Конспект лекций по курсу «Эволюционные алгоритмы»

Еремеев А.В.

Abstract. This manuscript contains an outline of lectures on "Evolutionary Algorithms" course read by the author in Omsk State University n.a. F.M.Dostoevsky. The course covers Canonic Genetic Algorithm and various other genetic algorithms as well as evolutioanry algorithms in general. Some facts, such as the Rotation Property of crossover, the Schemata Theorem, GA performance as a local search and "almost surely" convergence of evolutionary algorithms are given with complete proofs. The text is in Russian.

Введение

Эволюционные алгоритмы (ЭА), к которым можно отнести эволюционные стратегии, генетические алгоритмы, эволюционное программирование, берут свое начало в работах А.Г. Ивахненко [15], Л.А. Растригина [22], Дж.Холланда [72], И.Реченберга [87], Л. Фогеля, А. Оуэнса, М. Уолша [26], и других авторов, вышедших в 60-70-х годах двадцатого века. Основная идея ЭА состоит в компьютерном моделировании процесса эволюции. При этом моделирование предназначено не для исследования биологических популяций, а для решения практических задач прикладной математики, в частности, задач оптимизации.

Области применения ЭА: Экономика, управление, инженерные задачи, переработка информации, космос, медицина и т.д. Область эволюционных вычислений возникла на стыке биологии и прикладной математики (искусственный интеллект, методы оптимизации, теория вероятностей). В настоящее время эволюционные алгоритмы зачастую относятся к методам искусственного интеллекта. С одной стороны, эти алгоритмы позволяют решать задачи машинного обучения, такие как задачи символьной регрессии, оптимизация структуры нейронной сети, выбор модели машинного обучения и др. С другой стороны, в процессе работы этих алгоритмов используются эвристики, традиционно применяемые человеком для принятия решений, такие как метод проб и ошибок (эволюционная стратегия (1+1) ES), жадная эвристика (декодирование решений), комбинация нескольких известных идей в одном изобретении (оператор рекомбинации), адаптация к среде (локальная оптимизация). Наконец, во многих системах искусственного интеллекта требуется решать задачи оптимизации большой размерности, которые не уда-

ется решить классическими методами математического программирования, однако эволюционные алгоритмы позволяют находить для них приемлемые решения.

Один из типичных представителей эволюционных алгоритмов в оптимизации – генетический алгоритм (ГА). При запуске ГА создается «виртуальная» популяция особей (как правило достаточно хранить только их генотипы), каждая из которых представляет элемент пространства решений оптимизационной задачи. Здесь и далее используются некоторые термины, заимствованные из биологии [23].

Приспособленность особей к условиям окружающей среды выражается некоторой монотонной функцией от значения целевой функции задачи. Чем лучше решение - тем выше приспособленность особей с соответствующим генотипом. Популяция развивается за счет отбора более пригодных особей и применения к ним случайных операторов, имитирующих мутацию генов и рекомбинацию родительских генотипов (кроссинговер).

Отбор может осуществляться по-разному. Особенно распространенными являются операторы пропорциональной селекции (вероятность выбора пропорциональна пригодности), срезающей селекции (задается равномерным распределением на множестве из $T\%$ лучших генотипов популяции) и турнирной селекции (s особей извлекаются с помощью равномерного распределения, затем берется лучшая из них).

ГА представляет собой универсальную схему решения широкого круга задач задач. Достаточно определить представление решений в виде генотипа, выбрать операторы селекции и кроссинговера, и алгоритм можно применять. Более того, при достаточно общих предположениях можно доказать, что с вероятностью единица алгоритм находит оптимальное решение, если время его работы не ограничено.

В области эволюционных алгоритмов находит себе применение бионический подход, состоящий в заимствовании принципов организации систем из живой природы. В данном случае имеет место использование принципа постепенных адаптивных преобразований в пределах популяции или вида в ходе, так называемой, микроэволюции. Как показывают исследования акад. Ю.П. Алтухова и других авторов (см., [1], § 6.4), более «масштабная» межвидовая изменчивость (макроэволюция) требует скачкообразных перестроек генотипа и не может быть выведена непосредственно из постепенной внутривидовой изменчивости. Чрезвычайно малые вероятности реализации элементарных сценариев эволюционного развития [1] не позволяют использо-

[1]Для примера оценим вероятностть возникновения нового белка с требуемой функцией в результате случайных мутаций. Пусть для адаптации некоторой бактерии к условиям окружающей среды требуется полявление нового белка, выполняющего некоторую конкретную функцию. В небольших молекулах белка содержится около 150 аминокислот. Будем считать, что искомый белок именно такой длины. Одна аминокислота кодируется триплетом нуклеотидов, т.е. для ее кодировки необходимо 450 нуклеотидных позиций в молекуле ДНК. Было бы серьезным упрощением считать, что заданным требованиям удовлетворяет

вать аналогии с макроэволюцией и процессами видообразования для обоснования работоспособности эволюционных алгоритмов. Вместо этого в теории эволюционных алгоритмов проводятся непосредственные исследования ЭА с использованием теории сложности и теории вероятностей (см. [94], § 1.3). В частности, такие результаты рассматриваются далее в п. 2.5 и гл. 4.

Генетические алгоритмы (ГА) наряду с алгоритмами имитации отжига, поиска с запретами и муравьиной колонии и др. относятся к классу *метаэвристик*. Вообще, эвристикой (от греч. еуриско - обнаруживаю) называют метод решения, основанный на неформальных, интуитивных соображениях, не гарантирующий получения наилучшего решения. Метаэвристика является эвристикой с универсальной схемой, применимой для поиска приближенных решений различных оптимизационных задач и представляющая собой итерационный процесс, в котором многократно используются подчиненные эвристики, учитывающие особенности задачи. В метаэвристике могут использоваться различные принципы исследования пространства решений и стратегии адаптации для учета полученной информации. На каждой итерации метаэвристики могут выполняться операции с единственным текущим решением (или частичным решением), либо с набором (популяцией) решений. Подчиненные эвристики могут быть процедурами высокого или низкого уровня, простым локальным поиском, градиентным методом построения решения или классическими оптимизационными процедурами.

только *одна* последовательность аминокислот (как, например, в [48]). Однако можно считать что примерно половина из нуклеотидных позиций (в нашем случае 225 шт.) имеют не больше двух «подходящих» нуклеотидов (ср., например, [43]), а значит вероятность случайного получения требуемого нуклеотида в такой позиции – не больше 0.5. Тогда вероятность получения подходящего белка целиком при однократном испытании (случайном выстраивании 450 нуклеотидов) оценивается сверху величиной $2^{-225} \approx 10^{-67}$.

Найдем грубую оценку для числа попыток построить подходящий белок за историю существования нашей планеты. Число бактерий на земле оценивается величиной $2 \cdot 10^{30}$, см. [61]. Число нуклеотидов в геноме одной бактерии - до 12.2 млн. [86], т.е. единовременно могжет быть протестировано не более $\frac{12.2 \cdot 10^6}{450}$ подпоследовательностей по 450 нуклеотидов в каждй. За один год популяция одной из наиболее быстро размножающихся бактерий *e.coli* проходит около 2000 поколений. Возраст земли оценивается как $4 \cdot 10^9$ лет. Тогда число испытаний в истории развития всех бактерий не превышает величины порядка $2 \cdot 10^{30} \cdot \frac{12.2 \cdot 10^6}{450} \cdot 8 \cdot 10^{12} \approx 10^{47}$.

Отсюда получаем оценку сверху для вероятности хотя бы однократного получения подходящего белка за всю историю существования земли: 10^{-20}. И это только для одного эпизода эволюционного развития, где требуется возникновение одного нового белка.

1. Классический генетический алгоритм

Рассмотрим задачу безусловной оптимизации в общем виде:

$$\max\{f(x) : x \in X\}, \tag{1.1}$$

где X- пространство решений (мощности континуума, счетное или конечное).

Генетический алгоритм (ГА) представляет собой эвристический алгоритм оптимизации, в основу которого положены биологические принципы естественного отбора и изменчивости. Процесс работы алгоритма представляет собой последовательную смену поколений, состоящих из фиксированного числа особей-точек пространства решений, причем особи с бо'льшим значением целевой функции (более приспособленные) получают больше потомков в каждом следующем поколении. Кроме того, при формировании следующего поколения часть потомков полностью идентична родителям, а часть изменяется некоторым случайным образом в результате мутации и кроссинговера (скрещивания).

При использовании генетического алгоритма для поиска в дискретном пространстве X, каждой строке из l символов некоторого алфавита A должен быть сопоставлен элемент пространства X , т.е. определена функция $x : B \to X$, (называемая также *схемой представления*), где $B = A^l$. Строки $\xi \in B$ принято называть генотипами, а их образы $x(\xi) \in X$ – фенотипами.

В классическом генетическом алгоритме (КГА) используется двоичный алфавит $A = \{0, 1\}$. Вообще, по умолчанию будем предполагать, что алфавит A конечный.

Популяцией $\Pi = (\xi^1, \xi^2, ..., \xi^\lambda)$ численности λ является вектор пространства B^λ, координаты которого называются генотипами особей (индивидов) данной популяции. Как правило, нумерация особей популяции не имет значения. Численность популяции λ фиксирована от начала работы алгоритма до конца. Предполагается, что λ – четное.

Целевая функция $f : X \to R$ исходной задачи заменяется в ГА на функцию приспособленности генотипа[1] $\Phi(\xi) = \phi(f(x(\xi)))$, где $\xi \in B$. Здесь $\phi : R \to R_{\geq 0}$ – некоторая монотонно возрастающая функция. В биологической интерпретации функция приспособленности отражает степень приспособленности индивида с генотипом ξ к условиям «окружающей среды», заданным функцией $\Phi(\xi)$. При этом максимумы целевой функции соответствуют наиболее приспособленным генотипам для данной «окружающей среды». Простейшим примером функции приспособленности является сама целевая функция при условии, что она неотрицательна.

[1] В англоязычной литературе используется термин *fitness function.*

Один из лучших найденных генотипов к поколению t будем обозначать через $\tilde{\xi}_t$:

$$\tilde{\xi}_t \in \arg\max\{\Phi(\xi^{i,\tau}), i = 1, ..., \lambda, \tau = 0, ..., t\}.$$

Приведем общую схему *генетического алгоритма с полной заменой популяции*. Этой схеме соответствует, в частности, КГА. Используемые здесь вероятностные операторы $\mathrm{Sel} : B^\lambda \to \{1, \ldots, \lambda\}$, $\mathrm{Cross} : B \times B \to B \times B$ и $\mathrm{Mut} : B \to B$ будут описаны ниже.

Генетический алгоритм с полной заменой популяции

0. Положить $t := 0$.
1. Для k от 1 до λ выполнять:
1.1. Построить случайным образом генотип $\xi^{k,0}$.
2. Для k от 1 до $\lambda/2$ выполнять шаги 2.1-2.3:
2.1. Селекция: выбрать генотипы $\xi := \xi^{Sel(\Pi^t),t}$, $\eta := \xi^{Sel(\Pi^t),t}$.
2.2. Скрещивание: построить $(\xi', \eta') := \mathrm{Cross}(\xi, \eta)$.
2.3. Мутация: положить $\xi^{2k-1,t+1} := \mathrm{Mut}(\xi')$, $\xi^{2k,t+1} := \mathrm{Mut}(\eta')$.
3. Положить $t := t + 1$.
4. Если выполнен критерий остановки, то идти на шаг 5, иначе – на шаг 2.
5. Результатом работы алгоритма является лучшее из найденных решений $x(\tilde{\xi}_{t-1})$.

Поясним приведенную схему. На этапе инициализации (шаги 0 и 1) формируется начальная популяция Π^0, элементы которой генерируются в соответствии с равномерным распределением на множестве генотипов B, т.е. $P\{\xi_k^{i,0} = 0\} = P\{\xi_k^{i,0} = 1\} = 1/2,\ i = 1, \ldots, \lambda,\ k = 1, \ldots, l$.

Вероятностный оператор селекции особей на пространстве популяций $\mathrm{Sel}(\Pi)$ имеет то же значение, что и естественный отбор в природе. Действие этого оператора состоит в выборе номера родительской особи для построения очередного потомка.

В КГА генотип $\xi^{i,t}$ с номером i, $i = 1, \ldots, \lambda$ из популяции Π^t оказывается родительской особью при формировании очередного генотипа $\xi^{k,t+1}$ популяции Π^{t+1} с вероятностью

$$P_{\mathbf{s}}(i, \Pi^t) = \frac{\Phi(\xi^{i,t})}{\sum_{j=1}^{\lambda} \Phi(\xi^{j,t})}. \tag{1.2}$$

Если окажется, что $\sum_{j=1}^{\lambda} \Phi(\xi^{j,t}) = 0$, то есть все генотипы имеют нулевую приспособленность, условимся выбирать номер особи с равномерным распределением из 1,...,λ.

В КГА не исключается выбор $\xi^{i,t}$ одновременно в качестве ξ и η на шаге 2.1. Описанный оператор Sel иногда также называют селекцией методом

рулетки [24, 63]. Предположим, что колесо рулетки разбито на λ секторов, причем сектор i соответствует особи i и имеет радианную меру $2\pi P_{\mathtt{s}}(i, \Pi^t)$. Тогда селекцию особи $\xi^{i,t}$ можно представлять, как выбор i-го сектора на колесе рулетки.

Описанный оператор селекции также называется *пропорциональным* в связи с тем, что при фиксированном составе популяции вероятность выбора особи в качестве родителя пропорциональна ее приспособленности.

Упражнение 1.0.1. *Показать, что при известных значениях приспособленности особей текущей популяции* Π^t *селекция всех родительских особей для построения новой популяции* Π^{t+1} *может быть выполнена в КГА за время* $O(\lambda \log_2(\lambda))$.

1.1. Процедуры кроссинговера и мутации классического генетического алгоритма

Опишем двуместный оператор кроссинговера (скрещивания) $\mathrm{Cross}(\xi, \eta)$ и одноместный оператор мутации $\mathrm{Mut}(\xi)$, действие которых носит случайный характер.

Результат кроссинговера $(\xi', \eta') = \mathrm{Cross}(\xi, \eta)$ с вероятностью $P_{\mathtt{c}}$ формируется в виде

$$\xi' = (\xi_1, \xi_2, ..., \xi_\chi, \eta_{\chi+1}, ..., \eta_l),$$
$$\eta' = (\eta_1, \eta_2, ..., \eta_\chi, \xi_{\chi+1}, ..., \xi_l),$$

где случайная координата скрещивания χ выбрана с равномерным распределением от 1 до $l-1$. С вероятностью $1 - P_{\mathtt{c}}$ оба генотипа сохраняются без изменений, т.е. $\xi' = \xi,\ \eta' = \eta$. Влияние оператора кроссинговера регулируется параметром $P_{\mathtt{c}}$. Данный оператор принято называть *одноточечным кроссинговером.*[2]

Оператор мутации в каждой позиции генотипа с заданной вероятностью $P_{\mathtt{m}}$ изменяет ее содержимое. В противном случае ген остается без изменений. Таким образом, мутация элементов генотипа происходит по схеме Бернулли с вероятностью успеха $P_{\mathtt{m}}$.

Изменение вероятностей мутации и кроссинговера позволяет регулировать работу КГА и настраивать его на конкретные задачи. Увеличение вероятности мутации до 0.5 превращает КГА в простой случайный перебор, имеющий весьма ограниченное применение (см. [22], § 6.1). Уменьшение же $P_{\mathtt{m}}$ до нуля приводит к малому разнообразию генотипов в популяции и может вызвать «зацикливание» КГА, когда на каждой итерации генерируются лишь ранее встречавшиеся генотипы. Величины $P_{\mathtt{c}}$ и λ также могут существенно влиять на скорость сходимости популяции к решениям приемлемого качества

[2] В англоязычной литературе используется термин *one-point crossover.*

(см., например, [9,24]). Настраиваемые параметры КГА выбирают, как правило, в следующих диапазонах: $0 \le P_{\mathtt{c}} \le 1,\ 10^{-3} \le P_{\mathtt{m}} \le 0.3,\ 30 \le \lambda \le 10000$.

В отличие от большинства представителей животного мира, особи генетических алгоритмов имеют не двойной хромосомный набор (диплоидный), а одинарный (гаплоидный), т.к. хранение дублирующих друг друга генотипов, полученных потомком от обеих родительских особей при решении оптимизационных задач не целесообразно. Особи ГА сходны с такими организмами, как мхи-гаметофиты или некоторые виды водорослей, которые имеют одинарный набор хромосом в течение длительного этапа жизни.

1.2. Примеры использования классического генетического алгоритма

Рассмотрим задачу оптимизации с ограничениями:

$$\max\{F(x) : x \in D \subseteq X\}, \tag{1.3}$$

где D – область допустимых решений.

1.2.1. Максимизация функции $F : \{a, a+1, ..., b\} \to \mathbf{N}$.

Здесь $X = \mathbf{Z}$, т.е. множество целых чисел, $D = \{a, a+1, ..., b\}$. Воспользуемся бинарной кодировкой решений: $l = \lceil \log_2(b-a+1) \rceil$,

$$x(\xi) = a + \sum_{j=0}^{l-1} \xi_{l-j} 2^j. \tag{1.4}$$

При $x(\xi) \in \{a, a+1, ..., b\}$ полагаем $\Phi(\xi) = f(x(\xi))$, иначе: $\Phi(\xi) = 0$.

1.2.2. Задача о разрезе максимального веса.

Дан граф $G = (V, E), V = \{v_1, ..., v_n\}$, *каждому ребру приписан вес* $w : E \to \mathbb{R}^+$, *где* $\mathbb{R}^+$ – *множество полложительных вещественных чисел. Найти разрез* $\{U, U'\} : U \subseteq V, U' = V \backslash U$, *максимального веса*

$$W(\{U, U'\}) = \sum_{e=(u,v) \in E : u \in U, v \in U'} w(e).$$

Задача NP-трудна [8].

Здесь $D = X = \{\{U, U'\} : U \subseteq V, U' = V \backslash U\}, x = \{U, U'\}, f(x) = W(x)$. Кодировка определяется так: $U(\xi) = \{v_j \in V : \xi_j = 1, j = 1, ..., n\}$, $l = n$, $x(\xi) = \{U(\xi), V \backslash U(\xi)\}, \Phi(\xi) = f(x(\xi))$.

Очевидно, операторы Mut и Cross сохраняют допустимость решений, поэтому они могут быть использованы непосредственно без каких-либо усовершенствований.

Есть, однако, проблема, связанная с вырожденностью кодировки (иногда называют «конкуренцией конвенций»), т.к. один и тот же разрез $\{U, U'\}$ может быть представлен двумя способами (либо 1 кодирует вершину, лежащую в U, либо в U'). Такая неоднозначность может привести к снижению эффективности работы ГА и бессмысленности скрещивания особей, закодированных в разных «конвенциях».

Упражнение 1.2.1. *Предложить взаимно-однозначное представление решений задачи о максимальном разрезе в графе при* $l = n - 1$.

1.2.3. Применение КГА в непрерывной оптимизации и геометрический смысл кроссинговера

Рассмотрим случай, когда D – евклидово пространство размерности n. Можно было бы в данном случае предположить, что алфавит $A = \mathbb{R}$ – множество вещественных чисел, например, как в [79] и строить ГА в таком предположении. Однако мы будем работать в рамках сделанного выше предположения о том, что $A = \{0, 1\}$.

Если $D \subset X = \mathbb{R}^n$ ограничено, его можно дискретизовать, например, путем введения достаточно мелкой регулярной сетки. При этом задача сводится к поиску оптимума на дискретной решетке в некотором n-мерном параллелепипеде $\Omega \subset X$. Пусть область D погружена в n-мерный параллелепипед Ω:

$$D \subseteq \Omega = \{x \in \mathbb{R}^n : a_1 \le x_1 \le b_1, ..., a_n \le x_n \le b_n\}.$$

Обозначим $d_j = b_j - a_j,\ j = 1, ..., n$ Одним из наиболее «естественных» способов кодировки представляется стандартная двоичная кодировка координат векторов в строке генотипа.

Пример: (001 010 011 100) $\xrightarrow{x(\xi)}$ (1,2,3,4).

Далее предполагается, что на кодирование каждой координаты x_j используется k_j бит, и подстрока $g^j = (\xi_{i_{j-1}+1}, ..., \xi_{i_j})$ кодирует j-ю координату, где $i_j = k_1 + \ldots + k_j$ – номер последней битовой позиции, отвечающей за кодировку компоненты x_j. В частности, $l = i_n$. В общем виде:

$$x(\xi)_j = a_j + \frac{d_j}{2^{k_j} - 1} \sum_{i=0}^{k_j - 1} \xi_{i_j - i} 2^i, \quad j = 1, ..., n. \tag{1.5}$$

Утверждение 1.2.1. *(О геометрическом смысле кроссинговера (Р.Т. Файзуллин [60]) Если найдется такое* r, *что* $\chi = i_r,\ 1 \le r < n$, *то результат*

скрещивания $(\xi', \eta') = Cross(\xi, \eta)$ *может быть получен некоторым вращением* $R_{\chi,\xi,\eta}$ *родительских фенотипов* $x(\xi), x(\eta) \in \mathbb{R}^n$, *оставляющим неподвижной точку* $x^0 = \frac{x(\xi)+x(\eta)}{2}$. *Т.е.* $x(\xi') = R_{\chi,\xi,\eta}(x(\xi)), \quad x(\eta') = R_{\chi,\xi,\eta}(x(\eta))$. *При этом середина отрезка, соединяющего родительские фенотипы,* $x^0 = \frac{x(\xi)+x(\eta)}{2}$ *остается неподвижной, т.е.* $R_{\chi,\xi,\eta}(x^0) = x^0$.

1.2.4. Учет ограничений задачи

Существует несколько подходов к обработке точек из $X \backslash D$.

a) Использование штрафной функции. Существует следующий простейший способ, который всегда применим: $f(x) = F(x)$ при $x \in D$, иначе $f(x) = -M$, где M – достаточно большая константа. Недостаток: все точки вне допустимой области одинаково плохи и ГА не имеет информации о близости недопустимой точки к D. Во многих задачах оптимизации поиск допустимого решения сам по себе представляет достаточно сложную задачу (например, задача ЛП сводится к поиску допустимой точки некоторой системы линейных неравенств) и без такой дополнительной информации ГА может не обнаружить ни одного допустимого решения.

Другие более эффективные способы использования штрафов (см., например, [16, 96]) состоят в «градации недопустимости» решений. Например, если область D задана системой неравенств $D = \{x \in X = \mathbb{R}^n : f_1(x) \leq 0, ..., f_m(x) \leq 0\}$, в качестве штрафной функции может быть использована $P(x) = r \sum_{i=1}^{m} \max\{f_i(x), 0\}$, где r достаточно велико. Далее полагают $f(x) = F(x) - P(x)$. Желательно, чтобы выполнялось условие $F(x(\xi)) - P(x(\xi)) < F(x^*) - P(x^*) = F(x^*)$ для любого ξ такого, что $x(\xi) \notin D$.

Можно воспользоваться и классическим способом сведения задач математического программирования к задачам безусловной оптимизации. Для этого рассматривается возрастающая последовательность (теоретически возрастающая до бесконечности) $r_1, r_2, ...$ Для каждого $r_\theta, \theta = 1, 2, ..., \Theta$ решается задача (1) где $f(x) = F(x) - P(x)$ и $P(x)$ найдена при $r = r_\theta$. Величина Θ задает число «больших» итераций алгоритма и выбирается исходя из значимости выполенения ограничений. Естественно при этом каждый новый запуск ГА осуществлять не со случайной начальной популяции, а с последней популяции предыдущего запуска (тогда каждое обновление r_θ можно понимать как изменение «окружающей среды» с точки зрения эволюции популяции).

b) Корректировка недопустимых решений может выполняться с помощью методов непрерывной оптимизации или каких-либо эвристик, которые, стартуя с недопустимого решения $x(\xi) \notin D$, находят некоторое «подходящее» допустимое решение $x' \in D$. Например, если D – множество точек шара, то недопустимые решения могут проецироваться на границу шара и после это-

го кодироваться как особи новой популяции. Аналогично устроена процедура масштабирования решений в [13]. В случае дискретных задач подобный подход использован в ГА для задачи о покрытии множества, где жадная эвристика В. Хватала применяется для достройки недопустимого решения до покрытия [35].

c) Выбор подходящей кодировки, при которой $x(B) \subseteq D$. Примером служит рассмотренная выше задача о разрезе максимального веса.

