу методов сопоставления вероятности (probability matching techniques), в которых вероятность выбора значения параметра пропорциональна эффективности его применения. В данной работе использовался один из наиболее эффективных алгоритмов данного класса, называемый *earpc*.

1.2.1. Метод *earpc*

В методе *earpc* выбор значений параметров происходит независимо друг от друга. В данном методе для выбора значения параметра диапазон допустимых значений делится во время работы алгоритма на два подинтервала.

Схема работы алгоритма *earpc* представлена на листине 2. Будем называть назначением параметров набор (v) подобранных значений параметров $(v_1, .., v_n)$. В ходе работы алгоритма каждому назначению параметров (v) сооветствует некоторый функционал качества $q((v))$. Выбор новых значений параметров осуществляется следующим образом. Ранее используемые назначения параметров разбиваются на два кластера c_1 и c_2, например, с помощью алгоритма *k-means*. Затем для каждого параметра v_i интервал его допустимых значений разбивается на два подинтервала. Для этого необходимо выбрать подходящую точку разбиения. Для этого все ранее используемые назначения параметра v_i сортируются по возрастанию. В качестве кандидатов на точку разбиения рассматриваются средние значения между двумя соседними назначениями в полученной упорядоченной последовательности. Для каждого кандидата s на точку разбиения множество ранее используемых назначений разбивается в соответствии с s на два множества p_1 и p_2. Во множестве p_1 находятся все ранее используемые назначения параметра v_i, меньшие или равные s, а в множестве p_2 находятся все ранее используемые назначения параметра v_i, большие s. Обозначим $c_i(p_j)$ – подмножество p_j, где $i, j \in \{1, 2\}$, соответствующее кластеру c_i. Для каждого разбиения по формуле (1.1) считается *энтропия*. В качестве итоговой точки разбиения s выбирается та, при которой полученная энтропия минимальна. Для множеств p_1 и p_2 считается среднее качество Q_1 и Q_2 соответственно. Множеству p_1 соответствует интервал $[v_{min}, s]$, а множеству p_2 – интервал $(s, v_{max}]$, где v_{min} и v_{max} нижняя и верхняя границы диапазона допустимых значений параметра v_i соответственно. Из двух данных интервалов значений случайным образом выбирается один, при этом

вероятность выбора первого интервала пропорциональна Q_1, а второго – Q_2. Затем значение параметра v_i случайным образом выбирается из соответствующего множества.

$$
\begin{aligned}
e_{p_1} &= -\frac{|c_1(p_1)|}{|p_1|} ln(\frac{|c_1(p_1)|}{|p_1|}) - \frac{|c_2(p_1)|}{|p_1|} ln(\frac{|c_2(p_1)|}{|p_1|}), \\
e_{p_2} &= -\frac{|c_1(p_2)|}{|p_2|} ln(\frac{|c_1(p_2)|}{|p_2|}) - \frac{|c_2(p_2)|}{|p_2|} ln(\frac{|c_2(p_2)|}{|p_2|}), \\
H &= \frac{|p_1|}{|c_1|} e_{p_1} + \frac{|p_2|}{|c_2|} e_{p_2}
\end{aligned}
\tag{1.1}
$$

Листинг 2 Алгоритм *earpc* в случае деления на два подинтервала.

1: Ранее выбранные назначения параметров $\{(v_1, .., v_n)\}$ разбиваются на два кластера c_1 и c_2 с помощью алгоритма *k-means*.
2: **for** параметр v_i **do**
3: Отсортировать назначения параметров по значению i-ого
4: $H_{best} \leftarrow \infty$
5: **for** точка разбиения $s = \frac{v_{ij}+v_{i(j+1)}}{2}$ **do**
6: Разбить назначения в соответствии с точкой разбиения s на множества p_1 и p_2
7: Расчитать энтропию H разбиения по точке s по формуле (1.1)
8: **if** $H_{best} < H$ **then**
9: $H_{best} \leftarrow H$
10: Запомнить множества p_1 и p_2
11: **end if**
12: **end for**
13: $Q_1 = \frac{1}{|p_1|} \sum\limits_{\mathbf{v} \in p_1} q(\mathbf{v})$, $Q_2 = \frac{1}{|p_2|} \sum\limits_{\mathbf{v} \in p_2} q(\mathbf{v})$
14: Случайным образом выбрать интервал значений, при этом вероятность выбора первого пропорциональна Q_1, а второго – Q_2
15: Случайным образом выбрать значение параметра v_i из выбранного интервала
16: **end for**

Недавно Karafotias et al. был предложен эффективный метод настройки параметров ЭА с помощью обучения с подкреплением. Однако сравнение эффективности применения данного метода и алгоритма earpc не проводилось. В данной работе проводится сравнение данных подходов, а также предлагается новый метод адаптивного выбора параметров ЭА. Далее рассматриваются основные принципы работы алгоритмов обучения с подкреплением и метод, предложенный Karafotias et al.

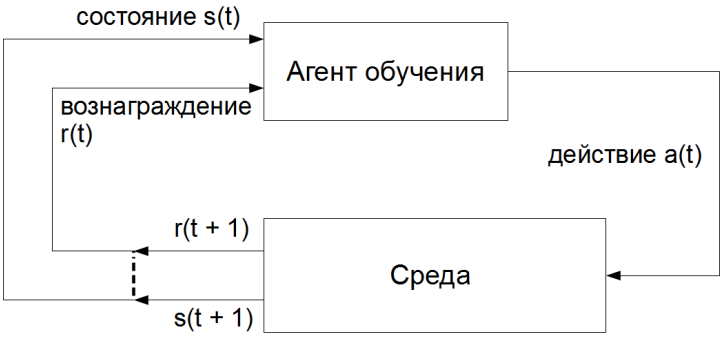

Рис. 1.1: Схема алгоритма обучения с подкреплением.

1.2.2. Обучение с подкреплением

Алгоритмы обучения с подкреплением часто используются для выбора стратегий поведения в интерактивной среде. Большинство таких алгоритмов не требуют заранее подобранных тестовых примеров, так как их обучение происходит одновременно с применением накопленного опыта.

Принцип работы алгоритма обучения с подкреплением представлен на схеме 1.1. У агента есть некоторый набор возможных действий. На каждом шаге алгоритма агент воздействует на среду, которая находится в некотором состоянии, выбирая одно из возможных действий и применяя его к среде. В следствие этого среда может перейти в новое состояние. За выбор действия агент получает численную награду. Задачей агента является максимизация суммарной награды. Действие, выбранное агентом, определяет не только полученную награду, но и состояние, в которое перейдет среда после его применения.