1.2.5. Задача целочисленного линейного программирования

Рассматривается задача целочисленного линейного программирования (ЦЛП) следующего вида: найти

$$F(x) = (c, x) \to \max \tag{1.6}$$

при условиях

$$Ax \leq b, x \geq 0, \tag{1.7}$$

$$x \in \mathbf{Z}^n. \tag{1.8}$$

Здесь A – $(m \times n)$ - матрица, $c = (c_1, ..., c_n)$, $b = (b_1, ..., b_m)^T$, $x = (x_1, ..., x_n)$. Далее предполагается, что множество $\mathcal{M}$, определяемое системой неравенств (1.7), ограничено. Будем называть вектор целочисленным, если все его компоненты целочисленны.

Общая схема ГА может быть легко адаптирована для задач целочисленного линейного и нелинейного программирования. Ограничимся рассмотрением ГА для задачи ЦЛП.

Многогранник допустимых решений погружается в n-мерный параллелепипед $\Omega = \{x \in \mathbb{R}^n | 0 \leq x_j \leq d_j, j = 1, ..., n\}$ с минимальным объемом. Границы параллелепипеда d_j могут быть найдены решением n соответствующих задач ЛП.

Минимальная длина битовой строки для кодировки координаты j целочисленной точки из Ω имеет вид $k_j = \lceil \log_2(d_j + 1) \rceil$. При кодировке допустимым целочисленным точкам в Ω сопоставляются элементы B, состоящие из n последовательно записанных двоичных представлений координат:

$$x(\xi)_j = \sum_{i=0}^{k_j - 1} 2^i \xi_{k_1 + ... + k_j - i}, \quad j = 1, ..., n. \tag{1.9}$$

Таким образом, пространство генотипов B есть $\{0, 1\}^{k_1 + ... + k_n}$.

Оператор мутации вводится стандартным образом, а схема кроссинговера может быть адаптирована для данной задачи с помощью дополнительного условия: $\chi \in \{k_1, k_1 + k_2, ..., k_1 + ... + k_{n-1}\}$. (Строго говоря, при такой модификации алгоритм уже не является КГА.)

Функция приспособленности $\Phi(\xi)$ может быть определена по-разному. Определим вспомогательную функцию

$$f(x) = \begin{cases} F(x) & \text{при} \quad s(x) = 0 \\ -rs(x) & \text{при} \quad s(x) > 0, \end{cases}$$

где $s(x)$ – сумма нарушений системы ограничений (1.7) для точки x и r – некоторая положительная константа, величина которой достаточна, чтобы выполнялось $f(x') < f(x)$ для всех $x, x' \in \Omega \cap \mathbf{Z}$, таких что $x \in \mathcal{M}, x' \notin \mathcal{M}$.

Функция приспособленности может быть выбрана следующим образом [63]:

$$\Phi(\xi) = \frac{f(x(\xi)) - f^t_{min}}{f^t_{avg} - f^t_{min}},$$

где f^t_{avg}, f^t_{min} – среднее и минимальное значения функции $f(x(\xi))$ на текущей популяции Π^t. Легко видеть, что определенная таким образом функция $\Phi(\xi)$ неотрицательна и неубывает с ростом $f(x)$. Отметим, что такой вариант функции приспособленности позволяет решать с помощью ГА задачи без ограничения на знак целевой функции.

Решение задачи линейного программирования (1.6)-(1.7) назовем непрерывным оптимумом.

Модификация КГА: индивиды первого поколения порождаются с помощью n-мерного нормального распределения с математическим ожиданием в точке непрерывного оптимума и с последующим округлением дробных координат.

Эксперимент показал, что если вероятность мутации достаточно велика, дисперсия начального распределения может быть установлена равной нулю. Кроме того, выяснилось, что как правило ГА относительно быстро (по сравнению с точными алгоритмами, например, Гомори) обнаруживает допустимые решения, близкие к оптимальному по целевой функции, однако часто имеют место случаи преждевременной сходимости[3] ГА к некоторому приближенному решению, в результате процесс поиска оптимума замедляется. Для преодоления этого затруднения разработан точный гибридный алгоритм [55], сочетающий ГА с перебором L-классов.

1.2.6. Простейшая задача размещения производства

Рассматриваается следующая задача оптимального размещения производства, известная также как задача стандартизации [2]. Пусть имеется возможность построить m предприятий, каждое из которых может обслуживать

[3] В англоязычной литературе исполльзуется термин *premature convergence.*

любого из n клиентов. При этом открытие i-го предприятия $(i = 1, ..., m)$ стоит $c_i \geq 0$ единиц, а обслуживание j-го клиента $(j = 1, ..., n)$ на предприятии i обходится в $C_{ij} \geq 0$ единиц стоимости. Задача состоит в минимизации функционала

$$F(z) = \sum_{i=1}^{m} c_i z_i + \sum_{j=1}^{n} \min_{i:z_i=1} C_{ij},$$

при условии, что

$$\sum_{i=1}^{m} z_i \geq 1,$$

где $z_i \in \{0, 1\}$. Решение z^* представляет собой оптимальный вектор - набор предприятий при поставленных условиях, причем по вектору z^* легко могут быть назначены и оптимальные прикрепления клиентов к открытым предприятиям. Очевидно, что единственным недопустимым решением является нулевой вектор $z = \mathbf{0}$.

Не теряя общности, можно предположить, что все $c_i > 0$, т.к. при $c_i = 0$ предприятие i можно включить в вектор z решения задачи в обязательном порядке – полученный после этого план обслуживания будет оптимален. При этом, если предприятие i не оказалось назначенным ни для одного клиента, его можно исключить из вектора решения.

Опишем схему КГА в применении к простейшей задаче размещения. В качестве генотипа удобно взять вектор предприятий z. В таком случае пространство генотипов совпадает с пространством фенотипов и отображение $x(\xi)$ – тождественное. Для определения функции приспособленности необходимо исходную задачу минимизации с ограничениями свести к задаче максимизации без ограничений: положим что при некотором достаточно малом $\varepsilon > 0$

$$\Phi(\xi) = \begin{cases} \frac{1}{F(\xi)} & \text{при} \quad \xi \neq 0 \\ \varepsilon & \text{при} \quad \xi = 0. \end{cases}$$

Очевидно, что тогда любой допустимый вектор пространства решений имеет бóльшую приспособленность $\Phi(\xi) = 1/f(x(\xi)) = 1/f(\xi)$, чем нулевой недопустимый вектор. Операторы мутации и кроссинговера полностью соответствует КГА.

Другой возможный подход: ГА с недвоичным представлением, где $l = n$. $\xi_j \in \{1, ..., m\}$, $j = 1, ..., n$. Кодировка: предприятие i обслуживает клиента j тогда и только тогда, когда $\xi_j = i$, и если хотя-бы один клиент обслуживается предприятием i, то полагаем $z_i = 1$. Данная модификация генетического алгоритма уже не укладывается в схему КГА.

1.3. Теорема о схемах

Схемой H с K фиксированными позициями будем называть множество генотипов

$$H = \{\xi \in B | \xi_{j_1} = h_1, \xi_{j_2} = h_2, ..., \xi_{j_K} = h_K\},$$

где $j_1 < j_2,\ j_2 < j_3, ..., j_{K-1} < j_K$. Число K принято называть порядком схемы. Длиной $\delta(H)$ схемы H будем считать расстояние между крайними фиксированными позициями, т.е. $\delta(H) = j_K - j_1$. По определению полагаем $\delta(B) = 0$. Очевидно, что при заданной кодировке любая точка пространства состояний является частным случаем схемы порядка l.

Ввиду того, что одним генотипом могут обладать несколько особей популяции, далее удобно ввести обозначение $N(H, \Pi^t)$ для числа представителей схемы H в поколении t, аналогично $N(\xi, \Pi^t)$ – число представителей генотипа ξ в поколении Π^t. Введем специальное обозначение для среднего значения функции приспособленности на особях схемы H в поколении t:

$$\Phi(H, \Pi^t) = \frac{\sum\limits_{i:\xi^{i,t} \in H} \Phi(\xi^{i,t})}{N(H, \Pi^t)}.$$

Рассмотрим оценку среднего числа представителей схем среди особей нового поколения, иногда называемую теоремой о схемах или фундаментальной теоремой генетических алгоритмов [63, 72].

Теорема 1.3.1. *Пусть H – схема порядка K и величина c такова, что $\Phi(H, \Pi^t) \geq c\Phi(B, \Pi^t)$. Тогда в классическом генетическом алгоритме*

$$E[N(H, \Pi^{t+1})] \geq c \cdot \left(1 - \frac{\delta(H) P_{\mathrm{c}}}{l-1}\right) \cdot (1 - P_{\mathrm{m}})^K N(H, \Pi^t). \tag{1.10}$$

Доказательство. Будем рассматривать очередную итерацию КГА с номером $t+1$ в вероятностном пространстве, определенном описанной выше схемой КГА, его параметрами и совокупностью особей популяции t. Для начала рассмотрим вероятность того, что выбранный при селекции генотип принадлежит H. Она имеет вид

$$P\{\xi^{Sel(\Pi^t),t} \in H\} = \sum_{i:\xi^{it} \in H} \frac{\Phi(\xi^{it})}{\sum\limits_{j=1}^{\lambda} \Phi(\xi^{jt})} = \frac{\Phi(H, \Pi^t) N(H, \Pi^t)}{\Phi(B, \Pi^t)\lambda} \geq c \frac{N(H, \Pi^t)}{\lambda}. \tag{1.11}$$

Кроме оператора селекции необходимо учитывать также действие мутации и кроссинговера, которые могут разрушать, а могут и создавать особи схемы H. Для получения искомой нижней оценки будем рассматривать

только возможность разрушения элементов схемы H. Рассмотрим случайную величину $\zeta_i \in \{0,1\}$, равную 1, если $\xi^{i,t+1} \in H$, и 0 иначе. Если мы оценим снизу математическое ожидание для всех ζ_i, то это даст возможность получить оценку снизу и на величину $E[N(H,\Pi^{t+1})]$, т.к.

$$E[N(H,\Pi^{t+1})] = \sum_{i=1}^{\lambda} E[\zeta_i] = \sum_{i=1}^{\lambda} P\{\zeta_i = 1\}. \tag{1.12}$$

Построим нижнюю оценку для вероятности $P\{\zeta_i = 1\}$, рассматривая следующий сценарий возникновения этого события:

- родительский генотип, откуда при кроссинговере была скопирована позиция j_1, принадлежал H;
- при построении $\xi^{i,t+1}$ на этапе кроссинговера все гены с номерами $j_1, j_2, ..., j_K$ были скопированы из одной родительской особи;
- при мутации ни один бит, отвечающий за принадлежность к схеме H, не изменит свое значение.

Во-первых, неравенство (1.11) дает оценку вероятности того, что родительский генотип, откуда при кроссинговере была скопирована позиция j_1, принадлежал H. Во-вторых, из определения кроссинговера вытекает, что все гены с номерами $j_1, j_2, ..., j_K$ копируются из одной родительской особи с вероятностью $1 - \frac{\delta(H)P_c}{l-1}$. В-третьих, вероятность того, что при мутации ни один бит, отвечающий за принадлежность к схеме H, не изменит свое значение, равна $(1-P_m)^K$, т.к. генные мутации происходят побитно и независимо.

Таким образом, ввиду независимости операторов кроссинговера и мутации от всех других событий в ГА, получаем

$$P\{\zeta_j = 1\} \geq c\frac{N(H,\Pi^t)}{\lambda}(1-P_m)^K\left(1 - \frac{\delta(H)P_c}{l-1}\right). \tag{1.13}$$

Далее, с учетом (1.12) получаем (1.10). □

Таким образом, при выборе кодировки решений разработчик ГА должен стремиться к тому, чтобы перспективные свойства решений были бы представлены в генотипе в виде как можно более коротких участков хромосом. В таком случае эти свойства будут проще обнаруживаться в процессе работы ГА и решения с такими свойствами будут активно исследоваться.

Пример 1. Рассмотрим случай, когда число представителей схемы H, состоящей из близких к оптимуму генотипов, увеличивается. Пусть используется стандартная двоичная кодировка решений $x \in \{0,1,\ldots,2^l\}$ для функции

$f(x) \equiv x$ и $\Phi(\xi) = \sum\limits_{j=1}^{l} \xi_j 2^{l-j}$, $l \geq 2$. Очевидно, $x^* = 2^l - 1,\ \xi^* = (1, 1, ..., 1)$. Если фиксировать k первых единиц, $0 < k < l$, то в случае, когда начальная популяция содержит всевозможные генотипы по одному экземпляру, имеем

$$\Phi(H, \Pi^0) = \frac{\min\limits_{\xi \in H} \Phi(\xi) + \max\limits_{\xi \in H} \Phi(\xi)}{2} = \frac{\sum\limits_{j=1}^{k} 2^{l-j} + 2^l - 1}{2} =$$

$$\frac{2^l \sum\limits_{j=0}^{k} 2^{-j} - 1}{2} = \frac{2^l \cdot \frac{1-\left(\frac{1}{2}\right)^{k+1}}{1-\frac{1}{2}} - 1}{2}.$$

С другой стороны, $\Phi(B, \Pi^0) = \frac{2^l - 1}{2}$. Поэтому получаем оценку снизу:

$$\frac{\Phi(H, \Pi^0)}{\Phi(B, \Pi^0)} = \frac{2^{l+1}\left(1 - \frac{1}{2^{k+1}}\right) - 1}{2^l - 1} \geq 2\left(1 - \frac{1}{2^{k+1}}\right) = 2 - \frac{1}{2^k},$$

т.к. $2(1 - \frac{1}{2^{k+1}}) \geq 1$ и $(xa - 1)/(a - 1) \geq x$ при $a > 1, x \geq 1$.

Таким образом, пусть $c = 2 - 2^{-k}$ и $P_{\mathrm{c}} = 1, k = l/4$. Тогда теорема о схемах дает:

$$E[N(H, \Pi^1)] \geq c\left(1 - \frac{l/4}{l-1}\right)(1 - P_{\mathrm{m}})^{l/4} N(H, \Pi^0),$$

и при больших l правая часть приближается к $2 \cdot \frac{3}{4}(1 - P_{\mathrm{m}})^{l/4}$, что при $P_{\mathrm{m}} < 1 - \sqrt[l/4]{2/3}$ превышает 1, т.е. при достаточно малой вероятности мутации имеет место рост числа представителей схемы H в среднем. Например, при $l = 100$ получаем $P_{\mathrm{m}} < 0.00003$.

Пример 2. Если же фиксировать k последних единиц (обозначим такую схему H'), то

$$\Phi(H', \Pi^0) = \frac{\min\limits_{\xi \in H'} \Phi(\xi) + \max\limits_{\xi \in H'} \Phi(\xi)}{2} = \frac{2^k - 1 + 2^l - 1}{2} = \frac{2^k(1 + 2^{l-k}) - 2}{2},$$

и, следовательно,

$$\frac{\Phi(H', \Pi^0)}{\Phi(B, \Pi^0)} < \frac{1 + 2^{l-k}}{2^{l-k} - 2^{-k}}.$$

при $l - k \to \infty$ правая часть стремится к 1, т.е., при больших длинах l и существенно меньших k теорема о схемах не будет гарантировать рост числа представителей H'. Например, при $k = 0.4 \cdot l$ и $l = 100$ получаем

$$\frac{\Phi(H', \Pi^0)}{\Phi(B, \Pi^0)} < \frac{1 + 2^{60}}{2^{60} - 1} < 1.0000000000000000018.$$

В произведении со множителями, отвечающими за мутацию и кроссинговер, результат, скорее всего, уже будет меньше 1.

1.4. Анализ разнообразия популяции

Пусть $q \in \{1, \ldots, l\}$ – некоторая позиция в генотипе. Обозначим $H_q = \{\xi : \xi_q = 0\}$ и рассмотрим величину

$$a_q = \frac{\Phi(H_q, \Pi^t) N(H_q, \Pi^t)}{\Phi(B, \Pi^t)\lambda} = \frac{\sum_{i:\xi^i \in H_q} \Phi(\xi^{it})}{\sum_{k=1}^{\lambda} \Phi(\xi^{kt})},$$

как характеристику «качества» нулевого значения гена в позиции q и его распространения в популяции Π^t.

Получим формулы, определяющие зависимость между вероятностью вырождения гена в позиции q от параметров генетического алгоритма. Заметим, что a_q есть вероятность выбора особи, принадлежащей H_q из популяции Π^t при пропорциональной селекции.

Теорема 1.4.1. *[20] Для любого значения $q \in \{1, \ldots, l\}$ в КГА:*

$$P\{N(H_q, \Pi^{t+1}) = \lambda\} = (a_q + (1 - 2a_q)P_{\mathrm{m}})^{\lambda}, \tag{1.14}$$

$$P\{N(H_q, \Pi^{t+1}) = 0\} = (1 - a_q + (2a_q - 1)P_{\mathrm{m}})^{\lambda}. \tag{1.15}$$

Доказательство. Рассмотрим подробно только равенство (1.14), так как доказательство (1.15) проводится аналогично. Рассчитаем вероятность того, что для каждого $i = 1, \ldots, \lambda/2$, имеет место $\xi^{2i,t+1} \in H_q$ и $\xi^{2i-1,t+1} \in H_q$, т.е. пара передаваемых в популяцию Π^{t+1} особей будет иметь нулевое значение в позиции q. Обозначим, соответственно, η^{2i} и η^{2i-1} генотипы этих особей перед применением к ним оператора мутации. Тогда

$$P\{\xi_q^{2i,t+1} = 0, \xi_q^{2i-1,t+1} = 0\} = P\{\eta_q^{2i} = \eta_q^{2i-1} = 0\}(1 - P_{\mathrm{m}})^2 + \tag{1.16}$$

$$+2P\{\eta_q^{2i} = 1,\ \eta_q^{2i-1} = 0\}(1 - P_{\mathrm{m}})P_{\mathrm{m}} + P\{\eta_q^{2i} = \eta_q^{2i-1} = 1\}P_{\mathrm{m}}^2.$$

Рассчитаем вероятность $P\{\eta_q^{2i} = 0, \eta_q^{2i-1} = 0\}$. В результате скрещивания могут получиться две особи, у которых в позиции q находится значение 0, только в том случае, если у обеих родительских особей в позиции q находилось значение 0. Вероятность выбора таких особей, с учетом вероятностного смысла величин a_q и независимости селекции каждой особи, равна $(a_q)^2$.

Аналогично находим вероятность того, что у одной из полученных в результате скрещивания особей в позиции q находится значение 1, а у второй – значение 0. $P\{\eta_q^{2i} = 1,\ \eta_q^{2i-1} = 0\} = a_q(1 - a_q)$.

Далее, $P\{\eta_q^{2i} = \eta_q^{2i-1} = 1\} = (1 - a_q)^2$, так как в результате скрещивания могут получиться две особи, у которых в позиции q находится значение 1,

только в том случае, если это значение находилось в позиции q у обеих родительских особей.

Подставив найденные выражения в формулу (1.16), для каждого i, $i = 1, \ldots, \lambda/2$ получим:

$$P\{\xi_q^{2i,t+1} = 0, \xi_q^{2i-1,t+1} = 0\} = \tag{1.17}$$
$$(a_q)^2(1-P_\mathrm{m})^2 + 2a_q(1-a_q)P_\mathrm{m}(1-P_\mathrm{m}) + (1-a_q)^2P_\mathrm{m}^2 =$$
$$= (a_q(1-P_\mathrm{m}) + (1-a_q)P_\mathrm{m})^2 = (a_q + (1-2a_q)P_\mathrm{m})^2.$$

Так как $\lambda/2$ пар особей в Π^{t+1} генерируются независимо одним и тем же способом, то $P\{N(H_q, \Pi^t) = \lambda\}$ равна

$$P\{\xi_q^{2i,t+1} = 0, \xi_q^{2i-1,t+1} = 0 \ \forall\ i = 1, \ldots, \lambda/2\} = (a_q + (1-2a_q)P_\mathrm{m})^\lambda.$$

□

Данная теорема позволяет сделать вывод о том, что вероятность вырождения гена и, следовательно, степень разнообразия популяции зависят только от вероятности мутации и размера популяции, но не зависят от вероятности кроссинговера. Таким образом проявляется важное свойство оператора одноточечного кроссинговера, который порождает новые комбинации из уже имеющихся «блоков», но при этом никакие элементарные «блоки» не теряются. Более детальное обсуждение взаимодействия кроссинговера с модульными структурами в ЭА и в живой природе можно найти в [25].

Из теоремы следует, что вероятность вырождения значения 0 или 1 в гене q равна

$$p(q, P_\mathrm{m}, \lambda) = (a_q + (1-2a_q)P_\mathrm{m})^\lambda + (1 - a_q + (2a_q - 1)P_\mathrm{m})^\lambda.$$

Таким образом, при $0 < a_q < 1$ вероятность вырождения гена в любой позиции уменьшается с увеличением размера популяции λ, а также с приближением P_m к $1/2$, ибо условие $\frac{\partial p(q,P_\mathrm{m},\lambda)}{\partial P_\mathrm{m}} = 0$ эквивалентно

$$(a_q + (1-2a_q)P_\mathrm{m})^{\lambda-1} = (1 - a_q + (2a_q - 1)P_\mathrm{m})^{\lambda-1},$$

что означает $P_\mathrm{m} = 1/2$, т.к. на концах интервала $p(q, 0, \lambda) = p(q, 1, \lambda) \geq p(q, 1/2, \lambda)$. Таким образом, наибольшее разнообразие достигается при $P_\mathrm{m} = 1/2$, как и следовало ожидать.

На практике использование $P_\mathrm{m} = 1/2$ не целесообразно, т.к. это эквивалентно случайной инициализации популяции на каждом поколении ГА. Вместо этого в современных ЭА используются процедуры автоматической настройки параметров алгоритма [51], эмпирический подбор этих параметров на основании вычислительных экспериментов или теоретически обоснованный выбор параметров, исходя из свойств функции приспособленности и операторов ЭА, например, как в п. 1.5 ниже.

1.5. Время первого достижения оптимума

В теории ЭА, как правило, рассматриваются вопросы трудоемкости отыскания наилучшего возможного генотипа в процессе работы некоторого ЭА на выбранном классе задач. При этом в первую очередь интересуются средним числом пробных решений до первого получения оптимального генотипа, в зависимости от размерности задачи. Исследуемые классы задач могут содержать примеры сколь угодно большой размерности, как это принято в теории вычислительной сложности [8]. Для анализа времени первого достижения оптимального генотипа, как правило, используются такие методы теории вероятностей, как цепи Маркова, мартингалы, случайные процессы со сносом, стохастическое доминирование и др.

Рассмотрим КГА при $p_c = 0,\ p_m = \chi/n$ с постоянным значением параметра $\chi > \ln 2$. Как выясняется, этот алгоритм неэффективен даже на классе линейных функций приспособленности.

Теорема 1.5.1. *[45] Если $f(x) \equiv \sum_{i=1}^{n} a_i x_i$, то КГА при численности популяции $\lambda \geq n^{2+\delta}$, и вероятности мутации χ/n при константных $\delta > 0$, $\chi > \ln(2)$, получает оптимум функции f с вероятностью не более $\lambda e^{-dn^{\delta}}$ за e^{cn} поколений, где c, d – положительные константы.*

Аналогичный вывод был получен из анализа КГА при $p_c = 1$ в случае функции OneMax [82, 83]. По определению, $ONEMAX(\mathbf{x}) = \sum_{i=1}^{n} x_i$ на всем множестве $\mathrm{Sol} = X = \{0,1\}^n$.

Если же вероятность мутации снизить на порядок, то алгоритм будет за полиномиальное время находить оптимум, как показано в следующей теореме.

Теорема 1.5.2. *[45] Если $f(x) = \sum_{i=1}^{n} a_i x_i$, то КГА при $p_c = 0$, $p_m = \chi/n$, где $\chi = (1-c)/(na_{\max})$, $a_{\max} := \max_{i=1}^{n} a_i$, для любой положительной константы $c < 1$, и размере популяции $\lambda \geq dn^2 a_{\max}^2 \ln(na_{\max})$ при достаточно большой константе $d > 0$, имеет среднее время достижения оптимума $O(n^2 + n\lambda \ln \lambda)$.*

Аналогичный результат получен в [45] для КГА с вероятностью мутации χ/n, где χ-константа, но при условии, что приспособленность имеет вид $f(x) = s^{\sum_{i=1}^{n} a_i x_i}$, и $s > e^{\chi}$. Эти оценки имеют, в основном, теоретическую ценность, т.к. на практике прменение ЭА для поиска экстремума в задачах с линейной функцией приспособленности нецелесообразно.

2. Модификации генетических алгоритмов

2.1. Операторы селекции

Для сравнения различных операторов селекции, которые будут описаны далее, необходимо выбрать некоторую характеристику, которая имеет одинаковый смысл для всех из них. Возмем в качестве такой характеристики число, показывающее сколько раз фиксированная особь отбирается в качестве родительской из популяции Π^t в процессе построения очередного поколения.

2.1.1. Стохастическая универсальная селекция Бэкера

Упрощенно действие стохастической универсальной селекции Бэкера [32][1] может быть описано по аналогии с селекцией методом рулетки. Селекция Бэкера осуществляется путем определения размера секторов рулетки также как в селекции из КГА. Представим себе, что λ маркеров размещено на равном расстоянии вдоль внешней стороны колеса рулекти и вращение рулетки осуществляется один раз на случайный угол. Количество потомков, которое получает особь, рассчитывается как число маркеров, попавших в соответствующий сектор. Для снижения вероятности попадания двух идентичных генотипов на вход кроссинговера выбранные генотипы родителей перемешиваются случайным образом.

Для формального описания данного алгоритма рассмотрим всю последовательность генотипов, выбранных оператором селекции в процессе построения очередной популяции (далее – выходная последовательность): $\xi^{i_1,t}, \ldots, \xi^{i_\lambda,t}$. Индексы $i_1, \ldots, i_\lambda$ – случайные величины. Оператор кроссинговера на итерации t примененяется в следующем порядке: $\mathrm{Cross}(\xi^{i_1,t}, \xi^{i_2,t}); \mathrm{Cross}(\xi^{i_3,t}, \xi^{i_4,t}); \ldots \mathrm{Cross}(\xi^{i_{\lambda-1},t}, \xi^{i_\lambda,t})$.

Алгоритм стохастической универсальной селекции Бэкера
1. Сгенерировать случайную величину x с равномерным распределением на $[0, 1]$.
2. Положить $i_k, k = 1, \ldots, \lambda$:

$$i_k = \min\left(i : \sum_{j=1}^{i} \frac{\Phi(\xi^{j,t})}{\sum_{s=1}^{\lambda} \Phi(\xi^{s,t})} \geq \left\{\frac{k}{\lambda} + x\right\}\right).$$

[1] Baker's stochastic universal selection

Здесь и далее фигурные скобки $\{\cdot\}$, заключающие вещественное число, обозначают дробную часть данного числа.
3. Применить случайную перестановку к последовательности $i_1, \ldots, i_\lambda$.

Пусть $Z_B(i, \Pi^t)$ – сколько раз особь с номером i отбирается в качестве родительской из популяции Π^t при селекции Бэкера.

Утверждение 2.1.1.

$$Z_B(i, \Pi^t) = \lfloor \lambda P_{sel}(i, \Pi^t) \rfloor + u_{it},$$

где $P_{sel}(i, \Pi^t)$ определена как в стандартной рулеточной селекции, а случайная величина $u_{it} \in \{0, 1\}$ такова, что $P\{u_{it} = 1\} = \{\lambda P_{sel}(i, \Pi^t)\}$.

Следующее утверждение показывает, что для повышения стабильности результатов в ГА предпочтительнее использовать селекцию Бэкера. Введем случайную величину $Z_R(i, \Pi^t)$ для стандартной рулеточной селекции по аналогии со случайной величиной $Z_B(i, \Pi^t)$.

Утверждение 2.1.2. *Оператор селекции Бэкера осуществляет пропорциональную селекцию, и при этом $D[Z_B(i, \Pi^t)] \leq 1/4$, в то время как $D[Z_R(i, \Pi^t)]$ может расти неограниченно с ростом λ.*

Доказательство. По формулам, характеризующим схему Бернулли для стандартной рулеточной селекции имеем: $D[Z_R(i, \Pi^t)] = \lambda p(1-p)$, где $p = P_{sel}(i, \Pi^t)$.

С другой стороны, для селекции Бэкера

$$E[Z_B(i, \Pi^t)] = \lfloor \lambda p \rfloor + \{\lambda p\} = \lambda p = E[Z_R(i, \Pi^t)];$$

$$D[Z_B(i, \Pi^t)] = (1 - \{\lambda p\})\{\lambda p\}^2 + \{\lambda p\}(1 - \{\lambda p\})^2 = \{\lambda p\}(1 - \{\lambda p\}) \leq 1/4.$$

Q.E.D.

2.1.2. Ранжирование особей в операторах селекции

В процессе работы ГА, как правило, происходит сближение приспособленности особей в популяции. В результате операторы пропорциональной селекции все меньше «отличают» (в смысле вероятности селекции) наиболее приспособленных особей от отстающих. Способ решения этой проблемы, предложенный Д.Голдбергом, состоит в масштабировании приспособленности в процессе работы ГА [63] (см. приведенный выше пример применения ГА к задаче ЦЛП). Альтернативный подход состоит в использовании *ранжирования особей.*

Для фиксированной популяции Π биекция $r_\Pi : \{1, ..., \lambda\} \to \{1, ..., \lambda\}$ называется ранжированием, если для всех $i, j \in \{1, 2, ..., \lambda\}$ выполняется:

$$\Phi(\xi^i) > \Phi(\xi^j) \Rightarrow r_\Pi(i) > r_\Pi(j).$$

Значение $r_\Pi(i)$ называется рангом особи ξ^i в популяции Π.

2.1.3. Ранговая селекция

Ранговая селекция (ranking selection) предложена в работе [65]. Здесь $\mathbb{R}_+$ обозначает множество неотрицательных вещественных чисел и пусть функция $\alpha : \{1, \ldots, \lambda\} \to \mathbb{R}_+$, такая что $\sum_{r=1}^{\lambda} \alpha(r) = 1$. Тогда α называется ранжирующей функцией.

При заданной ранжирующей функции оператор селекции с распределением вероятностей

$$P\{\text{выбрать особь с рангом } r\} = \alpha(r), \quad r = 1, \ldots, \lambda,$$

называется ранжирующей селекцией. Такая селекция не теряет «чувствительности» при сколь угодно малых различиях приспособленности особей.

Частный случай, где

$$\alpha(r) = \frac{\eta - 1}{\lambda} \left(\frac{2(r - \lambda)}{\lambda - 1} + \frac{\eta}{\eta - 1} \right),$$

при $\eta \in (1, 2]$ называется линейным ранжированием. Легко видеть, что условия ранжирующей функции выполняются.

Особь с рангом λ имеет вероятность селекции, равную η/λ, а особь с рангом 1 – вероятность $(2 - \eta)/\lambda$. Если положить $\eta = 2$, то особь с рангом 1 имеет нулевую вероятность селекции (наибольшая дифференциация селективности по рангу). Если же $\eta \to 1$, то распределение вероятностей селекции стремится к равномерному.

Ранговая селекция при $\eta = 2 - 2/(\lambda + 1)$ состоит в применении оператора селекции КГА с подстановкой рангов особей $r_\Pi(i)$ вместо значений их приспособленности.

2.1.4. Турнирная селекция

Оператор турнирной селекции (tournament selection) с размером турнира s (или оператор s-турнирной селекции) при построении очередного решения из текущей популяции извлекает s особей с равномерным распределением и выбирает лучшую из них (точнее, особь с наибольшим рангом).

Сравним среднее число повторений фиксированной особи ξ^{it} при 2-турнирной селекции (обозначаем далее через $Z_T(i, \Pi^t)$) и при пропорциональной селекции КГА с подстановкой рангов особей вместо значений их приспособленности (обозначаем через $Z_{RR}(i, \Pi^t)$). Заметим, что в турнирной селекции вероятность выбора особи i с рангом $r = r_{\Pi^t}(i)$ есть

$$p(r) = C_s^1 \frac{1}{\lambda} \left(\frac{r - 1}{\lambda} \right)^{s-1} + C_s^2 \left(\frac{1}{\lambda} \right)^2 \left(\frac{r - 1}{\lambda} \right)^{s-2} + \ldots + C_s^s \left(\frac{1}{\lambda} \right)^s =$$

$$= \left(\frac{1}{\lambda} + \frac{r-1}{\lambda}\right)^s - \left(\frac{r-1}{\lambda}\right)^s = \left(\frac{r}{\lambda}\right)^s - \left(\frac{r-1}{\lambda}\right)^s.$$

В частности, при $s = 2$ имеем:

$$E[Z_T(i, \Pi^t)] = \lambda p(r) = \frac{r^2 - r^2 + 2r - 1}{\lambda} = \frac{2r-1}{\lambda}.$$

Сравним эту величину с $E[Z_{RR}(i, \Pi^t)]$. Легко видеть, что при использовании ранжирования в пропорциональной селекции

$$P_{sel}(i, \Pi^t) = \frac{2r}{\lambda(\lambda+1)}, \quad E[Z_{RR}(i, \Pi^t)] = \frac{2r}{\lambda+1},$$

что приближается к $E[Z_T(i, \Pi^t)]$ при больших r, λ.

По формуле дисперсии для схемы Бернулли нетрудно показать, что

$$D[Z_{RR}(i, \Pi^t)] = \frac{2r(\lambda^2 - 2r + \lambda)}{\lambda^3 + 2\lambda^2 + \lambda},$$

$$D[Z_T(i, \Pi^t)] = \frac{(2r-1)(\lambda^2 - 2r + 1)}{\lambda^3},$$

и при $\lambda \to \infty$ имеем $\frac{D[Z_{RR}(i,\Pi^t)]}{D[Z_T(i,\Pi^t)]} \to \frac{2r}{2r-1}$, следовательно, при больших значениях λ и r, также и по дисперсии оператор 2-турнирной приближается к пропорциональной селекции с рангами вместо приспособленности.[2]

Упражнение 2.1.1. *Найти функцию $\alpha(r)$ для турнирной селекции.*

2.1.5. (μ, λ)-селекция

(μ, λ)-селекция – один из наиболее простых операторов селекции в ЭА, аналогичен массовому отбору в растениеводстве и животноводстве: из популяции численностью λ отбираются μ особей с наибольшими значениями функции приспособленности и родительские генотипы равновероятно выбираются из них для скрещивания и получения потомства.

2.2. Стратегии управления популяцией

Обновление всей популяции на каждой итерации КГА соответствует подходу, применяемому при имитационном моделировании в популяционной генетике (см., например, [1]), однако, для ускорения поиска генотипов с высокой приспособленностью, общая схема ГА зачастую модифицируется. Основная мотивация при этом состоит в том, что в КГА даже генотип, существенно

[2]Заметим, что именно особи с большим рангом r оказывают наибольший эффект при построении очередной популяции.

превышающий по пригодности все прочие особи популяции, с большой вероятностью будет исключен из рассмотрения уже на следующей итерации после его появления. Однако, в успешных приложениях ГА приспособленность потомков, как правило, имеет положительную корреляцию с приспособленностью родительских генотипов. В таких случаях целесообразно сохранять наиболее пригодные особи в течение ряда итераций ГА и генерировать с помощью кроссинговера и мутации оставшуюся часть популяции. Рассмотрим некоторые известные схемы управления популяцией, реализующие этот принцип.

Элитарная стратегия. На кажой итерации ГА, во-первых, строится очередная популяция Π^{t+1} по правилам КГА. Во-вторых, если по приспособленности все генотипы новой популяции уступают максимально приспособленной (*элитной*) особи $\xi_{\mathtt{e}}^t$ из предыдущей популяции, то один из наименее приспособленных генотипов в Π^{t+1} заменяется на $\xi_{\mathtt{e}}^t$.

Таким образом, если ГА применяется для решения задачи безусловной оптимизации (1.1), то последовательность значений целевой функции элитных фенотипов $f(x(\xi_{\mathtt{e}}^1)), f(x(\xi_{\mathtt{e}}^2)), \ldots$ будет неубывающей. Более того, как показал Г. Рудольф [95], при $x(B) = X$ и $0 < P_{\mathtt{m}} < 1$, дополнение КГА элитарной стратегией обеспечивает сходимость последовательности $f(x(\xi_{\mathtt{e}}^1)), f(x(\xi_{\mathtt{e}}^2)), \ldots$ к оптимальному значению целевой функции задачи (1.1) почти наверное.

Вместо одной элитной особи в ГА может сохраняться некоторое подмножество генотипов текущей популяции, имеющих высокую приспособленность (см., например, [55]). Такие стратегии называются *частичной заменой популяции*. Следующая стратегия может рассматриваться как предельный случай расширения множества элитных особей.

Стационарная стратегия управления популяцией. При этой стратегии на каждой итерации ГА в популяцию добавляются два генотипа, полученных применением операторов кроссинговера и мутации. Каждая новая особь замещает некоторый «неперспективный» генотип. При этом в качестве «неперспективного» может быть взят генотип с наименьшей приспособленностью, или генотип, выбранный с равномерным распределением среди имеющих приспособленность ниже средней в текущей популяции. В некоторых вариантах ГА на выходе кроссинговера имеется только один генотип – тогда изменяется только одна особь популяции.

Особенностью стационарной стратегии управления популяцией является значительно более быстрое «сужение» области поиска, по сравнению с КГА. В связи с этим, во многих реализациях стационарной стратегии управления популяцией при совпадении новой особи с одной из имеющихся в популяции, новая особь в популяцию не добавляется (см., например, [35, 52]).

Элитная рекомбинация (elitist recombination) [64] (не путать с популяцией с элитой): особи текущей популяции случайным образом переставляются и последовательно выбираются пары родительских особей

$(\xi^{1t}, \xi^{2t}), (\xi^{3t}, \xi^{4t}), \ldots$ для скрещивания. Каждая пара потомков сравнивается с соответствующими родительскими особями, и лучшие две из четырех особей помещаются в новую популяцию.

Правило перезапуска. Поскольку сохранение элитной особи создает риск локализации процесса поиска вблизи однажды найденных «хороших» решений, имеет смысл добавить внешний цикл в общую схему ГА, позволяющий многократно запускать алгоритм, пока не будет исчерпан общий лимит по времени счета или по числу вычислений целевой функции, или пока не сработает какое-то специальное правило остановки, например, получение выполняющего набора логических переменных в задаче SAT. Таким образом, ГА выполняется несколько раз с независимой инициализацией начальной популяции и с некоторым правилом перезапуска, срабатывающим в случае «стагнации» популяции.

Один из вариантов правила перезапуска (см., например, [68]) выглядит следующим образом. Обозначим через θ число поколений ГА, прошедшее с начала последней инициализации ГА до момента последнего улучшения рекордного значения приспособленности в популяции. Будем полагать, что условие перезапуска выполняется, если за последние θ поколений не было ни одного улучшения рекорда по приспособленности и, кроме того, $t > \lambda$. Последнее условие вводится для того, чтобы предотвратить преждевременный перезапуск ГА уже на первых итерациях алгоритма.

Как показывает практика, использование такого правила перезапуска на сложных многоэкстремальных задачах, как правило, приводит к более быстрому получению глобального оптимума, чем ГА без перезапуска. Дальнейшее сокращение времени получения оптимума возможно при использовании правил перезапуска, учитывающих структуру популяции – см., например, [59].

2.3. Операторы скрещивания и мутации

Наряду с оператором одноточечного кроссинговера КГА, в генетических алгоритмах используются и другие операторы рекомбинации родительских генотипов. Общей чертой для всех из них является, так называемое, свойство *передачи генов*: значение для каждого гена потомка выбирается из значений соответствующих генов одного или другого родителей.[3]

В некоторых вариантах кроссинговера результатом является один генотип (см., например, [9]), однако, наиболее распространены операторы с двумя выходными генотипами. Во втором случае, с целью сохранения разнообразия популяции, стремятся построить как можно более удаленные один от другого генотипы потомков.

[3]Данное свойство в работах N. Radcliffe названо *свойством передачи генов* (gene transmission) [88].

Пусть даны родительские генотипы ξ и η, для которых порождается пара генотипов потомков ξ', η'. Если в задаче отсутствуют ограничения, или схема представления решений такова, что $x(B) \subseteq D$, тогда уместно использовать, так называемую, *маску кроссинговера*. Под этим термином понимают вспомогательную последовательность $\mathbf{m} = (m_1, \ldots, m_l) \in B$, по которой строятся генотипы потомков:

$$\xi'_i = \begin{cases} \xi_i, & \text{если } m_i = 1 \\ \eta_i, & \text{иначе,} \end{cases} \quad ; \quad \eta'_i = \begin{cases} \eta_i, & \text{если } m_i = 1 \\ \xi_i, & \text{иначе,} \end{cases}$$

для $i = 1, \ldots, l$. Рассмотрим два примера использования маски кроссинговера.

Равномерный кроссинговер. Данный оператор определяется выбором маски кроссинговера с равномерным распределением на множестве B. При действии этого оператора i-ый ген, $i = 1, \ldots, l$, копируется в генотип потомка из i-той позиции генотипа одного или другого родителя с равными вероятностями, независимо от выбора других генов.

k-точечный кроссинговер. Данный оператор представляет собой обобщение одноточечного кроссинговера. В строке генотипа выбирается k различных координат скрещивания $0 < \chi_1 < \chi_2 < \ldots < \chi_k < l$ с равномерным распределением среди всевозможных таких наборов. Обозначим $\chi_0 = 0$, тогда маска кроссинговера определяется следующим образом:

$$m_i = \begin{cases} 1, & \text{если } \max\{j : \ \chi_j < i\} \text{ – четное число} \\ 0, & \text{иначе} \end{cases}$$

для $i = 1, \ldots, l$. Данный оператор имеет то свойство, что при задании четного числа точек скрещивания k, первая и последняя координаты одного родителя всегда переходят одному потомку. Наоборот, при нечетном k эти координаты копируются в каждый из генотипов потомков от разных родителей.

Каждый из описанных операторов кроссинговера может быть реализован и в варианте с одним генотипом потомка (для этого достаточно отбросить второй генотип).

2.3.1. Кроссинговер с частичным отображением для задач на перестановках

Рассмотрим некоторые операторы, применяемые в задаче коммивояжера и других задачах, где допустимыми решениями являются перестановки. При описании этих операторов под случайным выбором понимается выбор с равномерным распределением среди всех возможных вариантов.

Оператор кроссинговера с частичным отображением был предложен в работе Д. Голдберга и Р. Лингле [62] и кратко обозначается PMX.[4]

Пусть пара перестановок ξ' и η' вычисляется по заданным родительским перестановкам ξ и η. Два индекса χ_1 и χ_2, где $\chi_1 < \chi_2$, выбираются с равномерным распределением. Наша цель состоит в том, чтобы выполнить обмен участками генотипов от χ_1 до χ_2, а все прочие гены достроить допустимым образом. Рассмотрим первый генотип-потомок ξ'. Положим сначала $\xi' := \xi$. Для получения последовательности генов $\xi'_j = \eta_j$ для всех $j = \chi_1, ..., \chi_2$ достаточно пройти по этим позициям и поместить на них требуемые значения генов с помощью не более чем $\chi_2 - \chi_1$ транспозиций. Это всегда возможно, т.к. η – тоже перестановка. Аналогично строится η' на основе η и участка первого генотипа от χ_1 до χ_2.

Рассмотрим действие кроссинговера PMX на иллюстративном примере. Пусть даны следующие родительские генотипы с координатами скрещивания $\chi_1 = 3, \chi_2 = 7$:

$$\begin{array}{ll} \xi = (& 1 \quad 2 \quad 3 \mid 4 \quad 5 \quad 6 \quad 7 \mid 8 \quad 9) \\ \eta = (& 4 \quad 5 \quad 2 \mid 1 \quad 8 \quad 7 \quad 6 \mid 9 \quad 3). \end{array}$$

Результат PMX-кроссинговера имеет вид:

$$\begin{array}{ll} \xi' = (& 4 \quad 2 \quad 3 \mid 1 \quad 8 \quad 7 \quad 6 \mid 5 \quad 9) \\ \eta' = (& 1 \quad 8 \quad 2 \mid 4 \quad 5 \quad 6 \quad 7 \mid 9 \quad 3). \end{array}$$

Заметим, что некоторые сочетания родительских генотипов при определенных χ_1, χ_2 могут давать различные результаты скрещивания, зависящие от последовательности выполнения перестановок между позициями χ_1 и χ_2. В связи с этим, для увеличения разнообразия популяции, имеет смысл выполнять перестановки в случайном порядке.

2.3.2. Порядковый кроссинговер для задач на перестановках

Порядковый кроссинговер [47] сохраняет абсолютные позиции элементов, заимствованных от одного родителя, и относительные позиции элементов, заимствованных у другого. Проиллюстрируем на примере:

$$\begin{array}{cc|ccccc} 2 & 1 & 3 & 4 & 5 & 6 & 7 \\ & & & & & & \\ 4 & 3 & 6 & 2 & 7 & 1 & 5 \\ & \chi & & & & & \end{array} \quad \longrightarrow \quad \begin{array}{cc|ccccc} 2 & 1 & 4 & 3 & 6 & 7 & 5 \\ & & & & & & \\ 4 & 3 & 2 & 1 & 5 & 6 & 7 \\ & \chi & & & & & \end{array}$$

[4] От английского *partially mapped crossover*.

Данный оператор сохраняет абсолютные позиции элементов, заимствованных от одного родителя, и относительные позиции элементов, заимствованных у другого.

Как показали эксперименты, порядковый кроссинговер, также как кроссинговер PMX, дает хорошие результаты в задачах составления расписаний, где большое значение имеет то, какие работы выполняются раньше, а какие – позже. В то же время, для задачи коммивояжера и задач составления расписаний, где важно учитывать длительность переналадки с одной работы на другую, лучшие результаты дают операторы рекомбинации, основанные на наследовании свойства смежности вершин (см., например, [41, 56]).

2.3.3. Мутация в задачах на перестановках

Мутация обмена[5] состоит в обмене пары генов из случайно выбранных позиций в данной на вход перестановке. С точки зрения локального поиска для задачи коммивояжера, при действии этого оператора выполняется шаг в случайно выбранную точку из окрестности *2-city swap* [19].

Мутация сдвига[6] состоит в перемещении гена из случайно выбранной позиции на случайное число позиций влево или вправо. Содержимое всех промежуточных генов при этом сдвигается на одну позицию.

Мутация «2-замена» определяется наиболее просто в случае задачи коммивояжера в терминах фенотипов, то есть обходов графа G. В обходе, заданном входным генотипом случайным образом выбираются два несмежных ребра и заменяются двумя новыми ребрами, которые в данном случае определяются однозначно. С точки зрения локального поиска, действие данного оператора представляет собой шаг в случайно выбранную точку из окрестности, определенной относительно 2-замены [21].

Упражнение 2.3.1. *Описать алгоритм, осуществляющий мутацию «2-замена» в кодировке решений задачи коммивояжера с помощью перестановок.*

2.4. Задача оптимальной рекомбинации

Пусть решается задача условной максимизации в пространстве двоичных строк длины $\ell = n$. Рассмотрим вычислительную сложность задачи отыскания «наилучшего» генотипа, как результата кроссинговера для заданной пары родительских генотипов при условии выполнения свойства передачи генов.

[5] В англоязычной литературе принят термин *exchange mutation.*

[6] В англоязычной литературе принят термин *shift mutation.*

С учетом свойства передачи генов сформулируем *задачу оптимальной рекомбинации*: для произвольных заданных родительских генотипов p^1, p^2, представляющих допустимые решения, требуется найти представляющий допустимое решение генотип ξ, такой что:

1) для каждого $j = 1, \ldots, n$ выполняется $\xi_j = p^1_j$ или $\xi_j = p^2_j$;

2) ξ имеет максимальное значение приспособленности среди всех генотипов, удовлетворяющих условию 1) и кодирующих при этом допустимые решения.

Далее множество номеров координат, в которых родительские генотипы различны, будем обозначать через $D(p^1, p^2)$.

В качестве примера эффективно разрешимой задачи оптимальной рекомбинации рассмотрим следующую известную задачу из теории графов. Пусть имеется граф $G = (V, E)$ с множеством вершин $V = \{v_1, \ldots, v_n\}$ и множеством ребер E. Задача о наибольшем независимом множестве состоит в отыскании такого подмножества $S \subseteq V$, что ни одно ребро $e \in E$ не инцидентно сразу двум вершинам из S (т.е. S – независимое множество) и мощность этого множества максимальна.