Задачу обучения с подкреплением в большинстве случаев можно описать как *марковский процесс принятия решений*. Для этого необходимо определить:

- дискретное множество состояний среды S;
- дискретное множество действий агента A;
- функцию награды $R : S \times A \to \mathbb{R}$;
- функцию переходов $T : S \times A \times S \to \mathbb{R}$ При этом $T(s, a, s')$ определяет вероятность перехода из состояния s в состояние s' после применения действия a.

Выделяют класс алгоритмов обучения с подкреплением, строящих модель среды, которые используют функцию награды R и функцию переходов T для определения стратегии поведения. В частности, возможны стратегии в которых алгоритм будет получать незначительную награду в течение некоторого времени, чтобы достичь некоторого состояния среды, которому соответствует большая ожидаемая награда. В рамках данной работы такие алгоритмы не рассматривались.

1.2.2.1. Q-обучение

Алгоритм Q-обучения относится к классу алгоритмов обучения с подкреплением не строящих модель среды. Псевдокод алгоритма представлен на листинге 3. Во время работы алгоритма аппроксимируется функция полезности $Q : S \times A \to \mathbb{R}$, которая описывает ожидаемую награду за действие a в состоянии s. Для расчета значений Q обычно используют TD-обучение(temporal difference learning). При этом значения Q изменяются по формуле $Q(s,a) = Q(s,a) + \alpha(\gamma Q(s',a') - Q(s,a))$, где α – скорость обучения, γ – дисконтный фактор.

Выбор действия определяется стратегией исследования среды. Одна из самых простых стратегий - *жадная* заключается в том, чтобы выбирать действие, за которое самое большое ожидаемое вознаграждение, т.е. $\arg\max_a \{Q(s,a)\}$. Однако в таком случае агент склонен выбирать локально максимальное значение награды, недостаточно исследовав среду. Для улучшения жадной стратегии можно выбирать с вероятностью ϵ случайное действие, иначе – действие с максимальной ожидаемой наградой. Такая стратегия называется ϵ-*жадной*. При этом значение ϵ может меняться во время работы алгоритма, что позволяет перейти от исследования среды к применению накопленного опыта.

Листинг 3 Алгоритм Q-обучения с ε-жадной стратегией исследования среды

Вход: ε — вероятность выбора случайного действия; α — скорость обучения; γ — дисконтный фактор.

Инициализировать $Q(s, a)$ для всех $s \in S$, $a \in A$
while (не достигнуто условие останова) **do**
  Получить состояние среды s
  $p \leftarrow$ случайное вещественное число $\in [0, 1]$
  if $(p \leq \varepsilon)$ **then**
    $a \leftarrow \arg\max_a Q(s, a)$
  else
    $a \leftarrow$ случайное действие $\in A$
  end if
  Применить действие a к среде
  Получить от среды награду r и состояние s'
  $Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$
end while

1.2.3. Настройка параметров ЭА как задача для обучения с подкреплением

Рассмотрим задачу выбора параметров эволюционного алгоритма, как задачу, решаемую при помощи обучения с подкреплением. В качестве среды выступает эволюционный алгоритм. Агент совершает действие – выбор значений настраиваемых параметров, таких как вероятность мутации или кроссовера. Затем среда генерирует следующее поколение особей, используя выбранные значения параметров ЭА, и переходит в новое состояние. Награда, возвращаемая агенту средой, является некоторой функцией от значений оптимизируемой функции – функции приспособленности, вычисленных на особях текущего и предыдущего поколений.

Значения параметров ЭА лежат в заданном интервале значений. Таким образом, чтобы установить значение параметра ЭА, агент дожнен выбрать некоторое значение из заданного интервала. Обычно эту задачу дискретизируют, разделяя диапазон допустимых значений параметра на подинтервалы. Каждый из подинтервалов соответствует действию агента. Совершив действие – выбор подинтервала, агент в качестве значения параметра устанавливает случайное значение из выбранного подинтервала. Разбиение на интервалы можно делать априорно, т.е. разбиение диапазона значений происходит до запуска алгоритма и не менятся в процессе его работы. Однако для некоторых методов адаптивной настройки пара-

метров было показано, что изменение разбиения во время работы способствует улучшению работы алгоритма. В частности, значение параметра можно подобрать тем точнее, чем меньше шаг разбиения. В то же время это усложняет задачу выбора оптимального подинтервала. Существующие методы настройки параметров ЭА с помощью обучения с подкреплением используют априорное разбиение.

1.2.4. Метод, предложенный Karafotias et al.

На конференции *GECCO 2014* Karafotias et al. предложили метод подбора параметров ЭА на основе обучения с подкреплением, в котором множество состояний среды формируется во время работы алгоритма, а множество действий задается до запуска алгоритма.

В качестве алгоритма обучения с подкреплением используется Q-обучение с ϵ-жадной стратегией исследования среды. Предположим, что число настраиваемых параметров равно k. Диапазон допустимых значений настраеваемого параметра v_i делится на m_i интервалов до начала работы алгоритма. Действием является выбор интервалов, из которых будет случайно выбрано значение параметра, для всех настраиваемых параметров и обозначается как $b_1, b_2, \ldots, b_k$, где $b_i, 0 < b_i <= m_i$ – номер интервала для параметра v_i. Таким образом число допустимых действий агента в каждом состоянии равно $\prod_{i=1}^{k} m_i$. Отметим, что действием агента осуществляется одновременный выбор значений для всех параметров.

Функции награды для ЭА, максимизирующего функцию приспособленности, задается формулой (1.2), где f_t – лучшее значение функции приспособленности, полученное на t-ой итерации.

$$R = c(\frac{f_{t+1}}{f_t} - 1) \tag{1.2}$$

Для ЭА, которые не ухудшают лучшее известное решение, функция награды неотрицательна. Стоит отметить, что значение функции приспособленности зачастую остается неизменным несколько итераций подряд. В самом деле, чтобы награда была положительная, необходимо улучшить луч-

шую известное значение функции приспособленности. Чтобы замедлить скорость обучения при нулевой награде менялся коэффициент скорости обучения α. А именно:

$$Q(s,a) = Q(s,a) + \alpha(r)(r + \gamma \max_{a'} Q(s',a') - Q(s,a))$$

$$\alpha(r) = \begin{cases} \alpha & \text{, при } r > 0 \\ \alpha_0 & \text{, иначе} \end{cases}$$

Стоит отметить, что $\alpha_0 \ll \alpha$.