Естественным будет представление решений с помощью вектора-индикатора из $\{0,1\}^n$, где $\xi_j = 1$ тогда и только тогда, когда вершина v_j принадлежит искомому подмножеству. Пусть $\Phi(\xi) = |x(\xi)|$ для любого допустимого решения $x(\xi)$. Как замечено в работе Э.Балаша и В.Нихауса [33], при использовании данного представления решений задача оптимальной рекомбинации разрешима за полиномиальное время.

Для того чтобы в этом убедиться, рассмотрим произвольные родительские независимые множества S_1 и S_2 и соответствующие им генотипы p_1 и p_2. Исходя из свойства передачи генов, решение-потомок S должно содержать все множество вершин $L = S_1 \cap S_2$, кроме того, в S не должно быть элементов множества $V \setminus (S_1 \cup S_2)$, а вершины с номерами из множества $D(p^1, p^2)$ необходимо выбрать оптимальным образом. Последнее требование формулируется, как задача о наибольшем независимом множестве в подграфе, порожденном множеством вершин с номерами из $D(p^1, p^2)$. Легко видеть, что данный подграф является двудольным.

Для отыскания наибольшего независимого множества в двудольном графе $H = (V', E')$ можно воспользоваться тем фактом, что наибольшее независимое множество всегда является дополнением наименьшего вершинного покрытия C', то есть такого наименьшего по мощности множества вершин, что каждое ребро инцидентно хотя бы одной из них.

Задача о наименьшем вершинном покрытии двудольного графа $H = (V', E')$ эффективно разрешима с помощью алгоритма построения минимального разреза во вспомогательном графе, состоящем из графа H и дополнительных вершины-источника v_0 и вершины-стока v_{n+1}. Источник v_0 соединяется со всеми вершинами одной доли, а сток v_{n+1} – со всеми вершинами дру-

гой доли. Ребрам из множества E' приписываются бесконечные пропускные способности, а ребрам, инцидентным дополнительным вершинам – единичные пропускные способности. Наименьшее вершинное покрытие C' формируется из вершин, инцидентных ребрам минимального разреза.

Генотип ξ, являющийся вектором-индикатором множества $L \cup (V' \setminus C')$ представляет собой решение задачи оптимальной рекомбинации для задачи о наибольшем независимом множестве.

Приведенный результат Балаша и Нихауса может быть сформулирован как

Теорема 2.4.1. *(Балаш, Нихаус [33]) Задача оптимальной рекомбинации для задачи о независимом множестве разрешима за полиномиальное время.*

Упражнение 2.4.1. *Показать, что для задачи о наименьшем вершинном покрытии при тех же предположениях о способе представления решений (т.е. $v_j \in C \Leftrightarrow \xi_j = 1$) задача оптимальной рекомбинации эффективно разрешима.*

Рассмотренная здесь постановка задачи оптимальной рекомбинации может быть модифицирована – см., например, [3, 9]. Выбор наиболее подходящей формулировки этой подзадачи и методов ее решения делается на основе вычислительного эксперимента.

2.5. Генетический алгоритм как метод локального поиска

2.5.1. Задачи комбинаторной оптимизации

Пусть $\{0,1\}^*$ обозначает множество всевозможных строк из нулей и единиц произвольной длины, а $\mathbb{N}$ – множество натуральных чисел. Для $S \in \{0,1\}^*$ символом $|S|$ обозначается длина строки S.

Определение 2.5.1. *Задача комбинаторной оптимизации – это тройка* $\mathcal{P} = (\mathrm{Inst}, \mathrm{Sol}, f_I)$, *где*

1) $\mathrm{Inst} \subseteq \{0,1\}^*$ – *множество индивидуальных задач из* $\mathcal{P}$;

2) $\mathrm{Sol}(I) \subseteq \{0,1\}^{n(I)}$ – *множество допустимых решений индивидуальной задачи* $I \in \mathrm{Inst}$, $n(I)$ – *размерность пространства решений;*

3) *для каждой* $I \in \mathrm{Inst}$ *определена функция* $f_I : \mathrm{Sol}(I) \to \mathbb{R}$, *которую требуется максимизировать (если* $\mathcal{P}$ *– задача максимизации) или минимизировать (если* $\mathcal{P}$ *– задача минимизации).*

Далее через f_I^* обозначается оптимальное решение индивидуальной задачи I, т. е. $f_I^* = \max\{f_I(\mathbf{x}) : \mathbf{x} \in \mathrm{Sol}(I)\}$, если $\mathcal{P}$ – задача максимизации, либо $f_I^* = \min\{f_I(\mathbf{x}) : \mathbf{x} \in \mathrm{Sol}(I)\}$, если $\mathcal{P}$ – задача минимизации.

Величина $a > 0$ называется полиномиально ограниченной относительно величины $b > 0$, если существует полином с положительными коэффициентами относительно b, ограничивающий сверху значения a. В тех случаях, когда идет речь о задаче комбинаторной оптимизации, по умолчанию под *полиномиально ограниченной* величиной будем подразумевать величину, полиномиально ограниченную относительно $|I|$. Функции и алгоритмы будем называть эффективно вычислимыми, если время их вычисления полиномиально ограничено.

Наибольший теоретический интерес представляют задачи комбинаторной оптимизации из класса NPO [40], также известные как *задачи* NP *оптимизации*, в которых вводятся следующие «технические» предположения.

Определение 2.5.2. *Задача комбинаторной оптимизации принадлежит классу* NPO, *если отношения* $I \in \mathrm{Inst}$ *и* $\mathbf{x} \in \mathrm{Sol}(I)$ *могут быть проверены за полиномиально ограниченное время, а функция* $f_I : \mathrm{Sol}(I) \to \mathbb{N}$ *полиномиально вычислима для любой* $I \in \mathrm{Inst}$.

Заметим, что если если отношения $\mathbf{x} \in \mathrm{Sol}(I)$ могут быть проверены за полиномиально ограниченное время, то размерность $n(I)$ полиномиально ограничена (иначе невозможно и начать проверку отношения $\mathbf{x} \in \mathrm{Sol}(I)$ без прочтения вектора x).

Класс NPO является аналогом класса NP для задач комбинаторной оптимизации (подробнее см., например, [40]). Функция $n(I)$ представляет собой размерность пространства решений индивидуальной задачи I, записанных в алфавите $\{0, 1\}$.

Если различные решения имеют разную длину записи, то $n(I)$ – наибольшая длина допустимого решения задачи. Далее через f_I^* обозначается оптимальное решение индивидуальной задачи I, т. е. $f_I^* = \max\{f_I(\mathbf{x}) : \mathbf{x} \in \mathrm{Sol}(I)\}$, если $\mathcal{P}$ – задача максимизации, либо $f_I^* = \min\{f_I(\mathbf{x}) : \mathbf{x} \in \mathrm{Sol}(I)\}$, если $\mathcal{P}$ – задача минимизации.

Далее ГА рассматривается в предположении $B = \{0, 1\}^{n(I)}$ и представление решений совпадает с кодировкой решений задачи $\mathcal{P}$, а задача комбинаторной оптимизации имеет критерий «на максимум».

Кроме того, будем предполагать, что при $\mathbf{x} \in \mathrm{Sol}(I)$, функция приспособленности имеет вид $\Phi(\mathbf{x}) = f(\mathbf{x})$. Если же $\mathbf{x} \notin \mathrm{Sol}(I)$, то функции приспособленности можно придать значение меньше, чем на любом допустимом решении, что будет соответствовать штрафу за выход из допустимой области.

2.5.2. Задача поиска локального оптимума

Пусть для всякого элемента $\mathbf{y} \in \mathrm{Sol}(I)$ определена некоторая его окрестность $\mathcal{N}_I(\mathbf{y}) \subseteq \mathrm{Sol}(I)$. Совокупность $\{\mathcal{N}_I(\mathbf{y}) : \mathbf{y} \in \mathrm{Sol}(I)\}$ называется *системой окрестностей*.

Определение 2.5.3. *Если для* $\mathbf{x} \in \mathrm{Sol}(I)$ *при всяком* $\mathbf{y} \in \mathcal{N}_I(\mathbf{x})$ *выполняется неравенство* $f_I(\mathbf{y}) \leq f_I(\mathbf{x})$ *в случае задачи максимизации или* $f_I(\mathbf{y}) \geq f_I(\mathbf{x})$ *в случае задачи минимизации, то решение* $\mathbf{x}$ *называется локальным оптимумом в системе окрестностей* $\mathcal{N}_I$.

Глобальный оптимум, т.е. оптимальное решение задачи комбинаторной оптимизации является частным случаем локального оптимума.

Если $\mathcal{D}(\cdot,\cdot)$ – метрика, заданная для всех элементов $\mathbf{x},\mathbf{y} \in \mathrm{Sol}(I)$, то $\mathcal{N}_I(\mathbf{x}) = \{\mathbf{y} : \mathcal{D}(\mathbf{x},\mathbf{y}) \leq k\}$, $\mathbf{x} \in \mathrm{Sol}(I)$ называется *системой окрестностей радиуса* k, *порожденной метрикой* $\mathcal{D}(\cdot,\cdot)$.

Алгоритм локального поиска начинает свою работу с некоторого допустимого решения. Далее, на каждой итерации алгоритма происходит переход от текущего решения к новому допустимому решению в его окрестности, имеющему лучшее значение целевой функции, чем текущее решение. Процесс продолжается, пока не будет достигнут локальный оптимум. Способ выбора нового решения в окрестности текущего решения зависит от специфики конкретного алгоритма локального поиска.

2.5.3. Достижение локальных оптимумов генетическим алгоритмом

Настоящий раздел посвящен изучению достаточных условий, при которых генетический алгоритм с полной заменой популяции и турнирной селекцией впервые посещает локальный оптимум в среднем за время, близкое к трудоемкости локального поиска (точнее, превышающее ее не более, чем в $\log(n)$ раз). Ограничим рассмотрение задачами безусловной оптимизации вида (1.1).

Мотивацией исследования служит тот факт, что ГА зачастую относят к классу методов локального поиска (см., например, [19]), поэтому представляет интерес детальное изучение случаев, когда работоспособность ГА объясняется сходством его поведения с локальным поиском.

Для простоты обозначений здесь предполагается двоичное представление решений, совпадающее с кодировкой решений задачи комбинаторной оптимизации, а «генотип» – то же, что элемент пространства решений $\{0,1\}^{n(I)}$.

Исследуется ГА с полной заменой популяции и турнирной селекцией. Для удобства анализа будем считать, что условие остановки ГА никогда не выполняется. Далее этот вариант ГА будем обозначать как GA.

Пусть имеется задача комбинаторной оптимизации $\mathcal{P} = (\mathrm{Inst}, \mathrm{Sol}, f_I)$ на максимум, причем $\mathrm{Sol}(I) = \{0,1\}^{n(I)}$. Последнему условию удовлетворяют многие задачи комбинаторной оптимизации, например, задача максимальной выполнимости логической формулы [8], разрез наибольшего веса [8], спиновое стекло в модели Изинга [34] и др.

Пусть выбрана некоторая система окрестностей $\{\mathcal{N}(\xi) \mid \xi \in \mathrm{Sol}(I)\}$.

Обозначим через h число всех неоптимальных значений целевой функции f, т. е. $h = |\{f(\xi) : \xi \in \mathrm{Sol}(I)\}| - 1$. Тогда, начиная с любого допустимого решения, локальный поиск достигает локального оптимума не более чем за h улучшающих целевую функцию итераций. Пусть L обозначает минимальную вероятность достижения решения в пределах окрестности в результате однократной мутации:

$$L = \min_{\xi \in \mathrm{Sol}(I),\ \xi' \in \mathcal{N}(\xi)} \mathbf{P}\{\mathrm{Mut}(\xi) = \xi'\}.$$

Чем выше величина L, тем больше согласованность оператора мутации с системой окрестностей. Будем считать, что численность популяции N, размер турнира s и величина L могут быть выбраны с учетом исходных данных задачи, т.е. являются функциями от I. Для простоты обозначений индивидуальную задачу I здесь и далее, как правило, не будем указывать.

Будем предполагать, что в результате кроссинговера с вероятностью не менее некоторой константы ε, $0 < \varepsilon \leq 1$, образуются особи $(\xi', \eta') = \mathrm{Cross}(\xi, \eta)$, хотя бы одна из которых не уступает по приспособленности родительским особям $\xi, \eta \in B$, т. е.

$$\mathbf{P}\{\max\{\Phi(\xi'), \Phi(\eta')\} \geq \max\{\Phi(\xi), \Phi(\eta)\}\} \geq \varepsilon \tag{2.1}$$

при любых $\xi, \eta \in B$. Под «констаной» в настоящем разделе понимается величина, не зависящая от индивидуальной задачи.

Для одноточечного кроссинговера условие (2.2) выполняется с $\varepsilon = 1 - P_{\mathrm{c}}$, если $P_{\mathrm{c}} < 1$ – константа, не зависящая от задачи. Условие (2.2) выполняется с $\varepsilon = 1$, если один из двух потомков – решение задачи оптимальной рекомбинации родительских решений.

Будем предполагать, что в результате кроссинговера с вероятностью не менее некоторой константы ε, $0 < \varepsilon \leq 1$, образуются особи $(\xi', \eta') = \mathrm{Cross}(\xi, \eta)$, хотя бы одна из которых не уступает по приспособленности родительским особям $\xi, \eta \in B$, т. е.

$$\mathbf{P}\{\max\{\Phi(\xi'), \Phi(\eta')\} \geq \max\{\Phi(\xi), \Phi(\eta)\}\} \geq \varepsilon \tag{2.2}$$

при любых $\xi, \eta \in B$. Под «константой» в настоящем разделе понимается величина, не зависящая от индивидуальной задачи.

Для одноточечного кроссинговера условие (2.2) выполняется с $\varepsilon = 1 - P_{\mathrm{c}}$, если $P_{\mathrm{c}} < 1$ – константа, не зависящая от индивидуальной задачи. Условие (2.2) выполняется с $\varepsilon = 1$, если один из двух потомков – решение задачи оптимальной рекомбинации родительских решений.

Пусть e – число Эйлера, т.е. $e \approx 2,718$.

Лемма 2.5.1. *$e^{-x} \geq 1 - x$ для любого $x \in \mathbb{R}$.*

Действительно,

$$e^{-x} = 1 - x + x^2/2! - ... \geq 1 - x.$$

Лемма 2.5.2. *$e^{-x} \leq 1 - x/e$ при $x \in [0, 1]$.*

Заметим, что $2(1 - 1/e) > 1$, а поэтому при $x \in [0, 1]$ имеем $x \leq 2(1 - 1/e)$, т.е. $1 - x/2 \geq 1/e$ и $x - x^2/2 \geq x/e$. Далее,

$$e^{-x} = 1 - x + x^2/2! - ... \leq 1 - x + x^2/2 \leq 1 - x/e.$$

Лемма 2.5.3. *Для любых событий $A_0, A_1, ..., A_n$*

$$P\{A_0 \& A_1 \& ... \& A_n\} = P\{A_0\} \prod_{i=0}^{n-1} P\{A_{i+1} | A_0 \& A_1 \& ... \& A_i\}.$$

Действительно, по определению условной вероятности,

$$P\{A_0 \& A_1 \& ... \& A_n\} = P\{A_n | A_0 \& A_1 \& ... \& A_{n-1}\} \cdot P\{A_0 \& A_1 \& ... \& A_{n-1}\} = ...$$

$$= P\{A_0\} \prod_{i=0}^{n-1} P\{A_{i+1} | A_0 \& A_1 \& ... \& A_i\}.$$

Теорема 2.5.1. *[11] Если* $\mathrm{Sol}(I) = \{0, 1\}^{n(I)}$, *размер турнира $s \geq r\lambda$, и*

$$\lambda \geq \frac{2(1 + \ln h)}{L\varepsilon(1 - 1/e^{2r})}, \tag{2.3}$$

где $r > 0$, $h > 1$, $L > 0$, то

1. *GA посещает локальный оптимум к итерации h с вероятностью не менее $1/e$, и*
2. *локальный оптимум достигается не позднее, чем за eh итераций GA в среднем.*

Доказательство. Пусть популяция Π^t еще не содержит локального оптимума и событие E_k^{t+1}, $k = 1, \ldots, \lambda/2$, состоит в выполнении следующих трех условий:

1. из популяции Π^t при построении k-той пары потомков для следующего поколения выбирается решение ξ_*^t наибольшей приспособленности;

2. при построении k-той пары потомков посредством кроссинговера, один из них имеет приспособленность не менее $\Phi(\xi_*^t)$ (пусть для определенности это ξ');

3. оператор мутации, примененный к ξ', осуществляет переход в наилучшее по приспособленности решение в окрестности $\mathcal{N}(\xi')$, т. е. $\Phi(\mathrm{Mut}(\xi')) = \max_{\eta\in\mathcal{N}(\xi')}\Phi(\eta)$.

Обозначим через p вероятность наступления хотя бы одного из событий E_k^{t+1}, $k = 1,\ldots,\lambda/2$, при известной популяции Π^t. Найдем оценку $p_{LB} \le p$, не зависящую от выбора Π^t. Согласно схеме GA, $\mathbf{P}\{E_1^{t+1}\} = \ldots = \mathbf{P}\{E_{\lambda/2}^{t+1}\}$. Обозначим эту вероятность через q. Ввиду независимости событий E_k^{t+1}, $k = 1,\ldots,\lambda/2$ при фиксированной Π^t, имеем $p \ge 1-(1-q)^{\lambda/2} \ge 1-e^{-q\lambda/2}$, где последнее неравенство следует из леммы 2.5.1. Оценим снизу вероятность q:

$$q \ge L\varepsilon\left(1-\left(1-\frac{1}{\lambda}\right)^{2s}\right).$$

Однако, $(1-1/\lambda)^{2s} \le (1-1/\lambda)^{2r\lambda} \le 1/e^{2r}$ снова по лемме 2.5.1, поэтому

$$q \ge L\varepsilon\left(1-\frac{1}{e^{2r}}\right) = Lc, \tag{2.4}$$

где $c := \varepsilon\left(1-\frac{1}{e^{2r}}\right)$ для краткости. В дальнейшем мы воспользуемся тем, что из (2.3) и (2.4) вытекает

$$\lambda \ge \frac{2}{L\varepsilon\,(1-1/e^{2r})} \ge 2/q. \tag{2.5}$$

Для оценки снизу вероятности p применим лемму 2.5.2, из которой следует что при любом $z \in [0,1]$

$$1-\frac{z}{e} \ge e^{-z}. \tag{2.6}$$

Положим $z = e^{-q\lambda/2+1}$. Тогда ввиду неравенства (2.5), $z \le 1$, и следовательно,

$$p \ge 1-e^{-q\lambda/2} \ge \exp\left\{-e^{1-q\lambda/2}\right\} \ge \exp\left\{-e^{1-Lc\lambda/2}\right\}. \tag{2.7}$$

От анализа потомков фиксированной популяции Π^t перейдем к случайной последовательности популяций $\Pi^0, \Pi^1, \ldots$. Заметим, что p_{LB}^h является оценкой снизу для вероятности достичь локальный оптимум за серию из не

более h итераций, улучшающих значение рекорда целевой функции. Действительно, пусть $A_t = E_1^{t+1} + \ldots + E_{\lambda/2}^{t+1},\ t = 0, 1, \ldots$. Тогда по лемме 2.5.3,

$$\mathbf{P}\{A_0 \& \ldots \& A_{h-1}\} = \mathbf{P}\{A_0\} \prod_{t=1}^{h-1} \mathbf{P}\{A_t | A_0 \& \ldots \& A_{t-1}\} \geq p_{LB}^h. \tag{2.8}$$

Итак, положим $p_{LB} = \exp\left\{-e^{1-Lc\lambda/2}\right\}$. Снова воспользовавшись условием (2.3), получаем оценку снизу для вероятности либо реализовать серию из h улучшающих рекорд итераций (и значит получить глобальный оптимум), либо получить локальный оптимум еще раньше:

$$p_{LB}^h = \exp\left\{-he^{1-Lc\lambda/2}\right\} \geq \exp\left\{-he^{-\ln h}\right\} = 1/e.$$

Первая часть утверждения теоремы доказана.

Для оценки среднего времени получения локального оптимума рассмотрим последовательность серий по h итераций в каждой. Пусть событием $D_i,\ i = 1, 2, \ldots,$ является отсутствие локального оптимума в популяции GA в i-той серии. При выполнении условий леммы вероятность каждого события $D_i,\ i = 1, 2, \ldots,$ не превышает $\mu = 1 - 1/e$ при любой предыстории работы алгоритма. По аналогии с (2.8) заключаем: $\mathbf{P}\{D_1 \& \ldots \& D_k\} \leq \mu^k$.

Заметим, что если до серии k, локальный оптимум не был найден, то с вероятностью 1 это событие произойдет впоследствии, т.к. вероятность обратного оценивается сверху как $\lim_{t\to\infty} \mu^t = 0$. Таким образом, если через Y обозначить случайную величину, равную номеру первой серии, на которой локальный оптимум будет получен, то, пользуясь свойствами математического ожидания (см., например, [6]), получаем

$$E[Y] = \sum_{i=0}^{\infty} \mathbf{P}\{Y > i\} = 1 + \sum_{i=1}^{\infty} \mathbf{P}\{D_1 \& \ldots \& D_i\} \leq 1 + \sum_{i=1}^{\infty} \mu^i.$$

Следовательно, $E[Y] = e$ и локальный оптимум достигается не позднее, чем за eh поколений GA в среднем. □

Пусть $\lceil \cdot \rceil$ обозначает округление вверх. Тогда в условиях теоремы, при

$$\lambda = 2 \left\lceil \frac{1 + \ln h}{L\varepsilon(1 - 1/e^{2r})} \right\rceil, \quad s = \lceil r\lambda \rceil, \tag{2.9}$$

обеспечено получение локального оптимума в GA за $O(h)$ поколений в среднем.

Упражнение 2.5.1. *Доказать с использованием леммы 2.5.2, что если расстояние Хэмминга между n-битовыми строками $\mathbf{x}$ и $\mathbf{y}$ равно 1, то при использовании оператора мутации из КГА, где $P_\mathrm{m} = 1/n$, вероятность того, что $Mut^*(\mathbf{x}) = \mathbf{y}$, оценивается снизу величиной $1/(e^e n)$.*

2.5.4. Задача ONEMAX

В качестве примера применения теоремы 2.5.1 рассмотрим одноэкстремальную задачу безусловной оптимизации с целевой функцией $ONEMAX(\mathbf{x}) = \sum_{i=1}^{n} x_i$ на множестве $\mathrm{Sol} = X = \{0,1\}^n$. В качестве функции приспособленности естественно выбрать $\Phi(\mathbf{x}) \equiv ONEMAX(\mathbf{x})$. В системе окрестностей, порожденной метрикой Хэмминга радиуса 1, точка $(1,1,...,1)$ является единственным локальным оптимумом, а значит и глобальным.

Пусть в ГА с турнирной селекцией используются операторы мутации и скрещивания из КГА при $P_\mathrm{m} = 1/n$ и при константной вероятности $P_\mathrm{c} < 1$. Как следует из упражнения 2.5.1, для любого $\mathbf{x} \in \mathrm{Sol}$ и любого $\mathbf{y} \in \mathcal{N}(\mathbf{x})$ выполнено $\mathbf{P}\{\mathrm{Mut}^*(\mathbf{x}) = \mathbf{y}\} \geq 1/(e^e n) =: L$. По теореме 2.5.1 заключаем, что при $s \geq r\lambda$, константном $r > 0$, и

$$\lambda = \left\lceil \frac{2e^e n(1+\ln n)}{\varepsilon(1-1/e^{2r})} \right\rceil,$$

ГА впервые посещает оптимум в среднем не позднее, чем за en поколений. Обозначим через T число обращений к функции приспособленности за время работы ГА до первого получения оптимального решения. Величину $E[T]$ в англоязычной литературе принято называть runtime.

Следствие 2.5.1. *Пусть в ГА с турнирной селекцией используется операторы мутации и скрещивания из КГА при $P_\mathrm{m} = 1/n$ и при константной вероятности $P_\mathrm{c} < 1$, тогда при $s \geq r\lambda$, константном $r > 0$, и $\lambda = \left\lceil \frac{2e^e n(1+\ln n)}{\varepsilon(1-1/e^{2r})} \right\rceil$, для функции приспособленности $ONEMAX(\mathbf{x})$ имеем $E[T] = O(n^2 \ln n)$.*

Для сравнения: локальный поиск с окрестностью единичного радиуса Хэмминга получает оптимум в этой задаче после просмотра $O(n^2)$ пробных решений.