Для выделения состояний среды используются следующие наблюдаемые характеристики ЭА:

- генетическое разнообразие;
- разнообразие значений функции приспособленности (среднеквадратичное отклонение);
- стагнация (число итераций без улучшения функции приспособленности);
- прирост значения функции приспособленности.

Множество состояний среды строится при помощи *UTree*, описанного далее. Отметим, что значения ожидаемой награды за действие $Q(s,a)$ при разделении состояния s копируется в новые состояния, поэтому значения ожидаемой награды в получившихся состояниях не изменяется.

1.2.4.1. *UTree*

Эффективность применения алгоритмов обучения с подкреплением экспоненциально уменьшается с увеличением числа возможных состояний среды. Однако в большинстве задач не все из них являются существенными. Одним из подходов, позволяющих уменьшить размерность множества состояний среды является объединение нескольких несущественных состояний. Таким образом множество состояний может быть разбито на несколько значимых состояний.

Одним из алгоритмов объединения состояний является алгоритм *UTree*. В алгоритме *UTree* строится дерево, листьями которого являются полученные в результате объединения состояния. Также существует вариант алгоритма *UTree*, применимый в случае непрерывного множества состояний. Отличие от алгоритма *UTree* для дискретного случая заключается в том, что он не требует изначального выделения состояний среды. Состояния среды выделяются автоматически в ходе работы алгоритма *UTree*.

Схема работы алгоритма представлена на листинге 4. Состояния среды выделяются на основании наблюдаемых параметров среды. По своей структуре алгоритм *UTree* представляет собой дерево решений в узлах которого стоят условия на параметры среды. Каждому листу соответствует состояние s среды алгоритма обучения с подкреплением, ожидаемое значение награды в котором обозначается как *V(s)*, где $V(s) = \max_a Q(s, a)$. Изначально в дереве существует лишь один лист, и таким образом у среды есть единственное возможное состояние s, для которого $V(s) = 0$. Алгоритм состоит из двух циклично повторяемых этапов: этапа сбора данных и этапа их обработки. На этапе сбора данных при помощи текущего построенного дерева по параметрам среды I определяется состояние среды алгоритма обучения с подкреплением s. Затем агент *жадно* выбирает действие a и сохраняет полученный кортеж *(I, a, I', r)*,где I – исходные параметры среды, a – выбранное действие, *I'* – параметры среды после применения действия a, r – награда, полученная агентом. Затем на основе полученного опыта агент обновляет значение *Q(s, a)*. На этапе обработки для каждого сохраненного кортежа вычисляется значение $q(I, a) = r + \gamma V(s')$ – значение ожидаемой награды после применения действия a к среде с параметрами I, где *s'* – состояние среды обучения с подкреплением, соответствующее параметрам среды *I'*. Для каждого состояния s при помощи критерия разбиения ищется точка разбиения. Если такая точка найдена, то состояние s разбивается на два новых состояния. Множество сохраненных кортежей *(I,*

a, I', r) распределяется по новым состояниям в соответствии с разбиением, затем обновляются функция награды и функция переходов.

Листинг 4 Алгоритм *UTree* для непрерывного случая с использованием Q-обучения

Фаза обучения

1: По параметрам среды I найти соответствующее состояние s, являющиеся листом дерева решений.
2: Выбрать действие (интервал значений параметра): $a = \arg\max_{a'} Q(s, a')$.
3: Применить выбранное действие к среде, получив награду r.
4: Сохранить переход (I, a, I', r) в состоянии s.
5: Обновить значение $Q(s,\ a)$ и $V(s) = \max_a Q(s, a)$.

Фаза разбиения

1: **for** состояния s **do**
2: **for** переход (I, a, I', r) в состоянии s **do**
3: $q(I, a) = r + \gamma V(s')$
4: **end for**
5: С помощью *критерия разбиения* определить параметр среды и его значение, по которому лучше разделить состояние.
6: **if** найдена точка разбиения **then**
7: Создать два новых состояния s_1 и s_2.
8: Распределить переходы состояния s по состояниям s_1 и s_2 в соответствии с разбиением.
9: Рассчитать $Q(s_1,\ a)$ и $Q(s_2,\ a)$ по сохраненным переходам.
10: Заменить состояние s в дереве решений, на вершину с детьми s_1 и s_2 и условием выбора, соответствующим точке разбиения.
11: **end if**
12: **end for**

1.2.4.2. Критерий типа Колмогорова-Смирнова

В статистическом анализе используют различные критерии однородности для проверки гипотезы о принадлежности двух независимых выборок одному закону распределения. Одним из наиболее используемых непараметрических критериев о проверке однородности двух эмпирических законов распределения является критерий однородности Смирнова.

Эмпирическая функция распределения является приближением теоретической функции распределения, построенное с помощью выборки из него. Пусть $\{X_i\}_{i=1}^n$ выборка из случайной величины X, объема n. Эмпирической функцией распределения случайной величины X называется случайная величина $F(x) = \frac{1}{n}\sum_{i=1}^{n} H(x - X_i)$, где H – функция Хевисайда. По сути заданная таким образом функция распределения в точке x равна частоте элементов выборки, не превосходящих x.

Критерий позволяет найти точку, в которой сумма накопленных ча-

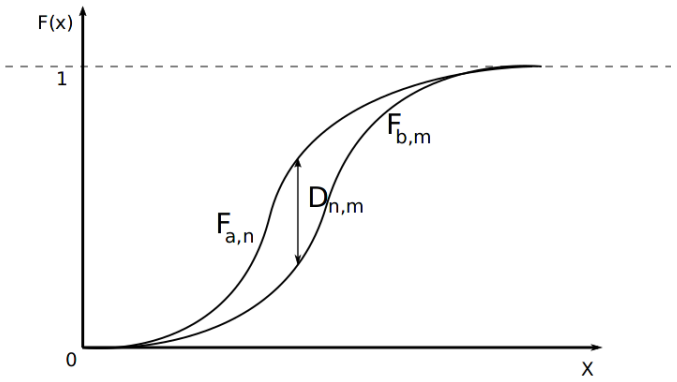

Рис. 1.2: График $F_{a,n}$ и $F_{b,m}$

стот расхождений наибольшая, и оценить достоверность этого расхождения. В качестве нулевой гипотезы H_0 принимается, что две исследуемые выборки подчиняются одному закону распределения случайной величины. Для двух независимых выборок a и b, объемами n и m соответственно, строятся эмпирические функции распределения $F_{a,n}$ и $F_{b,m}$. Затем считается значение $\sqrt{\frac{nm}{n+m}}D_{n,m}$, где $D_{n,m} = \sup\limits_x |F_{a,n}(x) - F_{b,m}(x)|$. Если рассчитанное значение превышает квантиль распределения Колмогорова K_α для заданного уровня значимости α, то нулевая гипотеза H_0 отвергается.