Упражнение 2.5.2. *Доказать, что при использовании равномерного кроссинговера с $P_\mathrm{c} = 1$ в случае задачи $ONEMAX$ корректно полагать $\varepsilon = 0.5$.*

Полиномиально ограниченные задачи из класса NPO. Задача комбинаторной оптимизации из класса NPO называется *полиномиально ограниченной,* если существует полином от $|I|$, ограничивающий значения $f_I(\mathbf{x})$, $\mathbf{x} \in \mathrm{Sol}(I)$. Заметим, что процедура турнирной селекции требует времени $O(s) = O(N)$. Следовательно, имеет место

Теорема 2.5.2. *Если $\mathrm{Sol}(I) = \{0,1\}^{n(I)}$ и полиномиально ограничены:*

1. *задача комбинаторной оптимизации $\mathcal{P} = (\mathrm{Inst}, \mathrm{Sol}, f_I)$,*

2. *трудоемкости операторов Mut и Cross,*

3. *а также функция* $1/L(I)$,

то при подходящем выборе параметров GA и функции приспособленности локальный оптимум впервые достигается в среднем за полиномиально ограниченное время.

Если семейство окрестностей $\mathcal{N}(\xi)$ порождено метрикой Хэмминга с константным радиусом окрестности, то существует оператор мутации $\mathrm{Mut}(\xi)$, вычислимый за полиномиально ограниченное время и осуществляющий равновероятный выбор особей-потомков из множества $\mathcal{N}(\xi)$ при заданном ξ. Тогда $1/L$ также ограничена сверху некоторым полиномом от $|I|$. Таким образом, теорема 2.5.2 применима ко многим известным системам окрестностей для задач комбинаторной оптимизации.

В настоящем разделе не учитывался тот факт, что в результате действия кроссинговера приспособленность потомков может оказаться выше приспособленности родителей. Улучшение известных теоретических оценок для ГА за счет такой возможности является более сложной задачей, которую пока удалось решить только для нескольких небольших классов задач дискретной оптимизации. Примеры результатов в этом направлении могут быть найдены в [46, 50].

3. Примеры использования генетических алгоритмов

3.1. Генетический алгоритм построения торговых хабов для рынка электроэнергии

В этом разделе рассматривается задача построения торговых хабов для рынка электроэнергии с маржинальным узловым ценообразованием. Задано число искомых хабов и известны значения цен электроэнергии для каждого узла сети и каждого субъекта рынка за некоторый достаточно продолжительный период. Требуется выбрать кластеры узлов сети (хабы) и прикрепить к ним субъектов рынка с целью минимизации суммарного взвешенного среднеквадратического отклонения узловых цен субъектов от индексов хабов. Ввиду вычислительной сложности задачи, для ее решения предложены эволюционные алгоритмы: генетический алгоритм и гибридная эвристика локального поиска. Обсуждается тестирование и сравнение алгоритмов на реальных данных. Представленные здесь результаты опубликованы в [39].

В современных электроэнергитических рынках, при использовании узлового маржинального ценообразования, цена на электроэнергию не определяется единым значением, но зависит от узла электроэнергетической сети и периода времени. Узел i, $i = 1, \ldots, n$, электроэнергетической сети соответствует точке присоединения генератора, потребителя электроэнергии или сетевой подстанции. В некоторых узлах цена может быть временно не определена в связи с аварийными ситуациями или плановыми ремонтами.

В таких условиях участники рынка заинтересованы в использовании небольшого числа *референтных* цен с целью хеджирования своих ценовых рисков посредством заключения фьючерсных контрактов. Референтные цены могут вычисляться как среднее арифметическое или среднее взвешенное узловых цен по некоторому подмножеству узлов, имеющих «типичную динамику» цен для данного региона. Подмножество узлов $H \subseteq \{1, \ldots, n\}$ с набором весов α_i, $i \in H$, приписанных им, называется *торговым хабом* (или коротко *хабом*). Поведение цен на электроэнергию иллюстрирует рис. 3.1. График содержит данные о ценах в двух узлах и двух хабах рынка «реального времени» системного оператора PJM (США) на 13.02.2007, которые доступны на сайте http://www.pjm.com.

Если в какой-то период времени в некоторых узлах хаба цены не определены, то цена хаба вычисляется как среднее значение цен в остальных узлах (если же таких узлов не остается, то цена хаба в данном интервале считается неопределенной). Подробное описание методов хеджирования ценовых рисков с помощью фьючерсных и форвардных контрактов, заключаемых на цену

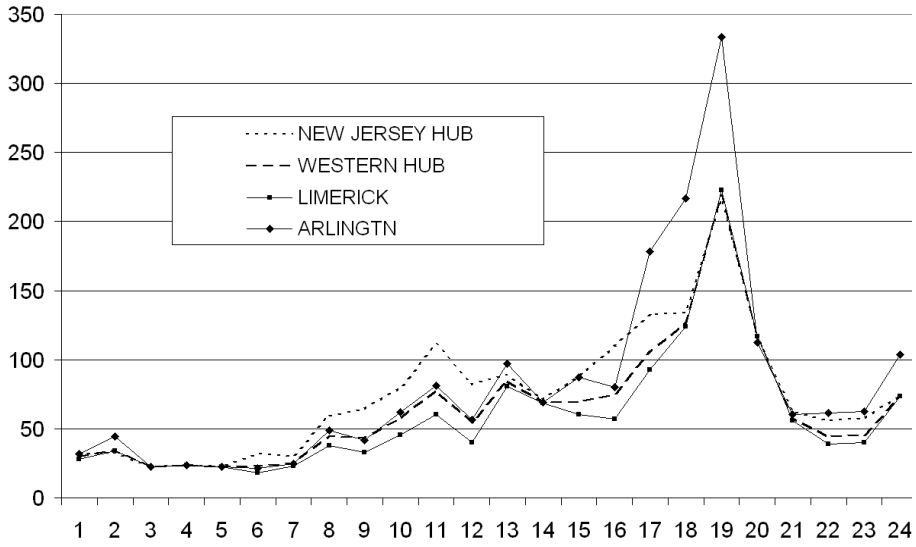


Рис. 3.1. Узловые цены рынка «реального времени» в двух узлах и двух хабах PJM Interconnection на дату 13.02.2007. Согласно графику, хаб Western представляется предпочтительным для хеджирования узла Limerick, в то время как динамика цены в узле Arlingtn сильно отличается от цен обоих хабов и они менее пригодны для хеджирована Arlingtn.

хаба, и некоторые примеры хабов в существующих рынках электроэнергии могут быть найдены, например, в [39].

Без использования математического формализма задачу построения хабов можно представить как задачу отыскания заданного числа хабов, дающих по возможности точную аппроксимацию цен в точках присоединения участников рынка, с учетом некоторых дополнительных ограничений, зависящих от конкретной модификации задачи. Ввиду того, что будущие значения узловых цен не известны, целесообразно предположить, что в будущем характер изменения цен будет аналогичен наблюдаемому в предшествующий достаточно длительный период (год или более).

Одно из ключевых требований к набору торговых хабов для рынка электроэнергии выражается в условии *ликвидности* фьючерсной торговли в этих хабах, т. е. наличии достаточного числа участников рынка, заинтересованных в покупке и продаже фьючерсных контрактов в каждом из хабов. Условие ликвидности включает в себя требование достаточно низкой концентрации рынка в каждом из хабов. Последняя характеристика измеряется индексом Херфиндаля — Хиршмана (см. п. 3.1.2). Перечисленные требования могут быть сформулированы как дополнительные ограничения в задаче построения хабов.

Электроэнергетическая сеть включает в себя тысячи узлов, при этом задача построения хабов должна решаться на основе анализа узловых цен, определенных для нескольких тысяч временных интервалов, что приводит

к необходимости решения сложной задачи оптимизации с большой размерностью. Ввиду вычислительной сложности данной задачи, вместо отыскания точного ее решения, предлагается два эволюционных метода поиска приближенных решений: генетический алгоритм и гибридная эвристика локального поиска, в которой комбинируется локальный поиск с эволюционной стратегией (1+1)-EA.

3.1.1. Постановка задачи

Будем для простоты считать, что каждый участник рынка владеет одним узлом сети и участвует в торгах на электроэнергитическом рынкеот лица либо отдельного предприятия-потребителя, либо генератора, либо населенного пункта. Всякий контракт участника в торгах относится к его узлу сети. Поэтому далее множество всех участников рынка совпадает со множеством узлов сети. Для всякого узла i, $i = 1, \ldots, n$, известен вектор цен $(c_{i1}, \ldots, c_{iT})$, содержащий цену электроэнергии для данного узла за весь исследуемый период. Для удобства далее временные интервалы $1, \ldots, T$ будем называть *часами* (как правило, на практике длина интервала постоянства цены на электроэнергию равна 1 часу).

Требуемое число хабов m задано (как правило, m не велико, в пределах 10). Для учета различий в уровне значимости разных участников (и соответствующих узлов сети) вводятся весовые коэффициенты w_i, $i = 1, \ldots, n$. Будем предполагать что веса равны средним значениям объемов торговли электроэнергией в узлах сети за весь анализируемый период.

Будем полагать, что мера близости между ценой C_{jt} хаба j, $j = 1, \ldots, m$, и ценой c_{it} имеет вид

$$d(i, j) = \sum_{t=1}^{T} (c_{it} - C_{jt})^2 w_i.$$

Естественным образом, в оптимальном решении каждый узел i должен быть прикреплен к хабу, дающему минимальное значение $d(i, j) = \sum_{t=1}^{T} (c_{it} - C_{jt})^2 w_i$ среди всех m имеющихся хабов.

3.1.2. Фрмулировка задачи с учетом надежности и ликвидности

Рассмотрим некоторые обусловленные практикой ограничения, которые могут учитываться при построении хабов.

Размер хаба. Узловые цены определяются только для узлов, которые не отключены от электроэнергетической сети. Однако во многих узлах периодически требуется проводить регламентные работы, либо происходят отключения, связанные с авариями. В таких случаях увеличение числа узлов в

хабе H_j приводит к более устойчивой и предсказуемой цене хаба C_{jt} и меньшей вероятности того, что C_{jt} окажется не определена в некоторый час. Простое предположение о том, что все узлы системы имеют равные вероятности отключения приводит к необходимости задания минимального допустимого числа узлов $n_{\min}$ в хабах. На практике хабы, как правило, состоят из десятков или сотен узлов (см., например, [70]).

Если цена хаба вычисляется как взвешенное среднее по всем его узлам, то отключение узла с высоким значением веса может оказать большое влияние на цену хаба. В связи с этим, определение цены хаба как среднего арифметического цены в его узлах представляется предпочтительным по сравнению со средневзвешенной ценой хаба. Кроме того, среднее арифметическое значение вычисляется более просто и понятнее для участников рынка.

Уровень конкуренции в хабе. Важным показателем уровня конкуренции является концентрация рынка, измеряемая индексом Херфиндаля — Хиршмана (HHI), который рассчитывается как сумма квадратов процентной доли предложения от каждого из участников. Высокие значения данного индекса показывают, что рынок делят между собой всего несколько участников. При значении HHI, превышающем 1800 рынок принято считать чрезмерно концентрированным [71].

При заданном наборе хабов, будем говорить что узел i принадлежит к *области предпочтения* $G_{j'} \subseteq G$ хаба j', если $d(i, j')$ минимально среди всех j, $j = 1, \ldots, m$. Далее будем предполагать, что $G_1, \ldots, G_m$ образуют разбиение множества $\{1, \ldots, n\}$.

Индексы HHI определяются как для генерации, так и для потребления следующим образом:

$$I_j^{\text{gen}} = \sum_{i=1}^{n} \left(\frac{100 \cdot \sum_{i \in G_j} W_i^{\text{gen}}}{\sum_{i \in G_j} W_i^{\text{gen}}} \right)^2, \ j = 1, \ldots, m, \tag{3.1}$$

$$I_j^{\text{con}} = \sum_{i=1}^{n} \left(\frac{100 \cdot \sum_{i \in G_j} W_i^{\text{con}}}{\sum_{i \in G_j} W_i^{\text{con}}} \right)^2, \ j = 1, \ldots, m, \tag{3.2}$$

где W_i^{gen}, W_i^{con} – объемы генерации и потребления за исторический период. Согласно рекомендациям практиков выбирается граница $I_{\max} = 1800$.

Постановка задачи. В дальнейшем всюду предполагается, что цена хаба C_{jt} вычисляется как среднее арифметическое узловых цен, т. е. полностью определяется множеством узлов хаба H_j и узловыми ценами c_{it}, $i \in H_j$. При заданном наборе хабов $H_1, \ldots, H_m$ критерий оптимизации может быть записан как

$$f(H_1,\ldots,H_m)=\sum_{i=1}^{n}\left(w_i\cdot\min_{j=1,\ldots,m}\sum_{t=1}^{T}\left(c_{it}-\frac{\sum_{i\in H_j}c_{it}}{|H_j|}\right)^2\right).$$

3.1.3. Генетический алгоритм

Рассматриваемый далее ГА основан на стационарной схеме воспроизведения [93]:

Генетический алгоритм со стационарной схемой воспроизведения

1. Построить начальную популяцию П случайным образом. Полагаем $\theta := 1$.
2. Пока не выполнены условия остановки, повторять:
   2.1 Выбрать родительские особи $\mathbf{x}, \mathbf{y}$ с помощью s-турнирной селекции из П.
   2.2 Построить потомка $\mathbf{z}$ применением кроссинговера: $\mathbf{z} := \mathrm{Cross}(\mathbf{x}, \mathbf{y})$.
   2.3 Выполнить мутацию: $\mathbf{z}' := \mathrm{Mut}(\mathbf{z})$.
   2.4 Выбрать особь $\mathbf{z}$ с наименьшим значение целевой фцнкции f из П и заменить ее на $\mathbf{z}'$.
   2.5 Перейти на следующую итерацию $\theta := \theta + 1$.
3. Вернуть лучшее найденное решение.

Для обозначения генотипов здесь используются символы $\mathbf{x}, \mathbf{y}$ или $\mathbf{z}$. Набор хабов $(H_1^{\mathbf{x}},\ldots,H_m^{\mathbf{x}})$ кодируется генотипом $(x_1,\ldots,x_n)$, где $x_i, i = 1,\ldots,n$ — номер хаба, к которому прикреплен узел i, т.е. $x_i = j$ тогда и только тогда, когда $i \in H_j^{\mathbf{x}}$. Некоторые узлы могут не принадлежать никакому хабу, в этом случае полагается $x_i = 0$.

Если особь $\mathbf{x}$ соответствует допустимому решению, то ее приспособленность $F(\mathbf{x})$ полагается равной значению целевой функции $f(\mathbf{x})$. В противном случае, используется дополнительный штраф за недопустимость. Для этого определим компоненты $H_j^{\mathbf{x}}$ для каждого хаба следующим образом:

- Пусть $d_j = \max\{0, n_{\min} - |H_j^{\mathbf{x}}|\}$.
- Чрезмерная концентрация рынка для генераторов измеряется с использованием индекса HHI как $D_j^{\mathrm{gen}} = \max\{0, I_j^{\mathrm{gen}} - I_{\max}\}$, где определение I_j^{gen} совпадает с (3.1).
- Чрезмерная концентрация рынка для потребителей измеряется с использованием индекса HHI как $D_j^{\mathrm{con}} = \max\{0, I_j^{\mathrm{con}} - I_{\max}\}$, где определение I_j^{con} совпадает с (3.2).

Тогда приспособленность произвольного решения (возможно, недопустимого) вычисляется как

$$F(\mathbf{x}) = f(\mathbf{x}) +$$

$$C \cdot \sum_{j=1}^{m} \left(\operatorname{sgn}(d_j) + \operatorname{sgn}(D_j^{\text{gen}}) + \operatorname{sgn}(D_j^{\text{con}}) + \frac{d_j}{n_{\max}} + \frac{D_j^{\text{gen}} + D_j^{\text{con}}}{D_{\max}} \right),$$

где C — максимальная узловая цена в системе за весь рассматриваемый период, т.е. $C = \max\{c_{it} : t = 1, \ldots, T,\ i = 1, \ldots, n\}$, $D_{\max} = 10^4 - I_{\max}$ — максимально возможное значение D_j^{gen} и D_j^{con}. Функция $\operatorname{sgn}(x)$ равна 1 для положительных чисел, -1 для отрицательных и 0 для $x = 0$. Легко видеть, что для любого недопустимого решения функция приспособленности $F(\cdot)$ принимает значение большее, чем для любого допустимого.

В решениях, составляющих начальную популяцию, хаб для каждого узла назначается случайным образом с равномерным распределением. На шаге 2.4 для удаления выбирается особь с наихудшим (максимальным) значением функции приспособленности. Наилучшим по итогам работы ГА считается решение с минимальным значением функции приспособленности среди всех построенных.

Можно заметить, что предложенная схема допускает неоднозначное описание одного и того же решения (для этого достаточно перенумеровать хабы), поэтому перед применением кроссинговера осуществляется *выравнивание*: изменение номеров хабов у одной из родительских особей таким образом, чтобы хабы (как множества узлов) обоих родителей с одинаковыми номерами имели наименьшую симметрическую разность. Более формально, требуется найти перестановку $\{\pi_1, \ldots, \pi_m\}$, минимизирующую функцию $\sum_{j=1}^{m} H_j^{\mathbf{x}} \,\Delta\, H_{\pi_j}^{\mathbf{y}}$, где $H_j^{\mathbf{x}}$ и $H_k^{\mathbf{y}}$ — множества узлов хабов j и k в родительских решениях $\mathbf{x}$ и $\mathbf{y}$, соответственно. В силу малого числа хабов, возникающая задача о назначениях решается полным перебором возможных вариантов.

В генетическом алгоритме используется стандартный равномерный кроссинговер: номер хаба для каждого узла выбирается случайным образом среди номеров хабов для этого узла в родительских решениях после процедуры выравнивания. Полученное решение подвергается мутации, в ходе которой каждый узел с вероятностью p_{mut} переносится в другой хаб, выбранный случайно с равномерным распределением.

Сходимость ГА.

Утверждение 3.1.1. *Пусть* $0 < p_{\text{mut}} < 1$ *и условие остановки никогда не выполняется, тогда описанный выше ГА сходится к оптимуму п. н.*

Параллельная реализация. Возможна параллельная реализация ГА. Заданное число потоков (по одному на каждое ядро процессора) независимо выполняют шаги 2.1 – 2.5 алгоритма. Еще один поток, управляющий, отвечает за изменение параметров алгоритма, генерирует заново при необходимости

подмножество часов, используемых в ГА, и прекращает работу алгоритма по истечении заданного времени.

В случае, если оказывается что результат турнирной селекции в одном из потоков был бы другим из-за обновления одной из особей на шаге 2.4 в параллельном потоке с меньшим номером, потомок $\mathbf{z}$, вычисленный в данном потоке, отбрасывается. Потери производительности, связанные с коммуникацией и синхронизацией потоков, становятся заметными лишь при большом числе потоков.

3.1.4. Алгоритм локального поиска (LS)

LS – это вариант известного эвристического алгоритма K-средних (см., напр., [73, 77]), адаптированного здесь для задачи построения хабов.

Локальный поиск начинает работу с начального множества хабов $H_1, \ldots, H_m$ и на каждой итерации перемещает узел из одного хаба в другой с целью уменьшить значение целевой функции $F(\cdot)$. На каждой итерации просматриваются все узлы и выполняется проверка, будет ли улучшение целевой функции при перемещении текущего узла в другой хаб. Из всех таких перемещений выбирается и применяется наилучшее. Итерации продолжаются до тех пор, пока не будет найден локальный оптимум, или пока количество просмотренных решений не достигнет заданного предела. Алгоритм может перезапускаться несколько раз с различными случайно выбранными начальными решениями.

3.2. Вычислительный эксперимент

Генетический алгоритм и алгоритмы LS, (1+1)-EA и HLS были запрограммированы и тестировались на 8-ядерной ЭВМ с процессорами Xeon E5420 QuadCore с тактовой частотой 2.5 ГГц. Для сравнения алгоритмов использовались реальные исходные данные рынка на сутки вперед Единой Энергетической Системы России с 200 участниками рынка. В структуре данных выделяются европейская зона, содержащая 1425 групп точек поставки, и сибирская зона, содержащая 216 групп точек поставки. Временной интервал составляет 1 час и цены рассчитаны в более чем 6000 узлов в европейской части и в более чем 500 узлах в Сибири. В ходе предварительного анализа узлов, которые могут быть включены в хабы, количество узлов было сокращено до 1161 в европейской зоне и 325 в сибирской зоне. Общий период наблюдения составил три года, начиная с 2007 г.

Были сформированы двенадцать задач с различными зонами, периодами и количеством хабов: с полным 3-годичным периодом в каждой зоне (задачи E′ и S′) и включая только 1-годичный период соответствующий годам 2008 или 2009 (задачи E8, E9, S8 и S9). Требования к хабам для европейской зоны: $n_{\min} = 100$, $m = 3$ или 4, для Сибири: $n_{\min} = 80$, $m = 2$ или 3.

Таблица 3.1. Значения целевой функции решений, найденных алгоритмами (1+1)-EA, LS, ГА и HLS для европейской зоны

	E′	E8	E9	E′	E8	E9
m	3	3	3	4	4	4
(1+1)-EA	62.99	58.33	62.93	60.74	55.21	60.06
LS	57.88	54.17	55.51	54.66	52.02	**52.8**
ГА	58.04	54.14	55.45	56.36	52.17	53.21
HLS	**57.78**	**54.09**	**51.45**	**54.51**	**51.23**	55.29

Для более полного использования возможностей многопроцессорной техники нами были реализованы параллельные версии алгоритмов LS и (1+1)-EA. Для каждой задачи осуществлялось 8 независимых запусков алгоритма (по одному на каждое ядро ЦПУ) со случайных начальных решений. Все потоки имели одинаковое ограничение по времени, указанное ниже. Для алгоритма LS в пределах одного запуска использовалась стратегия мультистарта: при достижении локального оптимума, если общий предел времени не был превышен, то случайным образом порождалось новое начальное решение, и алгоритм запускался заново. Для алгоритма (1+1)-EA перезапуск не применялся, его итерации просто продолжались в пределах заданного времени.

Генетический алгоритм также был запрограммирован с учетом наличия нескольких процессоров. Предварительные эксперименты показали, что качество полученного решения может существенно колебаться в зависимости от выбранной начальной популяции, поэтому для ГА также было решено применять мультистарт. В окончательной версии алгоритма производится 4 независимых запуска ГА, каждый из которых использует 2 процессорных ядра для выполнения требуемых операций. После истечения одной трети времени, отведенного на расчеты, остаются только две популяции из четырех, в которых были найдены лучшие рекордные значения, а для обновления каждой из них используется по 4 потока. В экспериментах нами использовались следующие параметры генетического алгоритма: $\lambda = 100, s = 5, p_{mut} = 2/n$. При тестировании алгоритма HLS было сделано восемь независимых запусков по одному на каждое ядро процессора.

Для всех трех алгоритмов был установлен одинаковый предел времени работы: 4 часа для задачи E′; 2 часа для S′; 1 час для E8, E9 и 30 минут для S8 и S9. Значения целевой функции найденных решений приведены в таблицах 3.1 и 3.2. Лучшие значения выделены жирным шрифтом. Для задачи S8 алгоритмами LS и (1+1)-EA допустимые решения не были найдены ни в одном из запусков, это отражено в таблице знаком “-”.

В ходе эксперимента было замечено, что алгоритм LS не достигает локального оптимума в пределах выделенного времени. Для самой большой задачи E′ алгоритмом LS сделано только 13 итераций ($13n$ вычислений целевой

Таблица 3.2. Значения целевой функции решений, найденных алгоритмами (1+1)-EA, LS, ГА и HLS для сибирской зоны

	S′	S8	S9	S′	S8	S9
m	2	2	2	3	3	3
(1+1)-EA	45.61	47.61	53.64	50.07	-	56.52
LS	44.36	47.16	52.07	50.57	-	56.49
ГА	44.47	**47.04**	**51.94**	**47.12**	**47.65**	**53.45**
HLS	**44.19**	47.10	51.96	47.52	48.49	54.84

функции) при $m = 3$ и 9 итераций при $m = 4$. Для задач E8 и E9 количество итераций составило 25 при $m = 3$ и 15 при $m = 4$. Локальный оптимум для задач европейской зоны был достигнут только для задачи E9 при $m = 4$. Напротив, для задач сибирской зоны, алгоритм LS находил локальный оптимум достаточно часто. Для задачи S′ около 50 локальных оптимумов было найдено при $m = 2$ и около 40 при $m = 3$ на каждом процессорном ядре. Для задач S8 и S9 на каждом процессорном ядре было получено около 30 локальных оптимумов как при $m = 2$ так и при $m = 3$.