Глава 2. Разработанные методы настройки параметров ЭА

2.1. Цель работы

Существует метод адаптивной настройки параметров ЭА с помощью обучения с подкреплением, позволяющий адаптивно выделять состояния среды. Множество действий агента определяется разбиением диапазона допустимых значений параметра, которое задается до начала выполнения алгоритма. Также существуют методы настройки параметров ЭА, позволяющие адаптивно разбивать диапазон допустимых значений параметра.

В данной работе предлагается исследовать эффективность адаптивного выделения множества действий агента за счет разбиения диапазона допустимых значений параметра в ходе работы алгоритма. Целью исследований являлась разработка метода адаптивной настройки параметров эволюционного алгоритма с помощью обучения с подкреплением. Предлагаемый алгоритм на основе обучения с подкреплением должен формировать множество действий агента во время работы, адаптивно разбивая диапазон допустимых значений параметра.

2.2. Метод на основе *earpc* и *UTree*

Данный метод является объединением методов *earpc* (1.2.1) и *UTree* (1.2.4.1). В отличие от метода, предложенного Karafotias et al. (1.2.4), выбор значений параметров происходит с помощью алгоритма *earpc*. В процессе работы по наблюдаемым характеристикам ЭА строится дерево решений *UTree*. Алгоритм *earpc* выбирает значения параметров на основе сохраненных в листе переходов (I, a, I', r). Для оценки качества выбранного назначения параметров для алгоритма *earpc* используется награда, полу-

чаемая агентом. Таким образом, необходимо по наблюдаемым значениям ЭА найти лист дерева *UTree*. Затем значения параметров выбираются при помощи алгоритма *earpc*, используя переходы, хранящиеся в найденном листе.

При построении дерева *UTree*, необходимо определить способ разбиения листа на два состояния. Для применения критерий разбиения Колмогорова-Смирнова, для каждого перехода (I, a, I', r), сохраненного в листе, вычисляется значение $q(I, a) = r + \gamma V(s')$, где *s'* – лист, соответствующий характеристикам ЭА *I'*. При этом необходимо посчитать ожидаемую награду $V(s')$. В качестве $V(s')$ предлагается использовать математическое ожидание награды в листе *s'*. Алгоритм *earpc* разбивает диапазон значений параметра на два подинтервала, один из которых выбирается с вероятностью пропорциональной средней награде на подинтервале. Таким образом $V(s')$ вычисляется по формуле $V(s') = \sum_{i=1}^{2} \frac{Q_i^2}{Q_1+Q_2}$, где Q_1 и Q_2 среднее значение награды на первом и втором подинтервале соответственно.

Кроме того, после выделения новых состояний среды необходимо пересчитать значения ожидаемой награды для получившихся состояний. В методе, предложенном Karafotias et al., значения $Q(s, a)$ копировалось в новое состояние, поэтому значения ожидаемой награды в получившихся состояниях не изменялось. В предлагаемом необходимо значение ожидаемой награды меняется, поскольку при выделении нового состояния в соответствии с алгоритмом *UTree* множество переходов перераспределяется между получившимися состояниями.

2.3. Метод с адаптивным выделением множества действий

Также в данной работе предлагается метод адаптивной настройки параметров ЭА с помощью Q-обучения с адаптивным выделением множества действий. В данном подходе действие определяется аналогично методу, предложенному Karafotias et al. Однако разбиение диапазона допусти-

мых значений параметров меняется в ходе работы алгоритма.

Агент выбирает действие на основе алгоритма Q-обучения с ϵ-жадной стратегией исследования среды. В случае когда значения ожидаемой награды примерно одинаковы для всех воможных действий, агент не может выбрать какое из действий наиболее эффективно. Поэтому в данном случае текущее разбиение диапазонов значения параметров пересчитывается. При этом в следствие изменения разбиения меняется множество допустимых действий агента.

В процессе работы алгоритма сохраняются выбранные назначения параметров и полученные за эти назначения награды. Сохраненные данные используются при переразбиении диапазона значений для каждого из настраиваемых параметров. Опишем процедуру переразбиения диапазона. Сначала диапазон делится на два подинтервала при помощи критерия Колмогорова-Смирнова. На каждой следующей итерации разбиения диапазона, полученные на текущей итерации подинтервалы при помощи критерия Колмогорова-Смирнова разбиваются на два подинтервала. В случае, если точка разбиения подинтервала не найдена, разбиение интервала не происходит. Таким образом, максимальное число подинтервалов на которые разбивается диапазон допустимых значений параметра равен 2^i, где i – число итераций разбиения диапазона. На листинге 5 представлен алгоритм разбиения диапазона для $i = 2$.

Листинг 5 Алгоритм разбиения диапазона с двумя итерациями в методе с адаптивным выделением множества действий

Вход: V — множество назначений параметров;

for параметр v **do**
  Разбиение $P \leftarrow \emptyset$
  Отсортировать множество назначений по параметру v
  С помощью критерия Колмогорова-Смирнова найти точку разбиения s множества V
  if Точка разбиения s не найдена **then**
    $P \leftarrow \{[v_{min}, v_{max}]\}$
  else
    Разбить множество V на L и R в соответствии с s
    Найти точку разбиения s_l для множества L
    Найти точку разбиения s_r для множества R
    if Точки разбиения s_l и s_r не найдены **then**
      $P \leftarrow \{[v_{min}, s], (s, v_{max}]\}$
    else if Точка разбиения s_l не найдена **then**
      $P \leftarrow \{[v_{min}, s], (s, s_r], (s_r, v_{max}]\}$
    else if Точка разбиения s_r не найдена **then**
      $P \leftarrow \{[v_{min}, s_l], (s_l, s], (s, v_{max}]\}$
    else
      $P \leftarrow \{[v_{min}, s_l], (s_l, s], (s, s_r], (s_r, v_{max}]\}$
    end if
  end if
end for

Глава 3. Результаты

В данной главе приводятся результаты тестирования разработанных методов адаптивного выбора параметров ЭА на модельных задачах. Приводятся результаты сравнения существующих методов с существующими методами настройки параметров ЭА с помощью обучения с подкреплением. Также исследуется эффективность существующих методов выделения состояний среды.

3.1. Описание экспериментов

В качестве модельных задач используются известные задачи числовой оптимизации, а именно нахождение глобального минимума некоторой многомерной функции в ограниченной области с заданной точностью. Пусть $F(x_1, \ldots, x_n)$ – оптимизируемая функция. При этом $x_i \in [min_i, max_i]$ для $1 \leq i \leq n$.

В качестве особи ЭА, решающего заданную задачу, рассматривался набор из n вещественных чисел. Пусть $x_1, \ldots, x_n$ – текущее решение задачи оптимизации. В качестве эволюционных операторов применялся только оператор мутации, определяемый следующим образом:

$$x_i = \begin{cases} max_i & \text{, при } x_i + \sigma dx_i > max_i \\ min_i & \text{, при } x_i + \sigma dx_i < min_i \\ x_i + \sigma dx_i & \text{, иначе} \end{cases} \text{, для } 1 \leq i \leq n \qquad (3.1)$$

где $dx_i \sim \mathcal{N}(0, 1)$, а σ – настраиваемый параметр, называемый *шагом*. Ожидаемым поведением методов адаптивного выбора параметров ЭА является уменьшение значения шага мутации при приближении к глобальному минимуму оптимизируемой функции. В проводимых экспериментах допустимые значения параметра σ задаются интервалом $[0, k]$, где k – некоторая константа. В рамках данной работы проводились эксперименты с различными значениями параметра k. Увеличение k означает увеличение

области значений параметра σ, что усложняет задачу поиска его оптимального значения.

В качестве ЭА, решающего поставленную задачу, используется $(\mu + \lambda)$ эволюционная стратегия. При увеличении λ увеличивается число использований каждого выбранного значения параметра. Это способствует более точной оценки эффективности выбранного значения, однако сказывается на производительности. Кроме того, тестирование предложенных и существующих методов настройки параметров ЭА нельзя ограничить лишь рассмотрением $(1 + \lambda)$ эволюционной стратегии. В самом деле для если в поколении всего одна особь, то среди предложенных для построения дерева *UTree* характеристик ЭА останется лишь число итераций без улучшения минимального известного значения функции. Таким образом в рамках данной работы рассматривались различные значения параметра μ, а в качестве характеристик ЭА также использовалось среднеквадратичное отклонение значений функции приспособленности для поколения и уменьшения среднего значения функции приспособленности.

Пусть f_t – минимальное значение функции приспособленности на t-ого поколения ЭА. Награда считается по формуле $R = c(\frac{f_t}{f_{t+1}} - 1)$, так как при поиске глобального минимума оптимизируемой функции значения уменьшаются. Так как для решения задачи используется $(\mu + \lambda)$ эволюционная стратегия, то f_t не возрастает. Поэтому агент всегда получает неотрицательную награду.

3.1.1. Значения параметров

В рамках проводимых экспериментов большинство параметров были взяты из статьи karafotias. Были использованы следующие значения параметров, используемых в Q-обучении:

- $\alpha_0 = 0.2$ – коэффициент скорости обучения при нулевой награде;
- $\alpha = 0.9$ – коэффициент скорости обучения при положительной награде;

- $\gamma = 0.8$ – дисконтный фактор;
- $\epsilon = 0.1$ – вероятность выбора случайного действия.

Коэффициент масштабирования награды $c = 100$. Эксперименты проводились для следующих значений парамеров ЭА:

- $\mu \in \{1, 5, 10\}$ – число особей в поколении;
- $\lambda \in \{1, 3, 7\}$ – число потомков, создаваемых для каждой особи в процессе формирования следующего поколения;
- $\sigma \in \{1, 2, 3\}$ – верхняя граница допустимых значений σ.

3.1.2. Описание результатов

Введем следующие обозначения для исследуемых методов настройки параметров ЭА:

- *gecco* – метод, предложенный Karafotias et al. (1.2.4);
- *q-learn* – метод настройки параметров ЭА с помощью Q-обучения с одним состоянием среды и множеством действий, заданным до начала работы алгоритма;
- *earpc* – метод Earpc (1.2.1);
- *uearpc* – метод на основе *earpc* и *UTree* (2.2);
- *dist* – Метод с адаптивным выделением множества действий (2.3);

Для каждого метода проводилось 50 запусков ЭА при всех комбинациях рассматриваемых значений параметров k, μ, λ. Для каждой задачи было составлено три таблицы с усредненным числом поколенй, потребовавшихся для решения задачи, при различных значениях параметров ЭА. В первой из них находятся результаты работы метода dist. Остальные таблицы содержат сравнение пар методов *gecco* и *q-learn*, *earpc* и *uearpc* при разных значениях μ, σ и k. Зеленым цветом текста выделяется лучшее значение среди всех исследуемых методов при заданных значениях параметров ЭА.

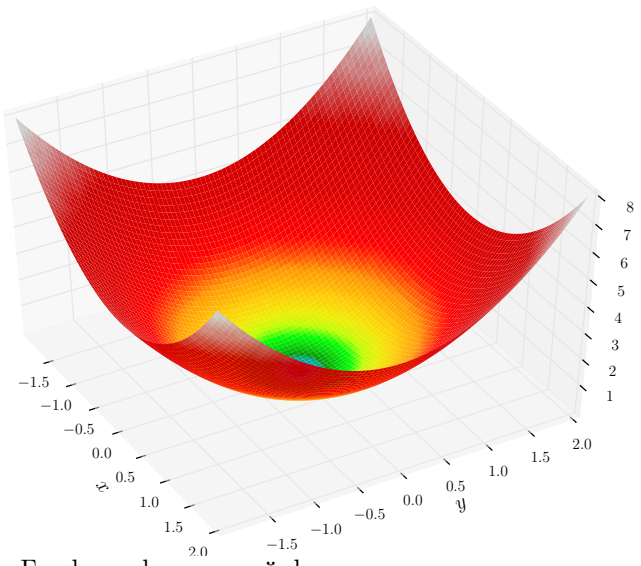

Рис. 3.1: График сферической функции для двух переменных.

Далее подробно описываются модельные задачи, на которых проводилось экспериментальное исследование методов настройки параметров ЭА. Для каждой задачи приводятся результаты ее решения.