Количество итераций, сделанных алгоритмом (1+1)-EA для задач европейской зоны, было достаточно велико: около 25000 для задачи E′ и 50000 для E8 и E9. Для задач сибирской зоны количество итераций было еще больше: 60000 для S′ и около 90000 для S8 и S9. Заметим, что во многих случаях улучшения происходили даже на последних итерациях, хотя они были очень незначительными. В большинстве случаев улучшения целевой функции происходили в течение всего времени работы ГА и HLS.

Поиск оптимального решения задачи построения хабов представляет значительные трудности для алгоритмов локального поиска с многократным перезапуском даже для задач небольшой размерности. Алгоритмы HLS и ГА показали лучшие результаты благодаря их способности выходить из локальных оптимумов.

Алгоритм HLS продемонстрировал преимущества на больших, наиболее трудных задачах, в то время как ГА оказался лучшим на задачах меньшей размерности. Это может быть объяснено достаточно медленной сходимостью ГА на задачах с «глубокими» локальными оптимумами. Заметим, что хотя по утверждению 3.1.1, генетический алгоритм с вероятностью 1 находит оптимальное решение за конечное число итераций, на практике время работы может быть недопустимо велико. Задачи европейской зоны сравнительно легки для HLS за счет комбинирования двух типов локальных изменений решения и выделения большего времени процессам поиска с лучшими текущими решениями. С другой стороны, для задач сибирской зоны важно не столько быстрое достижение локального оптимума, сколько удачное определение области поиска с наиболее перспективными решениями.



Рис. 3.2. Коэффициент корреляции между средними областными ценами европейской зоны и ценами в ближайшем из хабов: “Юг”, “Центр” и “Урал” (слева направо).

Практическое использование. Предложенные алгоритмы были использованы[1] для построения набора хабов “Центр”, “Юг”, “Урал ”, “Западная Сибирь” и “Восточная Сибирь”, действующих в настоящее время на рынке Единой энергосистемы России. Рис. 3.2 и 3.3 показывают величину коэффициента корреляции между ценами в этих хабах и средневзвешенными ценами на электроэнергию в областях, активно участвующих в рынке электроэнергии. Средневзвешенные цены для областей на каждый час вычислялись как среднее значение узловых цен по данной области с весами, равными абсолютному значению мощности, потребляемой или генерируемой в узле. Коэффициент корреляции показан интенсивностью штриховки. Для каждой области выбор ближайшего хаба изображен наклоном штриховки.

Ценовые индексы данных хабов регулярно публикуются на сайте ОАО «Администратор торговой системы оптового рынка электроэнергии» (https://www.atsenergo.ru/results/rsv/hubs/hubs) и используются при заключении фьючерсных контрактов на Московской энергетической бирже.

[1]При этом описанная выше мера расстояния между ценами хаба и субъекта рынка была заменена на расстояние, основанное на линейной регрессии.



Рис. 3.3. Коэффициент корреляции между средними областными ценами сибирской зоны и ценами в ближайшем из хабов: “Западная Сибирь”, ‘и “Восточная Сибирь” (слева направо).

4. Эволюционные алгоритмы

4.1. Общий вид опраторов

На каждой итерации эволюционного алгоритма (ЭА) имеется текщая популяция особей, переходящей с одного поколения на следующее. Далее при описании эволюционных алгоритмов для обозначения численности такой популяции будет использоваться симовол N. Кроме того, дальнейшем нам потребуется символ, обозначающий множество всех генотипов популяции Π, т.е. *генофонд* популяции. Запишем его как $\widehat{\Pi} = \cup_{i=1}^{N}\{\xi^i\}$. В общем случае работа ЭА может быть описана с помощью операторов, представляющих собой следующие рандомизированные процедуры, т.е. программы для вероятностной машины Тьюринга – см., например, [18], гл. 3.

- Начальная популяция $\Pi^0 = Init$ строится случайным образом с помощью рандомизированной процедуры $Init$.
- Оператором селекции $Select : B^N \to B^{N'}$ извлекается N' копий генотипов родителей из текущей популяции, которые помещаются в промежуточную популяцию $\Pi' = Select\,(\Pi^t)$, причем $\widehat{\Pi'} \subseteq \widehat{\Pi^t}$.
- Действием оператора вариации $Variate : B^{N'} \to B^{N''}$ вносятся некоторые случайные изменения в генотипы, полученные от родительских особей. Таким образом создаются N'' генотипов-потомков, составляющих популяцию Π''. (В частности, в случае КГА $N'' = N' = N$, и действие данного оператора состоит в последовательном применении скрещивания и мутации.)
- С помощью оператора выживания $Survive : B^N \times B^{N''} \to B^N$ определяются генотипы из популяции Π^t и их потомки из Π'', которые будут добавляться в очередную популяцию Π^{t+1}, т.е. $\widehat{Survive}\,(\Pi^t, \Pi'') \subseteq \widehat{\Pi^t} \cup \widehat{\Pi''}$.

Работа ЭА начинается со случайной начальной популяции $\Pi^0 = Init$ и продолжается итерациями случайного отображения

$$\Pi^{t+1} = Survive\left(\Pi^t, Variate\left(Select\left(\Pi^t\right)\right)\right),$$

пока не будет выполнено некоторое условие остановки. Работа заканчивается выводом в качестве ответа лучшего найденного решения $x(\tilde{\xi})$, где

$$\tilde{\xi}_t = \arg\max\{f(x(\xi^{i,\tau})) : \tau = 0, ...t,\ i = 1, ..., N\}.$$

Условие остановки может быть ограничением по общему числу итераций, либо по числу итераций без улучшения рекорда целевой функции $f(x(\tilde{\xi}_t))$. В некоторых задачах можно заранее определить требуемое значение целевой функции, по достижению которого алгоритм останавливается. Как правило, в дальнейшем при теоретическом исследовании алгоритмов для удобства будем полагать, что условие остановки никогда не выполняется.[1] Очевидно, всякий представитель класса эволюционных алгоритмов может быть реализован на вероятностной машине Тьюринга.

Распределение вероятностей на выходе процедур $Select$, $Variate$, $Survive$ должно полностью определяться входными данными решаемой задачи (которую будем считать фиксированной) и одной или двумя популяциями-аргументами, поданными на вход процедуры. Таким образом, имеют место *марковские свойства* указанных операторов, которые могут быть формализованы следующим образом.

Пусть $\mathbf{M}$ – вероятностная машина Тьюринга, реализующая данный ЭА, и Θ обозначает последовательность состояний, которые проходит машина $\mathbf{M}$ до применения рассматриваемого оператора $Select$, $Variate$ или $Survive$.

Условимся обозначать детерминированные популяции или реализации случайных популяций буквой π, а популяции, являющиеся случайными величинами – как и прежде, заглавными буквами Π (например, для реализации Π^t будем использовать π^t). Аналогично поступим с другими случайными величинами и их реализациями (ξ, Ξ и т.д.). Тогда марковские свойства операторов ЭА формулируются следующим образом.

- Для любых $\pi' \in B^{N'}$, $\pi \in B^N$, $t \geq 0$ и любой последовательности состояний θ до момента применения данного оператора выполнено равенство

$$P\left\{\pi' = Select(\pi)\right\} = P\left\{\pi' = Select(\pi) | \Theta = \theta\right\}. \tag{4.1}$$

- Для любых $\pi'' \in B^{N''}$, $\pi' \in B^{N'}$, $t \geq 0$ и последовательности θ до момента применения данного оператора, на шаге t выполнено равенство

$$P\left\{\pi'' = Variate(\pi')\right\} = \tag{4.2}$$
$$P\left\{\pi'' = Variate(\pi') |\ \Theta = \theta\right\}.$$

- Для любых $\pi \in B^N$, $\pi'' \in B^{N''}$, $\pi^{t+1} \in B^N$, $t \geq 0$ и последовательности состояний θ до момента применения данного оператора на шаге t, выполнено равенство

$$P\left\{\pi^{t+1} = Survive(\pi, \pi'')\right\} = \tag{4.3}$$
$$P\left\{\pi^{t+1} = Survive(\pi, \pi'') | \Theta = \theta\right\}.$$

[1] Однако, практика показывает, что многократный независимый перезапуск алгоритма зачастую позволяет значительно улучшить результаты при том же общем времени вычислений.

Очевидно, КГА и все рассмотренные ранее его модификации соответствуют приведенной общей схеме ЭА, при этом $N = \lambda$, а соответствующие операторы селекции и вариации (мутация и кроссинговер) обладают марковскими свойствами.

4.2. Эволюционные стратегии (μ, λ)-ES и $(\mu + \lambda)$-ES

Один из первых вариантов эволюционной стратегии (1+1)-ES был предложен Л.А. Растригиным [22], гл. 2, где этот алгоритм был назван *локальным поиском с пересчетом при неудачном шаге*. И. Реченберг [87] сформулировал более общие вычислительные схемы эволюционных стратегий, которые и приводятся ниже.

Прежде, чем дать описание алгоритмов, определим один вспомогательный оператор. Пусть оператор s_μ из данной на вход популяции генотипов (численностью не менее μ) выбирает без повторений μ особей с наибольшей приспособленностью и возвращает популяцию из них в качестве результата. Заметим, что если численность входной популяции равна λ, то данный оператор реализует уже рассмотренную ранее (μ, λ)-селекцию.

Общая схема алгоритмов (μ, λ)-ES и $(\mu + \lambda)$-ES

1. Построить $\Pi^{(0)} = (\xi^{1,0}, \ldots, \xi^{\mu,0}) := Init$, положить $t := 0$.
2. Пока не выполнен критерий остановки, выполнять:
   2.1 Для $i := 1$ до λ выполнять 2.1.1, 2.1.2:
      2.1.1 Выбрать u_i с равномерным распределением из $\{1, 2, \ldots, \mu\}$.
      2.1.2 Положить $\eta^i := Mut(\xi^{u_i,t})$.
   2.2 Положить $\Pi^{t+1} := \begin{cases} s_\mu(\eta^1, \ldots, \eta^\lambda) \text{ в алгоритме } (\mu, \lambda)\text{-ES} \\ s_\mu(\xi^{1,t}, \ldots, \xi^{\mu,t}, \eta^1, \ldots, \eta^\lambda) \text{ в алгоритме } (\mu + \lambda)\text{-ES.} \end{cases}$
   2.3 $t := t + 1$.
3. Результат – наиболее приспособленный из найденных генотипов $\tilde{\xi}_{t-1}$.

Соответствие между операторами ЭА и ES (здесь $N = \mu$):

- *Select* – равновероятно выборать λ особей из Π^t с повторением (таким образом вычисляется популяция Π', $N' = \lambda$);
- *Variate* – применение λ раз оператора Mut (таким образом вычисляется популяция Π'', $N'' = \lambda$);
- *Survive* – выбор μ лучших особей из Π' (в случае (μ, λ)-ES) или выбор μ лучших особей из Π' и Π^t (в случае $(\mu + \lambda)$-ES).

Эволюционные стратегии изначально были предложены для задач непрерывной оптимизации, где $X \subseteq \mathbb{R}^n$ имеет мощность континуума. В таких

задачах часто используются операторы мутации Mut_σ с нормально распределенным случайным шагом:

$$Mut_\sigma(\xi) = x^{-1}(x(\xi) + Z),$$

где $Z = (Z_1, ..., Z_n)$ и Z_i, $i = 1, ..., n$ – независимые нормально распределенные случайные величины со стандартным отклонением σ (это настраиваемый параметр алгоритма) и нулевым математическим ожиданием.

При решении задач дискретной оптимизации, как правило, используется оператор мутации из КГА. В таком случае алгоритмы обозначаются как (μ, λ)-EA, $(\mu + \lambda)$-EA, соответственно.

4.3. Сходимость эволюционных алгоритмов

Введем обозначение для целевой функции от фенотипа лучшей особи на поколении t:

$$F_t = \max\{f(x(\Xi^{1,t})), f(x(\Xi^{2,t})), ..., f(x(\Xi^{N,t}))\}.$$

Определение 4.3.1. *Будем говорить, что популяция ЭА сходится к оптимуму в задаче (1) почти наверное, если $F_t \to f^*$ почти наверное (п.н.) при $t \to \infty$.*

Для задачи (1.3) сходимость популяции ЭА к оптимуму п.н. определяется аналогично.

Пусть B^* обозначает множество оптимальных генотипов. По аналогии с [95], сформулируем несколько предположений относительно операторов ЭА, которые позволят в дальнейшем доказать сходимость к оптимуму для популяции ЭА.

Определение 4.3.2.

$\mathcal{A}_1$. *Для оператора Select существует такое $\epsilon_1 > 0$, что при любых $\pi^t \in B^N, \xi \in \widehat{\pi}^t$, $t \geq 0$*

$$P\{\xi \in \widehat{Select}(\pi^t)\} \geq \epsilon_1.$$

$\mathcal{A}_2$. *Для оператора вариации Variate существует $\epsilon_2 > 0$, такое что для любых $\xi \in B$, $t \geq 0$ найдется последовательность генотипов $\eta^0, \eta^1, ..., \eta^{k(\xi)}$, где $\eta^0 = \xi$, $\eta^{k(\xi)} \in B^*$ и при всех $i = 0, ..., k-1$ имеем*

$$P\{\eta^{i+1} \in \widehat{Variate}(\pi')\} \geq \epsilon_2 \ \ \forall\ \pi' : \eta^i \in \widehat{\pi}'.$$

$\mathcal{A}_3$. *Для оператора выживания Survive существует $\epsilon_3 > 0$, такое что при любом $t \geq 0$*

$$P\{\xi \in \widehat{Survive}(\pi^t, \pi'')\} \geq \epsilon_3 \ \ \forall\ \pi^t \in B^N, \pi'' \in B^{N''} : \xi \in \widehat{\pi''}.$$

$\mathcal{A}_4$. *Для оператора выживания Survive при любых* $\pi^t, \pi'', t \geq 0$ *выполняется*

$$\max\{f(x(\xi)) : \xi \in \widehat{Survive}(\pi^t, \pi'')\} \geq \max\{f(x(\xi)) : \xi \in \widehat{\pi^t} \cup \widehat{\pi''}\}.$$

Заметим, что ввиду марковских свойств операторов ЭА данное определение корректно, т.к. используемые в нем вероятности не зависят от предыстории работы алгоритма до вызова указанного оператора, если известны все аргументы на входе оператора.

Утверждение 4.3.1. *(об условных вероятностях) Для любых событий* A, A', A''

$$P\{A \ \& \ A'|A''\} = P\{A|A' \ \& \ A''\}P\{A'|A''\},$$

если эти условные вероятности определены.

Доказывается трехкратным применением формулы из определения условной вероятности.

Утверждение 4.3.2. *(формула полной условной вероятности) Для любых событий* A, A' *и альтернатив* $A_1, ..., A_k$ *таких, что* $A_i \cap A_j = \emptyset, i \neq j$ *и* $A' = \cup_{i=1}^k A_i$, *выполнено равенство*

$$P\{A|A'\} = \sum_{i=1}^{k} P\{A|A_i\}P\{A_i|A'\},$$

если эти условные вероятности определены.

Доказательство. Условная вероятность $P\{A|A'\} = P\{A \ \& \ A'\}/P\{A'\}$ по формуле вероятности для суммы взаимоисключающих событий представима в виде

$$P\{A|A'\} = \frac{P\{A \ \& \ A'\}}{P\{A'\}} = \sum_{i=1}^{k} \frac{P\{A \ \& \ A' \ \& \ A_i\}}{P\{A'\}} = \sum_{i=1}^{k} \frac{P\{A|A' \ \& \ A_i\}P\{A' \ \& \ A_i\}}{P\{A'\}}.$$

Пользуясь определением условной вероятности и тем, что $A' \ \& \ A_i = A_i$ для всех $i = 1, .., k$, приходим к требуемому равенству. □

Следующая теорема о непрерывности вероятностной меры известна из курса теории вероятностей (см., например, [5], гл. 2, §1).

Теорема 4.3.1. *Пусть* $\{A_n\}$ – *последовательность множеств из сигма-алгебры событий,* $A_{n+1} \subset A_n \ \forall n$, *тогда*

$$\lim_{n\to\infty} P(A_n) = P(\cap_{n=1}^{\infty} A_n).$$

При кодировке решений задачи (1.1) оптимум может не быть представлен в пространстве генотипов. В следующей теореме такая ситуация исключается предположением о том, что $B^* \neq \emptyset$.

Теорема 4.3.2. *О сходимости ЭА (Айбен, Аартс, ван Хи, 1989) [53, 95]. Пусть $B^* \neq \emptyset$ и условие остановки никогда не выполняется. Тогда*

1. В случае выполнения условий $\mathcal{A}_1 - \mathcal{A}_3$ имеем

$$P\{\exists t : \widehat{\Pi^t} \cap B^* \neq \emptyset\} = 1, \tag{4.4}$$

т.е. в ЭА оптимальный генотип порождается с вероятностью единица за конечное число итераций.

2. В случае выполнения условий $\mathcal{A}_1 - \mathcal{A}_4$ популяция ЭА сходится к оптимуму почти наверное.

Доказательство.

1. Рассмотрим популяцию Π^0 как случайную величину. Как будет видно в дальнейшем, вместо Π^0 можно было бы взять популяцию Π^t на любой другой итерации – это только усложнило бы обозначения. Пусть π^0 – некоторая вспомогательная (не случайная) популяция и ξ^1 – первый ее генотип.

Пусть $k(\xi^1)$ – число элементов в пути $\eta^0 = \xi^1, \eta^1, ..., \eta^{k(\xi^1)}$ от генотипа ξ^1 до B^* из условия $\mathcal{A}_2$ для оператора вариации, и пусть $k^* = \max\{k(\xi) : \xi \in B\}$. Очевидно, k^* конечно, т.к. B конечно.

Обозначим через $U(\xi^1, i)$ событие $\{\eta^0 \in \widehat{\Pi^0}, \eta^1 \in \widehat{\Pi^1}, ..., \eta^i \in \widehat{\Pi^i}\}$. Рассмотрим условную вероятность $p(i) = P\{\eta^{i+1} \in \widehat{\Pi^{i+1}} | U(\xi^1, i)\ \&\ \Pi^0 = \pi^0\}$ при наличии генотипов $\eta^j \in \widehat{\Pi^j}$ на всех итерациях j от 0 до $i-$той включительно, получить в новой популяции Π^{i+1} следующий генотип η^{i+1}. Здесь i принимает значения от 0 до $k(\xi^1) - 1$. Покажем, что $p(i) > 0$.

С учетом Утверждения 4.3.1, $p(i)$ оценивается снизу

$$p(i) \geq P\{\eta^{i+1} \in \widehat{\Pi^{i+1}}\ \&\ \eta^{i+1} \in \widehat{\Pi''}\ \&\ \eta^i \in \widehat{\Pi'} | U(\xi^1, i)\ \&\ \Pi^0 = \pi^0\} =$$

$$= p_{\text{surv}} \cdot p_{\text{sel \& var}}, \tag{4.5}$$

где p_{surv} обозначает условную вероятность

$$P\{\eta^{i+1} \in \widehat{\Pi^{i+1}} | \eta^{i+1} \in \widehat{\Pi''}\ \&\ \eta^i \in \widehat{\Pi'}\ \&\ U(\xi^1, i)\ \&\ \Pi^0 = \pi^0\},$$

а $p_{\text{sel \& var}}$ обозначает условную вероятность

$$P\{\eta^{i+1} \in \widehat{\Pi''}\ \&\ \eta^i \in \widehat{\Pi'} | U(\xi^1, i)\ \&\ \Pi^0 = \pi^0\}.$$

По Утверждению 4.3.1,

$$p_{\text{sel \& var}} = P\{\eta^{i+1} \in \widehat{\Pi''} | \eta^i \in \widehat{\Pi'}\ \&\ U(\xi^1, i)\ \&\ \Pi^0 = \pi^0\}.$$

$$\cdot P\{\eta^i \in \widehat{\Pi}' | U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\}. \tag{4.6}$$

Покажем, что последний сомножитель положителен. Заметим, что по утверждению 4.3.2

$$P\{\eta^i \in \widehat{\Pi}' | U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\} =$$

$$= \sum_{\pi:\ \eta^i\ \in \widehat{\pi}} P\{\eta^i \in \widehat{Select(\Pi^i)} | \Pi^i = \pi\} P\{\Pi^i = \pi | U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\} \geq$$

$$\geq \epsilon_1 \cdot \sum_{\pi\ :\ \eta^i \in \widehat{\pi}} P\{\Pi^i = \pi | U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\} =$$

$$= \epsilon_1 \cdot \sum_{\pi\ :\ \eta^i \in \widehat{\pi}} \frac{P\{\Pi^i = \pi \ \& \ U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\}}{P\{U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\}} = \epsilon_1.$$

Здесь суммирование ведется только по тем π, которые содержат η^i ввиду того, что оператор селекции может построить популяцию с η^i только если особь η^i имелась во входной популяции. Итак, положительность последнего сомножителя в (4.6) доказана.

Аналогичным способом, используя утверждение 4.3.2 и условие $\mathcal{A}_2$ для вариации, получаем $P\{\eta^{i+1} \in \widehat{\Pi''} | \eta^i \in \widehat{\Pi}' \ \& \ U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\} \geq \epsilon_2$. Следовательно, $p_{\text{sel \& var}} \geq \epsilon_2 \epsilon_1$.

Из условия $\mathcal{A}_3$ и утверждения 4.3.2 также вытекает, что $p_{\text{surv}} \geq \epsilon_3$. Таким образом, существует достаточно малая положительная величина $\delta = \epsilon_1 \epsilon_2 \epsilon_3$, такая что $0 < \delta < p(i)$.

Далее, пусть $p^*(\pi^0)$ будет условной вероятностью при $\Pi^0 = \pi^0$ попасть в B^* на итерации $k(\xi^1)$, следуя за цепочкой генотипов из условия $\mathcal{A}_2$ для вариации: $\eta^0 = \xi^1, \eta^1 \in \widehat{\Pi^1}, \eta^2 \in \widehat{\Pi^2}, ..., \eta^{k(\xi^1)} \in \widehat{\Pi^{k(\xi^1)}} \cap B^*$.

Заметим, что по лемме 2.5.3 для любых событий $A_0, A_1, ..., A_n$,

$$P\{A_0 \& A_1 \& ... \& A_n\} = P\{A_0\} \prod_{i=0}^{n-1} P\{A_{i+1} | A_0 \& A_1 \& ... \& A_i\}.$$

Таким образом,

$$p^*(\pi^0) = \prod_{i=0}^{k(\xi^1)-1} P\{\eta^{i+1} \in \widehat{\Pi^{i+1}} | U(\xi^1, i) \ \& \ \Pi^0 = \pi^0\}. \tag{4.7}$$

Оценивая $p^*(\pi^0)$ с помощью δ, имеем: $p^*(\pi^0) \geq \delta^{k(\xi^1)}$. Пусть $\Delta = \delta^{k^*}$. Тогда $\min_{\pi^0 \in B^N} p^*(\pi^0) \geq \Delta$, и

$$\max_{\pi^0 \in B^N} P\{F_t < f^*, t = 0, ..., k(\xi^1) | \Pi^0 = \pi^0\} \leq \max_{\pi^0 \in B^N} (1 - p^*(\pi^0)) \leq 1 - \Delta < 1. \tag{4.8}$$

Обозначим через $A(\vartheta)$ событие, состоящее в отсутствии оптимальных генотипов до итерации ϑ включительно. С учетом леммы 2.5.3, марковости операторов ЭА и формулы (4.8), для любого s имеем $P\{A(s\cdot(k^*+1))\}\le P\{A((s-1)(k^*+1))\}(1-\Delta)$, откуда по индукции заключаем, что

$$P\{A(\vartheta)\}\le(1-\Delta)^{\lfloor\vartheta/(k^*+1)\rfloor}.$$

Тогда с использованием теоремы 4.3.1 о непрерывности вероятностной меры получаем:

$$P\{\widehat{\Pi^t}\cap B^*=\emptyset\ \forall t\}=P\left\{\bigcap_{\vartheta=0}^{\infty}A(\vartheta)\right\}=$$

$$=\lim_{\vartheta\to\infty}P\left\{A(\vartheta)\right\}\le\lim_{\vartheta\to\infty}(1-\Delta)^{\lfloor\vartheta/(k^*+1)\rfloor}=0.$$

2. В случае, если для любых π,π''

$$\max\{f(x(\xi)):\xi\in\widehat{Survive}\,(\pi,\pi'')\}\ge\max\{f(x(\xi)):\xi\in\widehat{\pi}\cup\widehat{\pi''}\},$$

после обнаружения оптимального генотипа, в каждой популяции Π^t будет присутствовать такой генотип, и значит, порождение оптимума за конечное число итераций обеспечивает сходимость популяции ЭА к оптимуму п.н. □

Определение 4.3.3. *Оператор вариации $Variate(\Pi')$ удовлетворяет условию $\mathcal{A}_2'$, если существует такое $\epsilon>0$, что для любых $\Pi'\in B^{N'}$, $\eta\in B$ и $t\ge0$*

$$P\{\eta\in\widehat{Variate}(\Pi')\}\ge\epsilon.$$

Замечание 4.3.1. *Пусть $B^*\ne\emptyset$ и критерий остановки никогда не выполняется. Тогда при условиях $\mathcal{A}_2'$ и $\mathcal{A}_4$ имеет место сходимость популяции ЭА к оптимуму почти наверное.*

Доказательство аналогично п.1 теоремы 4.3.2, т.к., полагая $k(\xi^1)=1$, ввиду $\mathcal{A}_2'$ имеем $\delta>0$ для любой $\pi^0\in B^N$.