3.2. Сфера

Необходимо с точностью ϵ найти минимум унимодальной сферической функции (3.2).

$$f(x_1, .., x_n) = \sum_{i=1}^{n} x_i^2 \tag{3.2}$$

График функции для двух переменных представлен на рис. 3.1. При $x_i \in [-10; 15]$ глобальный минимум достигается в точке $(0, .., 0)$.

3.3. Функция Розенброка

Предложенная Ховардом Розенброком невыпуклая функция (3.3) часто используется для оценки производительности алгоритмов оптимизации. Обычно $a = 1$, $b = 100$. Считается, что поиск глобального минимума для данной функции является нетривиальной задачей. График функции для двух переменных представлен на рис. 3.6. Глобальный минимум находится в длинной, узкой *впадине*, найти которую обычно не составляет

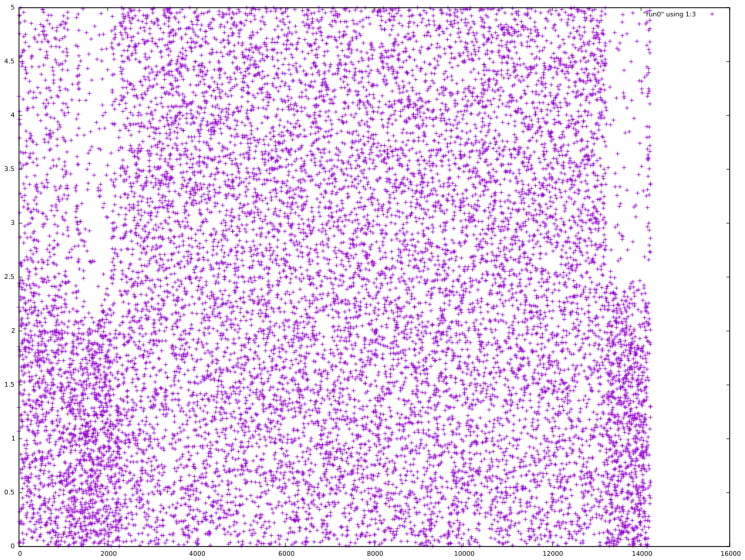

Рис. 3.2: График выбранных значений σ с помощью метода *earpc*

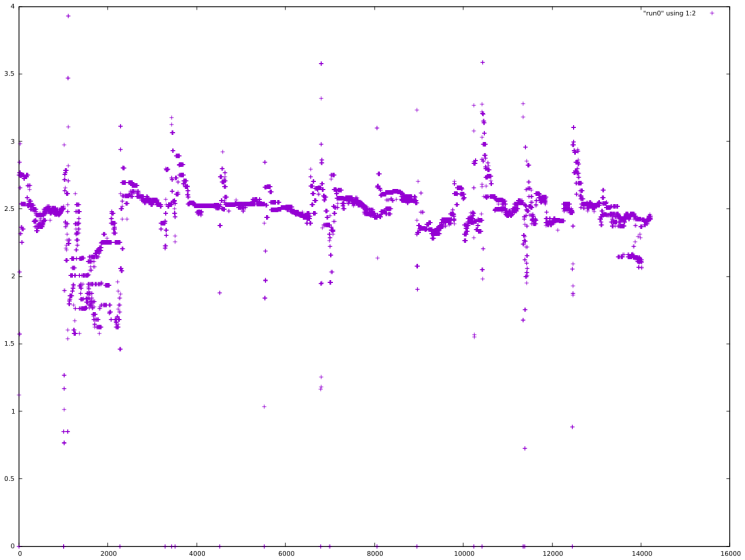

Рис. 3.3: График точки разбиения в методе *earpc*

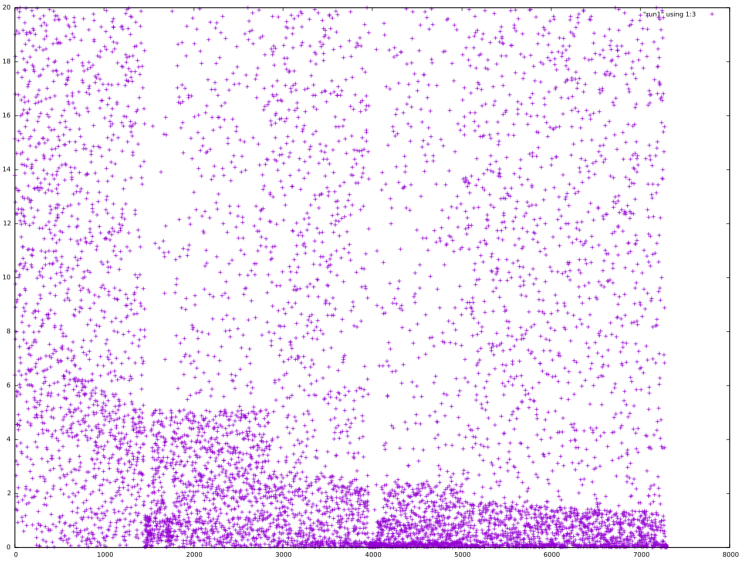

Рис. 3.4: График выбранных значений σ с помощью предложенного метода

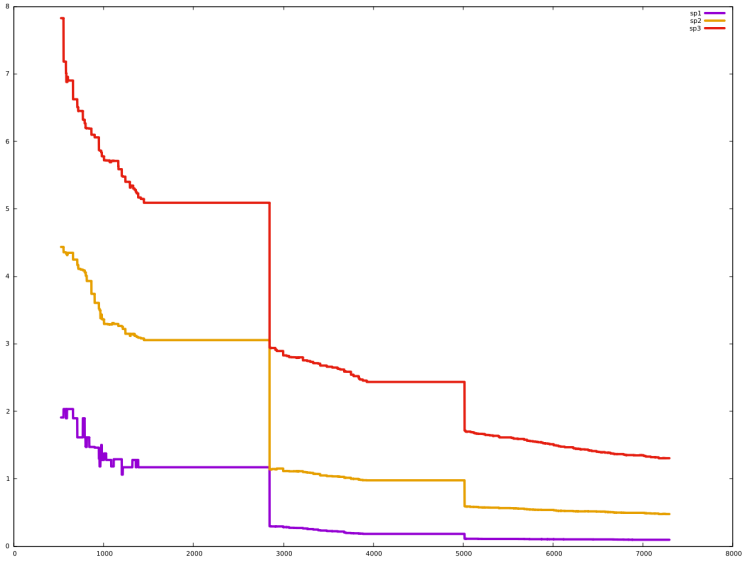

Рис. 3.5: График точек разбиения в предложенном методе

k = 1			
μ \ λ	1	3	7
1	4830	2462	878
5	1450	695	368
10	808	551	218
k = 2			
μ \ λ	1	3	7
1	4342	2333	1464
5	1891	974	695
10	1164	672	509
k = 3			
μ \ λ	1	3	7
1	8199	2826	1427
5	2447	1445	910
10	1996	1152	653

Таблица 3.1: Таблица с результатами применения метода dist

k = 1						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	8769	8048	4221	2683	1085	2620
5	1664	2076	406	824	390	473
10	959	703	378	358	195	167
k = 2						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	25192	28523	7681	6478	3360	3739
5	3814	3468	994	1163	716	756
10	2397	1833	445	825	320	252
k = 3						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	36698	29112	7845	14115	2907	5813
5	8328	5886	2790	2222	919	777
10	2398	2531	1074	1206	291	427

Таблица 3.2: Таблица с результатами применения методов q-learn и gecco

k = 1						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	5258	2434	3942	3070	1653	2247
5	4472	3893	1589	845	642	568
10	1438	728	534	400	342	422
k = 2						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	31738	16182	5152	4233	1688	2956
5	4908	5173	1631	946	1511	626
10	1778	5176	719	788	380	413
k = 3						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	54868	30710	12446	11985	10118	6207
5	3348	12313	979	1848	1424	1105
10	3173	3745	1114	1712	410	537

Таблица 3.3: Таблица с результатами применения методов q-learn и gecco

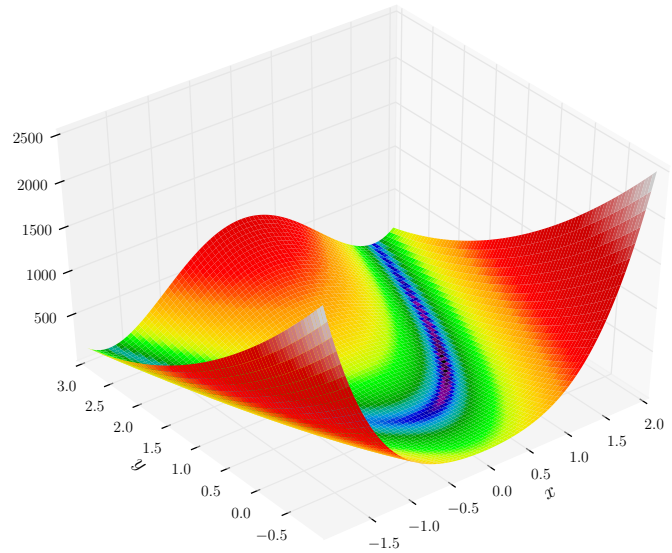

Рис. 3.6: График функции Розенброка для двух переменных.

k = 1			
μ \ λ	1	3	7
1	3631	2035	1226
5	1666	1078	502
10	1450	622	358
k = 2			
μ \ λ	1	3	7
1	4744	1839	1160
5	2467	1128	933
10	1722	988	615
k = 3			
μ \ λ	1	3	7
1	5159	2821	1517
5	2704	1544	902
10	2048	1296	1013

труда. Трудность заключается в поиске минимума в этой впадине. При $x_i \in [-15; 10]$ он достигается в точке $(1, .., 1)$.

$$f(x_1, x_2) = (a - x_1^2)^2 + b(x_2 - x_1^2)^2 \tag{3.3}$$

3.4. Функция Леви

Мультимодальная функция Леви для двух переменных задается формулой (3.4). График функции представлен на рис. 3.7. При $x, y \in [-10; 10]$ минимум достигается в точке $(1, 1)$.

k = 1						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	9653	8385	2226	1769	1105	1757
5	1706	2281	894	942	570	675
10	1103	1105	504	665	489	473
k = 2						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	23663	21043	6405	6825	2388	2183
5	3944	4029	1614	1780	1012	1461
10	2108	2255	1420	1116	856	712
k = 3						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	29082	23305	7977	10726	4680	4266
5	6208	5419	2514	2096	1120	1629
10	3747	3258	1740	1523	995	1203

k = 1						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	7794	8689	3148	3610	1422	1422
5	4341	4530	1958	2197	1679	1329
10	2681	2926	1460	1485	1103	1858
k = 2						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	27865	19142	9748	8997	3806	4085
5	5961	5750	2293	1720	633	1180
10	2411	2807	1486	1234	627	922
k = 3						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	24249	27327	6541	16040	4144	5408
5	5953	7665	2823	1304	1033	1055
10	5095	3873	913	1028	679	839

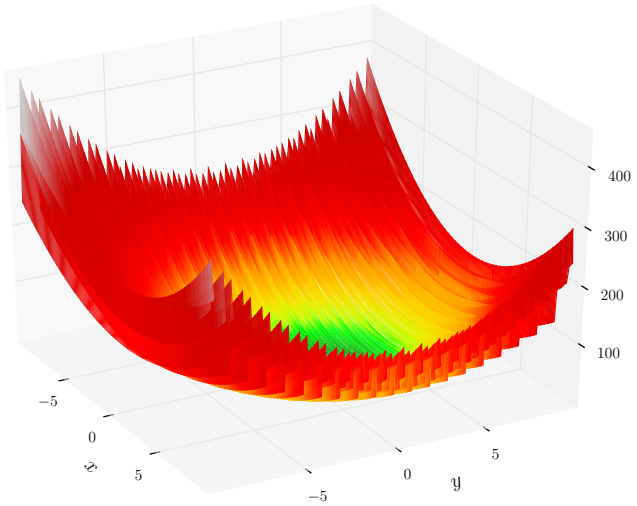

Рис. 3.7: График функции Леви для двух переменных.

k = 1			
μ \ λ	1	3	7
1	3496	1980	1321
5	1778	855	504
10	1305	717	456
k = 2			
μ \ λ	1	3	7
1	4947	2020	1205
5	1935	1085	953
10	1803	918	801
k = 3			
μ \ λ	1	3	7
1	5216	2900	1700
5	3105	1808	1017
10	2071	1330	904

$$f(x_1, x2) = sin^2(3\pi x) + (x-1)^2(1 + sin^2(3\pi y)) + \\ + (y-1)^2(1 + sin^2(2\pi y)) \tag{3.4}$$