Утверждение 4.1. *Если $0<p_m<1$, $p_c<1$, $B^*\ne\emptyset$, и для любого $\xi\in B$ выполнено $\Phi(\xi)>0$ тогда КГА удовлетворяет условиям $\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3$.*

Утверждение 4.2. *Если $0<p_m$, $p_c<1$, и существует $\xi\in B$, такое что $f(x(\xi))<f(x^*)$ тогда КГА не удовлетворяет условию $\mathcal{A}_4$.*

Утверждение 4.3. *Если $p_m=0$ или $p_m=1$ и существует $\xi\in B$, такое что $f(x(\xi))<f(x^*)$ и $f(\mathbf{1}-x(\xi))<f(x^*)$, где $\mathbf{1}=(1,1,\ldots,1)$ - сторока из ℓ единиц, тогда при любом p_c КГА не удовлетворяет условию $\mathcal{A}_2$.*

Утверждение 4.4. *Если (λ,λ) EA имеет оператор мутации где вероятность битовой мутации $p_m=0.5$ и $B^*\ne\emptyset$, то выполнены условия $\mathcal{A}_1\ldots,\mathcal{A}_3$.*

Примеры. Предположение $B^* \neq \emptyset$ будем считать выполненным.

1. КГА при $0 < p_m < 1$ обладает оператором вариации со свойством $\mathcal{A}_2'$, т.к. при мутации из любого генотипа с ненулевой вероятностью может быть получен любой генотип.

2. Если в КГА $p_m = 0$ или $p_m = 1$, то оператор вариации КГА не удовлетворяет условию $\mathcal{A}_2$, и можно привести примеры начальных популяций, начиная с которых КГА не сможет никогда найти некоторые решения.

3. Если в КГА оператор мутации заменить на одноточечную мутацию в случайно выбранном гене (с равномерным распределением), то оператор вариации будет удовлетворять условию $\mathcal{A}_2$ и будет применим п.1 теоремы о сходимости ЭА.

4. К КГА неприменим п.2 теоремы о сходимости ЭА, т.к. оператор выживания не удовлетворяет условию $\mathcal{A}_4$ и однажды найденное оптимальное решение может быть потеряно. Однако, это свойство может быть достаточно легко обеспечено, например, как это делается в ГА с элитой.

5. Популяция $(\mu+\lambda)$-ES или $(\mu+\lambda)$-EA с оператором мутации, который выдает генотип из B^* с ненулевой вероятностью (например, при мутации с нормально распределенным шагом), сходится к оптимуму почти наверное, ввиду замечания 4.3.1.

Замечание 4.3.2. *Свойства сходимости, полученные в последней теореме (хотя бы в смысле п.1, как в КГА), являются желательными для всякого ЭА. Однако, даже наличие такой сходимости не дает гарантии «надежной работы» алгоритма, т.к. на практике число выполняемых итераций за реальное время может оказаться слишком мало, чтобы вероятность получения оптимума приблизилась к 1.*

Как видно из примера 5, даже самый примитивный алгоритм (1+1)-EA с оператором мутации, имеющим равномерное распределение на множестве генотипов, обладает всеми свойствами сходимости в смысле теоремы 4.3.2. Наконец, детерминированный полный перебор генотипов всегда за конечное время находит лучший генотип, а значит, является сходящимся методом (как детерминированный алгоритм).

Из замечания 4.3.2 вытекает необходимость более детального исследования скорости сходимости ЭА к оптимуму или времени первого достижения оптимума с учетом других свойств задачи, например, как в следствии 2.5.1.

4.4. Алгоритмы генетического программирования

Алгоритмы генетического программирования (ГП) были предложены Н.Л.Крамером [42] и развиты далее в работах Дж. Козы [75] и других авторов. Идея ГП заключается в том, что в отличие от ГА, здесь все операции производятся не над строками, а над деревьями. При этом используются операторы, аналогичные селекции, скрещиванию и мутации ГА. С помощью деревьев предлагается кодировать программы для ЭВМ и математические формулы - таким образом можно организовать эволюцию программного кода для решения программистской задачи или поиск подходящей функции в аналитическом виде.

Можно считать, что в ГП фенотипом является программа, представленная как дерево с терминальными (листья дерева) и функциональными элементами (все прочие вершины). Терминальные элементы соответствуют константам, действиям и функциям без аргументов, а функциональные - функциям, использующим аргументы. Например, рассмотрим функцию x*3/5-1. Терминальные элементы здесь: $\{x, 3, 5, 1\}$, функциональные: $\{+, *, /\}$.

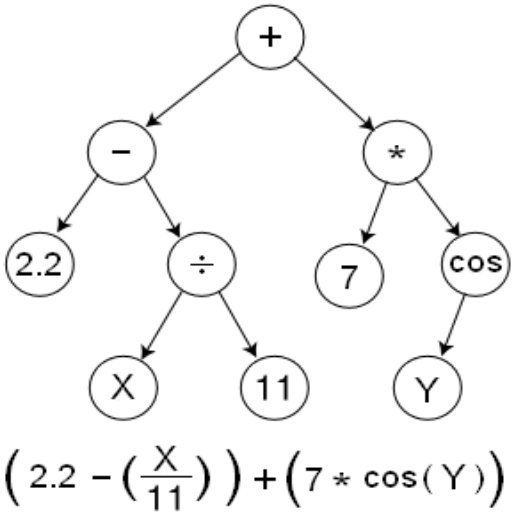


Схема ГП аналогична схеме ГА, однако операторы скрещивания и мутации имеют отличия. В операторе скрещивания выбираются случайные поддеревья родительских генотипов и происходит обмен.

При построении начальной популяции и в операторах вариации уделяется внимание глубине, ширине и структуре формируемых деревьев [75, 84]. Например для предотвращения чрезмерного разрастания дерева в высоту вводится ограничение на максимальное количество функциональных элементов в дереве или максимальную его глубину. Если при скрещивании двух деревьев один из потомков не удовлетворяет такому ограничению, вместо него копируется родительское дерево.

При действии оператора мутации случайно удаляется часть дерева и

вместо нее генерируется новое поддерево случайным образом. В некоторых случаях мутация сводится к случайному изменению терминальных элементов (тогда для каждого типа элементов должно быть задано распределение вероятностей, определяющее случайные изменения).

ГП может рассматриваться как частный случай ГА с изменяющейся длиной кодировки и специфическими операторами кроссинговера и мутации, поэтому для них можно применять теорему о сходимости и доказывать аналоги теоремы о схемах.

Алгоритмы ГП используются для решения задач построения нелинейных моделей (математических выражений, функций, алгоритмов, программ) на основе заданных экспериментальных данных, множества переменных, базовых функций и операций (например для предсказания уровня воды в водохранилище). Также алгоритмы ГП применяются в синтезе решающих деревьев и в некоторых других методах машинного обучения.

Работоспособность алгоритма ГП существенно зависит от выбора оператора кроссинговера, где родительские решения (деревья) обмениваются своими признаками. Наиболее распространенными вариантами оператора кроссинговера являются одноточечный кроссинговер [49] и равномерный кроссинговер [84]. При одноточечном кроссинговере в родительских решениях выбираются узлы и осуществляется обмен поддеревьями с корневыми вершинами в выбранных узлах. (рис. 4.1). Равномерный кроссинговер характеризуется тем, что значение каждого из узлов потомка наследуется из соответствующих значений в родительских деревьях с заданной вероятностью. (рис. 4.2).

Известны различные вариации указанных операторов кроссинговера, где учитываются координаты и значения вершин поддеревьев, их размеры и другие свойства (см., например, [78, 80]).

4.5. Алгоритм имитации отжига (Simulated annealing)

Алгоритм мотивирован аналогией с металлургическим производством и применим как к дискретным, так и к непрерывным задачам.

Схема алгоритма Имитации отжига (ИО) [74]

1. Построить решение y_1 случайным образом.
2. Положить $\tilde{y} := y_1; \tilde{F} := F(y_1)$.

Итерация t.

3. Выбрать $y \in \mathcal{N}(y_t)$ случайным образом.
4. Если $F(y) > \tilde{f}$ – положить $\tilde{F} := F(y), \tilde{y} := y$.
5. Выбрать α с равномерным распределением из отрезка [0,1], и если $\alpha \leq \exp\left\{\frac{F(y)-F(y_t)}{\tau(t)}\right\}$, то положить $y_{t+1} := y$, иначе положить $y_{t+1} := y_t$.

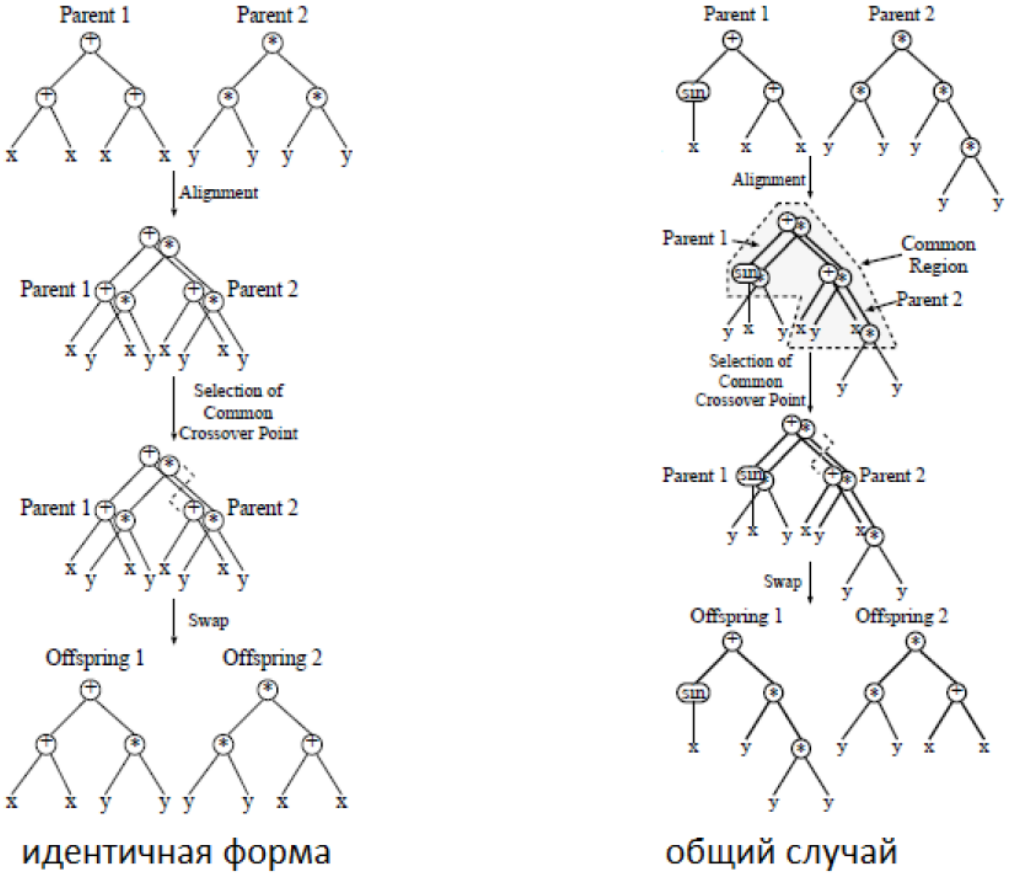


Рис. 4.1. Одноточечный кроссинговер

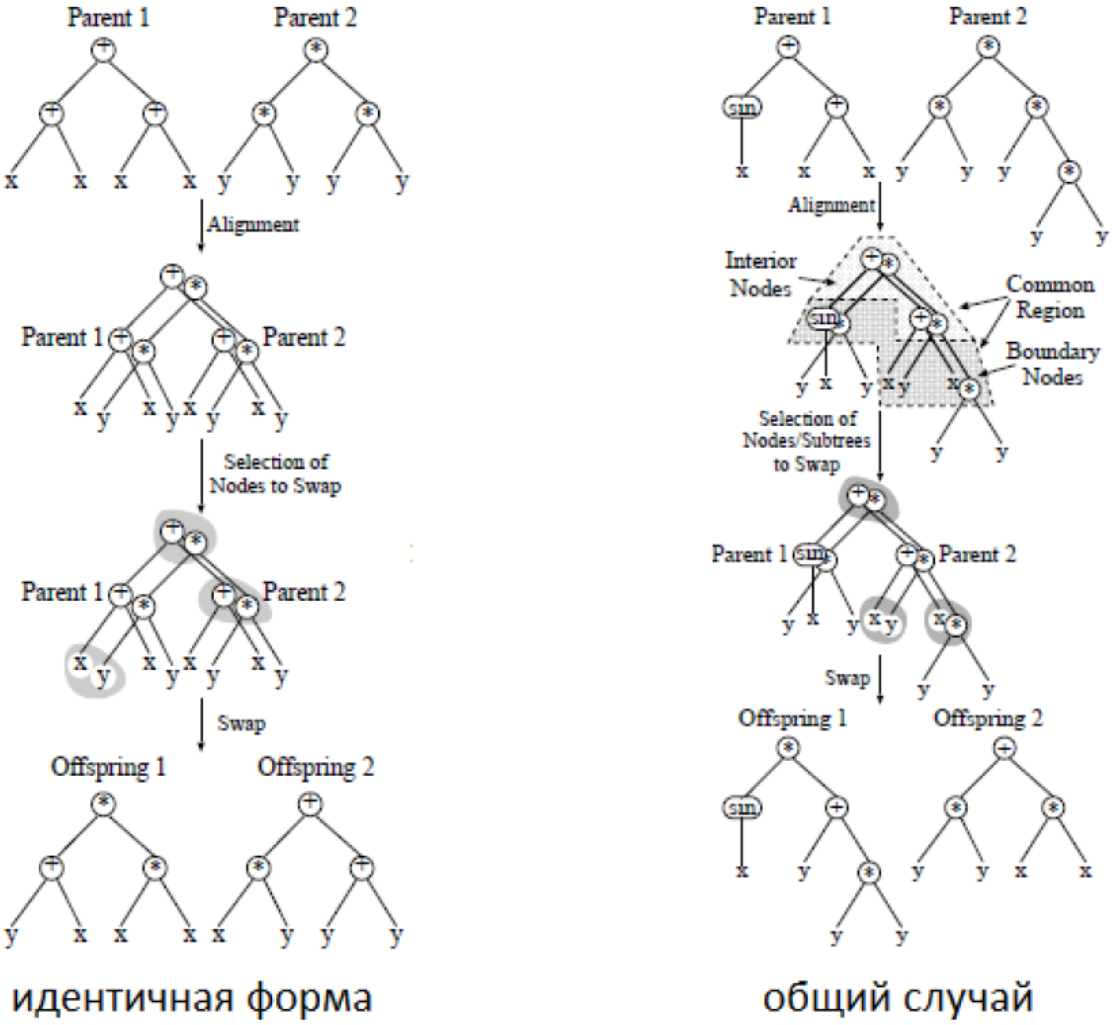


Рис. 4.2. Равномерный кроссинговер

6. Положить $t := t + 1$.
7. Если условие остановки не выполнено, то перейти на шаг 3.

Параметр $\tau(t)$ принято называть *температурой*, он постепенно снижается в процессе работы алгоритма. На практике чаще всего используют экспоненциальный график охлаждения: $\tau(t) = \tau(0)a^{\lfloor t/L \rfloor}$.

Пример (graph bipartition): Найти разбиение вершин графа на два равномощных множества, минимизирующее число ребер с концами в разных множествах. Решение x в данном случае – разрез.

Рекомендации: $\tau(0) \approx \max\{|F(y) - F(x)| : y \in N(x)\}$; $a \approx 0.95$; $L \approx \max\{|N(x)|, x \in D\}$. Подробнее см. в [28].

Замечание. Алгоритм ИО – частный случай ЭА, где распределение оператора выживания зависит от номера итерации t.

Замечание. В алгоритме ИО условие невырожденного оператора выживания не выполнено, поэтому теорема о сходимости ЭА неприменима.

Теорема 4.5.1. *(следствие из более общей теоремы [19,28]) Пусть система окрестностей такова, что за конечное число переходов в пределах окрестности текущей точки можно достичь оптимум. Тогда*

$$\lim_{\tau \to 0} \lim_{t \to \infty} P\{F(y_t) = F^*\} = 1.$$

4.6. Алгоритм поиска с запретами (tabu search)

Алгоритм поиска с запретами предложен Ф. Гловером (F.Glover) в 1986г. (см. [19, 85]). В классическом алгоритме локального поиска всякий раз выбирается лучшая точка в окрестности, или точка, имеющая значение целевой функции не хуже, чем текущая. Если текущая точка является локальным оптимумом, то на следующей итерации, вероятен возврат в этот же локальный оптимум. В алгоритме поиска с запретами разработан механизм предотвращения таких возвратов, основаный на использовании списка Λ из последних L решений: $\Lambda = (\ell^1, \ldots, \ell^L) = (\mathbf{y}^{t-L}, ..., \mathbf{y}^{t-1})$. Множество запрещенных решений при текущем решении $\mathbf{y}^t$ и списке Λ будем обозначать через $Tabu(\mathbf{y}^t, \Lambda)$.

Как показывает практика, часто удобнее хранить список запрещенных "ходов" (tabu list), т.е. список некоторых способов изменения решений, которые приводят к запрещенным решениям.

Примеры запрещенных "ходов": недавно использованное увеличение (или уменьшение) некоторой координаты; переход в решение со значением

целевой функции, присутствующим в списке $f(\ell^1), ..., f(\ell^L)$; то и другое одновременно и т.д.

Было бы странно, если список запретов не позволял переходить в заведомо лучшие решения, по сравнению с найденными ранее. Для того, чтобы этого не происходило, используется "условие стремления" (aspiration condition): если целевая функция в некоторой точке $\mathbf{x}$ из текущей окрестности выше значения целевой функции лучшего из найденных прежде решений, то следует разрешить переход в решение $\mathbf{x}$, независимо от Λ. Множество решений, попадающих в условия "стремления" при текущем рекорде $\tilde{f}$ будем обозначать через $Asp(\tilde{f})$.

Часто имеется некоторый способ выделения подокрестности $\mathcal{N}'(\mathbf{y}^t) \subseteq \mathcal{N}(\mathbf{y}^t)$ в окрестности $\mathcal{N}(\mathbf{y}^t)$ (детерминированно или случайно). Это позволяет снизить трудоемкость поиска и "разнообразить" его.

Общая схема алгоритма поиска с запретами

1. Выбрать $\mathbf{y}^1$ случайным образом.
2. Положить $\tilde{\mathbf{y}} := \mathbf{y}^1$; $\tilde{f} := f(\mathbf{y}^1)$.
3. Инициализировать список $\Lambda = (\ell^1, \ldots, \ell^L) := (\mathbf{y}^1, \mathbf{y}^1, ..., \mathbf{y}^1)$;

Итерация t.

4. Положить $f' := -\infty; \mathbf{y}' := \mathbf{y}^t$.
5. Для всех $\mathbf{y} \in \mathcal{N}'(\mathbf{y}^t) \backslash (Tabu(\mathbf{y}^t, \Lambda) \backslash Asp(\tilde{f}))$:
   если $f(\mathbf{y}) > f'$, то положить$f' := f(\mathbf{y}), \mathbf{y}' := \mathbf{y}$.
6. Положить $\mathbf{y}^{t+1} := \mathbf{y}'$.
7. Если $f' > \tilde{f}$, то положить $\tilde{f} := f(\mathbf{y}'), \tilde{\mathbf{y}} := \mathbf{y}'$.
8. Обновить список Λ: исключить первый элемент, сдвинуть все элементы влево на одну позицию, положить $\ell^L := \mathbf{y}^t$.
9. Положить $t := t + 1$.
10. Если $Terminate$ =«ложь», то перейти на шаг 4.

Замечание. Алгоритм поиска с запретами – частный случай ЭА, где $N = L + 2$ и оператор выживания является консервативным.

Вероятностный поиск с запретами (PTS).

Вероятностный поиск с запретами предложен Гончаровым Е. Н. и Кочетовым Ю. А. [7]. Пусть $D = X = \{0, 1\}^n$ – множество допустимых решений. $\mathcal{N}(\mathbf{x}) = \{\mathbf{y} \in D : \rho(\mathbf{x}, \mathbf{y}) \leq d\}$, где $\rho(\mathbf{x}, \mathbf{y})$ - расстояние в метрике Хэмминга. Пусть $P\{\mathbf{x}' \in \mathcal{N}'(\mathbf{x})\} = p$ – заданная константа для любых $\mathbf{x} \in D, \mathbf{x}' \in \mathcal{N}(\mathbf{x})$. (Очевидно, в таком случае $P\{\mathcal{N}'(\mathbf{x}) = \emptyset\} = (1-p)^{|\mathcal{N}(\mathbf{x})|} > 0$, если $p < 1$.)

$$Tabu(\mathbf{x}, \Lambda) =$$

$$\left\{\mathbf{y} \in D : \ \exists\, k \in \{1, ..., L\} : \ |(\mathbf{y} - \mathbf{x})_i| = |(\ell^{k+1} - \ell^k)_i| \ \forall\, i = 1, .., n\right\},$$

где под ℓ^{L+1} понимается текущее решение $\mathbf{x}$.

Настраиваемые параметры: p, d, L. Как правило, $d = 1$ или 2.

Теорема 4.6.1. *О сходимости вероятностного поиска с запретами [66]. Пусть $p \in (0,1)$. Тогда в вероятностном поиске с запретами $f(\mathbf{y}^t) \to f^*$ при $t \to \infty$ почти наверное.*

Если при пустой подокрестности $\mathcal{N}'(\mathbf{y}^t)$ список запретов оставлять без изменений и $0 < L < (n-1)n/4$, то указанная сходимость также имеет место [7].

4.7. Динамика численности особей с высокой приспособленностью в популяции эволюционного алгоритма

Настоящий раздел посвящен оценкам численности особей достаточно высокого качества на заданной итерации эволюционного алгоритма с (μ, λ)-селекцией и полной заменой популяции (т.е. без элиты) и без оператора кроссинговера (т. е. когда оператор кроссинговера представляет собой тождественное отображение). Для анализа ЭА предлагается математическая модель эволюционного процесса, возникающего при его работе. С использованием данной модели строятся нижние и верхние оценки среднего числа особей с приспособленностью не ниже заданного порогового значения (см., например, [57]).

При отсутствии кроссинговера в общей схеме ГА с полной заменой популяции нет необходимости генерировать особи каждой новой популяции обязательно парами. В связи с этим, для большей общности здесь будем предполагать, что условие о четности λ отсутствует и ЭА имеет следующую схему.

Алгоритм 4.1. ЭА без кроссинговера с полной заменой популяции

1. Положить $t := 0$.
2. Для k от 1 до λ выполнять:
2.1. Построить случайным образом особь $\xi^{k,0}$.