3.5. Функция Растригина

Леонард Растригин предложил нелинейную мультимодальную функцию для тестирования эффективности алгоритмов оптимизации. Нахождение минимума этой функции затруднено большим количеством локальных минимумов в области поиска. Функция задается формулой (3.5).

k = 1						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	7200	7265	3305	2903	1284	1600
5	1898	1865	882	724	606	444
10	840	764	502	624	331	294
k = 2						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	13420	14789	6884	6601	3521	2370
5	3517	3369	1031	1326	792	1089
10	1884	2197	988	1006	731	564
k = 3						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	21470	21953	9029	7071	4139	4019
5	6966	6742	1467	2664	1929	1788
10	2901	2943	1462	1832	1117	799

k = 1						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	7986	14092	2789	3688	1923	3820
5	2372	2246	1632	2171	1426	1236
10	2353	1451	1491	1134	510	766
k = 2						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	25721	30653	7477	4612	2533	2967
5	7222	5365	3039	2943	1153	2180
10	5139	3072	1045	2381	806	1064
k = 3						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	28900	35119	6966	10883	3375	2062
5	10480	9059	4593	3714	594	1439
10	5210	4814	3140	2389	521	845

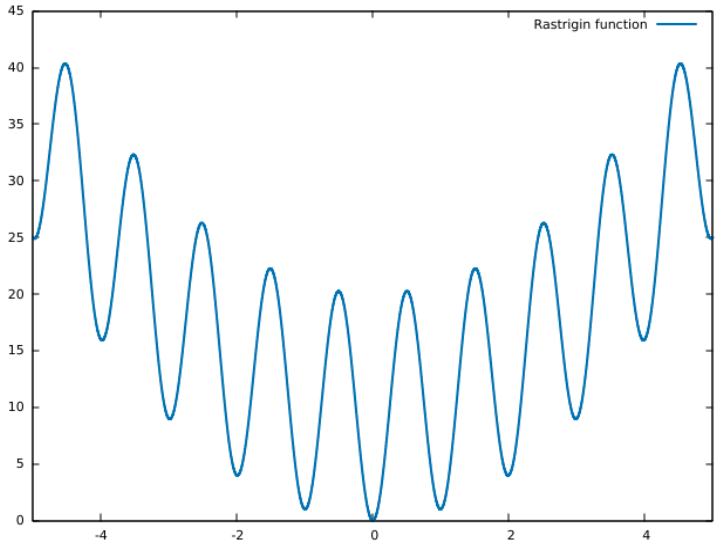

Рис. 3.8: График функции Растригина для одной переменной.

k = 1			
μ \ λ	1	3	7
1	734	273	147
5	166	64	40
10	93	56	22
k = 2			
μ \ λ	1	3	7
1	767	409	167
5	287	180	55
10	167	65	45
k = 3			
μ \ λ	1	3	7
1	889	579	282
5	655	165	115
10	247	98	38

$$f(x_1, .., x_n) = An + \sum_{i=1}^{n} \left[x_i^2 - Acos\left(2\pi x_i\right)\right] \tag{3.5}$$

3.5.1. Одномерный случай

3.5.2. Двумерный случай

3.6. Выводы

k = 1						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	994	767	331	329	133	224
5	160	145	66	106	62	49
10	111	109	53	34	31	31
k = 2						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	1123	1018	318	449	192	221
5	254	202	96	111	75	75
10	101	83	43	106	47	41
k = 3						
λ / μ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	2052	1258	400	957	245	255
5	213	234	131	106	98	94
10	114	190	109	82	65	28

k = 1						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	382	694	378	367	146	171
5	718	370	73	47	35	93
10	104	52	36	32	22	18
k = 2						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	1135	2793	243	256	393	250
5	379	337	110	259	35	77
10	85	313	134	80	23	20
k = 3						
λ / μ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	1712	2735	653	289	346	537
5	827	313	138	158	88	59
10	478	169	36	59	30	68

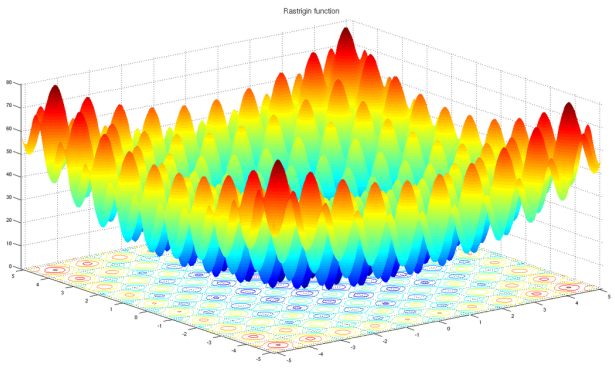

Рис. 3.9: График функции Растригина для двух переменных.

k = 1			
μ \ λ	1	3	7
1	5124	2301	1411
5	1859	1053	988
10	1617	1157	865
k = 2			
μ \ λ	1	3	7
1	5165	3461	1753
5	1990	1396	1095
10	2022	1354	998
k = 3			
μ \ λ	1	3	7
1	6535	3391	2573
5	3148	1721	1231
10	2352	1677	1119

k = 1						
μ \ λ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	15058	13418	3914	4167	1791	2330
5	2311	2809	869	748	533	665
10	1497	1593	488	616	290	235
k = 2						
μ \ λ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	23371	27239	7295	7169	3488	2968
5	8076	5810	2116	2705	1001	1247
10	3415	2787	1037	1106	811	666
k = 3						
μ \ λ	1		3		7	
	q-learn	gecco	q-learn	gecco	q-learn	gecco
1	28702	36597	11427	9873	5820	6462
5	12438	6791	2289	3673	1626	1255
10	4473	6531	2000	1650	1137	1098

k = 1						
μ \ λ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	9003	12905	3553	1701	2296	1941
5	5730	4453	1393	2749	1130	1258
10	2213	3085	1161	1225	391	429
k = 2						
μ \ λ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	20005	20819	7166	13342	5874	7870
5	11153	11856	3775	4542	1775	2584
10	8768	4603	3719	1648	1740	2558
k = 3						
μ \ λ	1		3		7	
	earpc	uearpc	earpc	uearpc	earpc	uearpc
1	32533	43832	13061	10068	9797	9775
5	8320	10539	4434	6799	2951	4914
10	8952	11581	2611	1307	1742	2479

Заключение