Итерация t.
3. Для k от 1 до λ выполнять шаги 3.1–3.2:
3.1. Селекция: выбрать особь $\xi := \xi^{\mathrm{Sel}(X^t),t}$.
3.2. Мутация: положить $\xi^{k,t+1} := \mathrm{Mut}(\xi)$.
4. Положить $t := t + 1$.
5. Если условие остановки выполнено, то идти на шаг 6, иначе – на шаг 3.
6. Результатом работы является особь с наибольшей приспособленностью среди найденных за все итерации.

Пусть заданы линии уровня функции приспособленности $\Phi_0, \ldots, \Phi_d$, такие что

$$\Phi_0 < \Phi_1 < \Phi_2 \ldots < \Phi_d.$$

Число линий уровня и соответствующие им значения функции приспособленности (кроме Φ_0) могут быть выбраны достаточно произвольно с учетом свойств решаемой задачи и оператора мутации.

Сопоставим выбранным линиям уровня следующие подмножества:

$$H_i = \{\xi : \Phi(\xi) \geq \Phi_i\}, \quad i = 0, \ldots, d.$$

Предположим, что Φ_0 достаточно мало, так что $H_0 = \mathcal{B}$. Для удобства обозначений положим $H_{d+1} = \emptyset$.

Пусть для всех $i = 0, ..., d$ и $j = 1, ..., d$ априори известны нижние α_{ij} и верхние β_{ij} оценки вероятности перехода из $H_i \backslash H_{i+1}$ в H_j при действии оператора мутации, а именно для любого $\xi \in H_i \backslash H_{i+1}$

$$\alpha_{ij} \leq \mathbf{P}\{\mathrm{Mut}(\xi) \in H_j\} \leq \beta_{ij},$$

где

$$\mathbf{P}\{\mathrm{Mut}(\xi) \in H_j\} = \sum_{\xi' \in H_j} \mathbf{P}\{\mathrm{Mut}(\xi) = \xi'\}.$$

Очевидно, что для любого оператора мутации существуют монотонные оценки (например, нулевая матрица $\{\alpha_{ij}\}$ и матрица $\{\beta_{ij}\}$ со всеми элементами, равными 1). Однако трудности могут возникать только из-за отсутствия оценок, достаточно точно передающих свойства этого оператора.

Представим популяцию на шаге t при помощи *вектора популяции*

$$\mathbf{z}^{(t)} = (z_1^{(t)}, z_2^{(t)}, \ldots, z_d^{(t)}),$$

где $z_i^{(t)} \in \mathbb{R}$ - доля генотипов из H_i в популяции X^t. Таким образом, $\mathbf{z}^{(t)}$ – случайный вектор, причем $z_i^{(t)} \geq z_{i+1}^{(t)}$ для $i = 1, ..., d-1$, так как $H_{i+1} \subseteq H_i$. Далее для удобства будем полагать $z_{d+1} = z_{d+1}^{(t)} = 0$ при любом t.

Пусть $\mathbf{P}\{\xi^{(t)} \in H_j\}$ – вероятность того, что генотип, полученный в результате селекции из популяции с номером $t-1$, после мутации будет принадлежать H_j. Заметим, что все особи поколения t в рассматриваемом алгоритме генерируются с одним и тем же распределением вероятностей, следовательно, $\mathbf{P}\{\xi^{(t)} \in H_j\} = \mathbf{P}\{\xi_1^{(t)} \in H_j\} = ... = \mathbf{P}\{\xi_\lambda^{(t)} \in H_j\}$ при любом $t > 0$ и $j = 1, ..., d$. Справедливо следующее

Утверждение 4.5. *При любом $t > 0$ имеет место равенство* $\mathbf{E}[z_i^{(t)}] = \mathbf{P}\{\xi^{(t)} \in H_i\}$.

Доказательство. Рассмотрим последовательность одинаково распределенных случайных величин $\mathcal{I}_1^i, \mathcal{I}_2^i, ..., \mathcal{I}_\lambda^i$, где $\mathcal{I}_k^i = 1$, если k-я особь популяции X^t лежит в H_i, и $\mathcal{I}_k^i = 0$ в противном случае. По определению вектора популяции имеем $z_i^{(t)} = \sum_{k=1}^{\lambda} \mathcal{I}_k^i/\lambda$, следовательно,

$$\mathbf{E}[z_j^{(t)}] = \sum_{k=1}^{\lambda} \frac{\mathbf{E}[\mathcal{I}_k^i]}{\lambda} = \sum_{k=1}^{\lambda} \frac{\mathbf{P}\{\xi^{(t)} \in H_i\}}{\lambda} = \mathbf{P}\{\xi^{(t)} \in H_i\}.$$

Получим нижние и верхние оценки доли особей с генотипами из подмножеств H_i для $i = 1, \ldots, d$ на заданной итерации.

Обозначим через $\mathbf{P}_{\text{ch}}(S, \mathbf{z})$ вероятность того, что генотип, выбранный при помощи (μ, λ)-селекции из популяции с вектором $\mathbf{z}$, принадлежит подмножеству $S \subseteq B$. Заметим, что если текущая популяция представляется вектором $\mathbf{z}^{(t)} = \mathbf{z}$, то условная вероятность того, что очередной генотип, полученный в результате селекции и мутации, принадлежит H_j, имеет вид:

$$\mathbf{P}\{\xi^{(t+1)} \in H_j | \mathbf{z}^{(t)} = \mathbf{z}\} = \sum_{i=0}^{d} \sum_{\xi \in H_i \setminus H_{i+1}} \mathbf{P}\{\text{Mut}(\xi) \in H_j\} \mathbf{P}_{\text{ch}}(\{\xi\}, \mathbf{z}). \quad (4.9)$$

Из формулы (4.9) и определения оценок β_{ij} следует, что

$$\mathbf{P}\{\xi^{(t+1)} \in H_j | \mathbf{z}^{(t)} = \mathbf{z}\} \leq \sum_{i=0}^{d} \beta_{ij} \sum_{\xi \in H_i \setminus H_{i+1}} \mathbf{P}_{\text{ch}}(\{\xi\}, \mathbf{z}) =$$

$$= \sum_{i=0}^{d} \beta_{ij} \mathbf{P}_{\text{ch}}(H_i \setminus H_{i+1}, \mathbf{z}).$$

При заданном параметре μ имеем $\mathbf{P}_{\text{ch}}(H_i, \mathbf{z}^{(t)}) = z_i^{(t)} \lambda/\mu$, если $z_i^{(t)} \leq \mu/\lambda$ и $\mathbf{P}_{\text{ch}}(H_i, \mathbf{z}^{(t)}) = 1$, при более высоких значениях $z_i^{(t)}$. Поскольку данная функция зависит только от компоненты $z_i^{(t)}$, а не от всего вектора $\mathbf{z}^{(t)}$, для более компактной записи будем обозначать $\mathbf{P}_{\text{ch}}(z_i^{(t)}) := \mathbf{P}_{\text{ch}}(H_i, \mathbf{z}^{(t)})$. Отсюда получаем оценку сверху:

$$\mathbf{P}\{\xi^{(t+1)} \in H_j | \mathbf{z}^{(t)} = \mathbf{z}\} \leq \sum_{i=0}^{d} \beta_{ij} (\mathbf{P}_{\text{ch}}(H_i, \mathbf{z}^{(t)}) - \mathbf{P}_{\text{ch}}(H_{i+1}, \mathbf{z}^{(t)})).$$

Пусть $Z_\lambda = \{\mathbf{z} \in \mathbb{R}^d : z_i \in \{0, \frac{1}{\lambda}, \frac{2}{\lambda}, \ldots, 1\}, z_i \geq z_{i+1}\}$ – множество векторов популяции из λ особей. По формуле полной вероятности получаем безусловную вероятность порождения очередной особи из H_j на шаге $t+1$:

$$\mathbf{P}\{\xi^{(t+1)} \in H_j\} = \sum_{\mathbf{z} \in Z_\lambda} \mathbf{P}\{\xi^{(t+1)} \in H_j | \mathbf{z}^{(t)} = \mathbf{z}\} \mathbf{P}\{\mathbf{z}^{(t)} = \mathbf{z}\} \leq \quad (4.10)$$

$$\leq \sum_{\mathbf{z}\in Z_\lambda}\sum_{i=0}^{d}\beta_{ij}(\mathbf{P}_{\rm ch}(z_i)-\mathbf{P}_{\rm ch}(z_{i+1}))\mathbf{P}\{\mathbf{z}^{(t)}=\mathbf{z}\}=$$

$$=\sum_{i=0}^{d}\beta_{ij}\mathbf{E}[\mathbf{P}_{\rm ch}(z_i^{(t)})-\mathbf{P}_{\rm ch}(z_{i+1}^{(t)})]=\mathbf{E}[\beta_{0j}\mathbf{P}_{\rm ch}(z_0^{(t)})]+\sum_{i=1}^{d}(\beta_{ij}-\beta_{i-1,j})\mathbf{E}[\mathbf{P}_{\rm ch}(z_i^{(t)})].$$

Последнее равенство получено перегруппировкой слагаемых.

Ввиду утверждения 4.5, $\mathbf{E}[z_j^{(t+1)}]=\mathbf{P}\{\xi^{(t+1)}\in H_j\}$. Следовательно, с учетом того, что $\mathbf{P}_{\rm ch}(z_{d+1})=0$ и $\mathbf{P}_{\rm ch}(z_0)=1$, из (4.10) получаем:

$$\mathbf{E}[z_j^{(t+1)}]\leq\beta_{0j}+\sum_{i=1}^{d}(\beta_{i,j}-\beta_{i-1,j})\mathbf{E}[\mathbf{P}_{\rm ch}(z_i^{(t)})]. \tag{4.11}$$

Матрицу размерности $d\times d$ с элементами γ_{ij} будем называть *монотонной*, если $\gamma_{i-1,j}\leq\gamma_{i,j}$ для всех i,j от 1 до d.

Для оценки сверху математического ожидания $\mathbf{E}[\mathbf{P}_{\rm ch}(z_i^{(t)})]$ применим неравенство Иенсена (под знаком математического ожидания стоит вогнутая функция от $z_i^{(t)}$). В результате имеем

Утверждение 4.6. *[57] При условии монотонности матрицы оценок $\{\beta_{ij}\}$ математическое ожидание компонента j вектора популяции $\mathbf{z}^{(t+1)}$ удовлетворяет неравенству*

$$\mathbf{E}[z_j^{(t+1)}]\leq\beta_{0j}+\sum_{i=1}^{d}(\beta_{ij}-\beta_{i-1,j})\mathbf{P}_{\rm ch}(\mathbf{E}[z_i^{(t)}]) \tag{4.12}$$

для всех $j=1,\ldots,d$.

Для получения нижних оценок на $\mathbf{E}[z_j^{(t+1)}]$ вместо неравенства Иенсена может быть использовано более простое неравенство $\mathbf{P}_{\rm ch}(z_i)\geq z_i$, справедливое для любого $z_i\in[0,1]$, из которого следует, что $\mathbf{E}[\mathbf{P}_{\rm ch}(z_i^{(t)})]\geq\mathbf{E}[z_i^{(t)}]$. Таким образом, аналогично утверждению 4.6 может быть получено

Утверждение 4.7. *При условии монотонности матрицы $\{\alpha_{ij}\}$ для всех $j=1,\ldots,d$ имеет место оценка*

$$\mathbf{E}[z_j^{(t+1)}]\geq\alpha_{0j}+\sum_{i=1}^{d}(\alpha_{ij}-\alpha_{i-1,j})\mathbf{E}[z_i^{(t)}]. \tag{4.13}$$

С использованием того же подхода, что и в утверждении 4.7, в [36, 37] нами получены нижние и верхние оценки вероятности получения генотипа из множества $H_j,\ j=1,\ldots,d$, на заданной итерации эволюционных стратегий

(1+1) EA и (1,λ) EA. Также в этих работах исследуется, насколько велики могут быть нижние оценки α_{ij} для полиномиально вычислимого оператора мутации при решении полиномиально ограниченной NP-трудной задачи. Установлено, что в случае эффективной вычислимости функции $\Phi(x)$ и оператора Mut, если верна известная гипотеза NP $\not\subseteq$ BPP (см., например, [18]), то для полиномиально ограниченных NP-трудных задач *все* оценки вида $\alpha_{i,i+1}$ не могут быть существенно отличны от нуля.

Литература

1. Алтухов Ю. П. Генетические процессы в популяциях. – М.: Академкнига, 2003. – 431 с.

2. Береснев В.Л., Гимади Э.Х., Деменьтьев В.Т. Экстремальные задачи стандартизации. - Новосибирск: Наука, 1978. - 385 с.

3. Борисовский П.А., Еремеев А.В. Генетический алгоритм для задачи о вершинном покрытии графа // Межвузовский сборник научных трудов «Математика и информатика: наука и образование», Вып. 7. Омск: ОмГПУ, 2008. - С.49-54.

4. П. А. Борисовский, А. В. Еремеев, О сравнении некоторых эволюционных алгоритмов, Автомат. и телемех., 2004, № 3, с. 3–9.

5. Боровков А.А. Курс теории вероятностей. - М.: Наука, 1972.

6. Гнеденко Б. В. Курс теории вероятностей. – М.: Наука, 1988. – 451 с.

7. Гончаров Е.Н., Кочетов Ю.А. Вероятностный поиск с запретами для дискретных задач безусловной оптимизации. // Дискретный анализ и исследование операций. Серия 2, 2002, Т. 9, № 2 с. 13-30. http://math.nsc.ru/LBRT/k5/Kochetov/koch-gon.ps

8. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. -М.: Мир, 1982. - 416 с.

9. Еремеев А.В. Генетический алгоритм для задачи о покрытии. Дискретный анализ и исследование операций. Сер. 2. 2000. Т. 7, N 1. С.47-60.

10. Еремеев А.В. Генетические алгоритмы и оптимизация. Учебное пособие. Омск, ОмГУ, 2008.

11. Еремеев А.В. Генетический алгоритм с турнирной селекцией как метод локального поиска. Труды XV Байкальской международной школы-семинара "Методы оптимизации и их приложения"(Иркутск, 23-29 июня 2011 г.). - Иркутск, РИО ИДСТУ СО РАН, Т.4, 2011. - С. 94-99.

12. Еремеев А. В. Исследование эволюционных методов решения задач комбинаторной оптимизации. Диссертация на соискание уч. ст. д.ф.-м.н. по специальности 05.13.17 - «Теоретические основы информатики», 2013, Омск, 300 с. http://iitam.omsk.net.ru/ eremeev/PAPERS.MAT/eremeev.pdf

13. *Еремеев А.В. , Тюнин Н.Н.* Алгоритм дифференциальной эволюции для оптимизации направленности фазированных антенных решеток // Математические структуры и моделирование, 2022, Т. 63, № 3. – С.57-68.

14. А. М. Зубков, Н. Н. Попов. Отношение частичного порядка, порожденное распределениями числа занятых ячеек Матем. заметки, 1982, том 32, выпуск 1, с. 97-102.

15. Ивахненко А. Г. Системы эвристической самоорганизации в технической кибернетике. – Киев: Техника, 1971. – 371 с.

16. Карманов В.Г. Математическое программирование. - М.: Наука, 1986.

17. Кемени Дж., Снелл Дж. Конечные цепи Маркова. – М.: Наука, 1970. – 272 с.

18. Китаев А., Шень А., Вялый М. Классические и квантовые вычисления. - М.: МЦНМО, ЧеРо, 1999. - 192 с.

19. Кочетов Ю.А. Вероятностные методы локального поиска для задач дискретной оптимизации // Дискретная математика и ее приложения: Сборник лекций молодежных научных школ по дискретной математике и ее приложениям. – М.: Изд-во центра прикладных исследований при механико-математическм факультете МГУ, 2001. – С 84-117.// http://math.nsc.ru/LBRT/k5/Kochetov/publ-rus.html

20. Малютина Э.Э. Разработка и применение генетических алгоритмов для анализа поведения сложных динамических систем. Дис. канд. физ.-мат. наук. - М., 2001. - 120 с.

21. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация. Алгоритмы и сложность. – М.: Мир, 1985.

22. Растригин Л.А. Статистические методы поиска. - М.: Наука, 1968. - 376 с.

23. Реймерс Н.Ф. Популярный биологический словарь. - М.: Наука, 1990.

24. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы. – М.: Горячая линия – Телеком, 2006.

25. А.В. Спиров, А.В. Еремеев. Модульность в биологической эволюции и эволюционных вычислениях. Успехи современной биологии. 139 (6), 2019, с. 523-539.

26. Фогель Л., Оуэнс А., Уолш М. Искусственный интеллект и эволюционное моделирование. – М.: Мир, 1969. – 230 с.

27. Aarts E. H. L., Korst J. H. M., Laarhoven van P. J. M. Simulated annealing // Local search in combinatorial optimization / ed. by E. Aarts, J.K.Lenstra. Chichester: Wiley, 1997. P. 91–120.

28. Aarts E. H. L., Korst J. H. M., Laarhoven van P. J. M. Simulated annealing // Local search in combinatorial optimization / ed. by E. Aarts, J.K.Lenstra. Chichester: Wiley, 1997. P. 91–120.

29. Aggarwal C.C., Orlin J.B., Tai R.P. An Optimized Crossover for Maximum Independent Set // Operations Research. - 1997. - Vol.45. - P.225-234.

30. Aloise D., Deshpande A., Hansen P., and Popat P. NP-hardness of Euclidean sum-of-squares clustering. Machine Learning, Vol. 75, N 2, 2009, pp. 245–248.

31. Altenberg, L. (1995). The schema theorem and Price's theorem. In D. Whitley and M. D. Vose (Eds.), *Foundations of Genetic Algorithms 3*, pp. 23–49. San Mateo, CA: Morgan Kaufmann.

32. Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. Proceedings of the Second International Conference on Genetic Algorithms, 14-21.

33. Balas E., Niehaus W. Optimized Crossover-Based Genetic Algorithms for the Maximum Cardinality and Maximum Weight Clique Problems // Journ. of Heuristics. - 1998. - Vol. 4, N 4, - P.107-122.

34. Barahona F.: On the computational complexity of Ising spin glass models. Journ. of Physics A, Mathematical and General. – 1982. – Vol. 15. pp. 3241–3253

35. Beasley J.E., Chu P.C. A Genetic Algorithm for the Set Covering Problem // European J. Oper. Res. – 1996. – Vol. 94, N 2. - P.394-404.

36. Borisovsky P. A., Eremeev A. V. On performance estimates for two evolutionary algorithms // Applications of Evolutionary Computing: Proc. of EvoWorkshops 2001 / Ed. by Boers E. J. W. [et al.]. – Berlin: Springer-Verlag, 2001. – P. 161–171. – (Lect. Notes Comput. Sci.; Vol. 2037).

37. Borisovsky P. A., Eremeev A. V. A study on performance of the (1+1)-evolutionary algorithm // Foundations of Genetic Algorithms 7 / Ed. by De Jong K., Poli R., Rowe J. – San Francisco: Morgan Kaufmann, 2003. – P. 271–287.

38. Borisovsky P. A., Eremeev A. V. Comparing Evolutionary Algorithms to the (1+1)-EA. Theoretical Computer Science, 2008. V. 403, N 1, – PP. 33-41.

39. P.A. Borisovsky, A.V. Eremeev, E.B. Grinkevich, S.A. Klokov and N.A. Kosarev. Trading hubs construction in electricity markets using evolutionary algorithms. Pattern Recognition and Image Analysis. 2014, Vol. 24, Issue 2, - P. 270-282.

40. Bovet D.P., Crescenzi P. Introduction to the theory of complexity. – New York: Prentice-Hall, 1993. – 290 p.

41. Cotta, C., Troya, J.M.: Genetic Forma Recombination in Permutation Flowshop Problems. Evolutionary Computation **6** (1), 25–44 (1998)

42. Cramer N.L. A representation for the adaptive generation of simple sequential programs // Proc. of Intern. Conf. on Genetic Algorithms and Their Applications (July 24-26, 1985, Pittsburgh, PA). 1985. pp 183-187.

43. Crooks G.E., Hon G., Chandonia J.-M., and Brenner S.E. WebLogo: A sequence logo generator // Genome Res. – 2004. – Vol. 14. pp. 1188-1190.

44. *Eiben A.E., Aarts E.H.L. and van Hee K. M.* Global convergence of genetic algorithms: A Markov chain analysis // In H.P. Schwefel and R. Manner (editors) Parallel Problem Solving from Nature. – 1991. Springer. – Berlin and Heidelberg. – P. 4-12.

45. D.-C. Dang, A.V. Eremeev, P.K. Lehre. Runtime Analysis of Fitness-Proportionate Selection on Linear Functions. arXiv:1908.08686v1 [cs.NE] https://arxiv.org/abs/1908.08686

46. D. -C. Dang et al., Escaping local optima using crossover with emergent diversity. IEEE Transactions on Evolutionary Computation, vol. 22, no. 3, pp. 484-497, June 2018.

47. Davis L. Job Shop Sheduling with Genetic Algorithms. In: Proceedings of International Conference on Genetic Algorithms and Their Applications, J.J. Grefenstette (ed.), Lawrence Erlbaum Associates, Hillsdale, NJ, 1985. pp. 136-140.

48. Dembski W.A. The chance of the gaps. In: Neil Manson, ed., God and Design: The Teleological Argument and Modern Science (London: Routledge, 2002), 251-274.

49. Dhaeseleer P. Context preserving crossover in genetic programming // Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence. – 1994. – Vol. 1. – P. 256-261.

50. Benjamin Doerr, Daniel Johannsen, Timo Kötzing, Frank Neumann, Madeleine Theile. More effective crossover operators for the all-pairs shortest path problem. Theoretical Computer Science. 2013. Vol. 471. pp. 12-26.

51. Doerr, B., Doerr, C. (2020). Theory of Parameter Control for Discrete Black-Box Optimization: Provable Performance Gains Through Dynamic Parameter Choices. In: Doerr, B., Neumann, F. (eds) Theory of Evolutionary Computation. Natural Computing Series. Springer, Cham. pp 271–321.

52. Dolgui A., Eremeev A., Kolokolov A., Sigaev V. A Genetic Algorithm for the Allocation of Buffer Storage Capacities in a Production Line with Unreliable Machines. Journal of Mathematical Modelling and Algorithms. 1 (2), 2002, pp. 89-104.

53. A.E. Eiben, E.H.L. Aarts, and K.M. van Hee. Global convergence of genetic algorithms: A Markov chain analysis. In H.-P. Schwefel and R. Männer, editors, Parallel problem Solving from Nature, pp. 4-12. Springer, Berlin and Heidelberg, 1991.

54. A.V. Eremeev, "On proportions of fit individuals in population of mutation-based evolutionary algorithm with tournament selection", *Evolutionary Computation*, vol. 26(2), pp. 269-297, 2018.

55. Eremeev A. V., Kolokolov A. A. On some genetic and L-class enumeration algorithms in integer programming // Proc. of the First International Conference on Evolutionary Computation and its Applications. – Moscow: Russian Academy of Sciences, 1996. – P. 297–303.

56. Eremeev, A.V., Kovalenko, J.V.: Experimental evaluation of two approaches to optimal recombination for permutation problems. In: Evolutionary Computation in Combinatorial Optimization, LNCS, vol. 9595, pp. 138–153. Springer International Publishing, Cham (2016)

57. A. Eremeev and A. Spirov. Modeling SELEX for regulatory regions using Royal Road and Royal Staircase fitness functions. Biosystems. vol. 200, February 2021, 104312.

58. Eremeev A.V., Reeves C.R. Evolutionary algorithms in discrete optimisation. Book of abstracts of Discrete Optimization and Operations Research Conference (DAOR-2002). Novosibirsk. pp.40-45.

59. Eremeev A., Zakharova Yu. A restart rule for genetic algorithms based on Schnabel census. Siberian Electronic Mathematical Reports. 2026. Vol. 23, N 1. pp. 554–570.
https://math-semr.ru/sites/math-semr.ru/files/2026-06/p0554-0570.pdf

60. Faizullin R.T. An approximations for genetic algorithms and star's pattern. In Proc. of The First Online Workshop on Soft Computing, Nagoya, 1996. – P. 77-79.

61. Flemming H.C., Wuertz S. Bacteria and archaea on Earth and their abundance in biofilms. Nature Reviews. Microbiology. 17 (4). pp. 247–260.

62. Goldberg D.E., Lingle R. Alleles, Loci and the Traveling Salesman Problem. In: Proceedings of International Conference on Genetic Algorithms and Their Applications, J.J. Grefenstette (ed.), Lawrence Erlbaum Associates, Hillsdale, NJ, 1985. pp. 154-159.

63. Goldberg D.E. Genetic Algorithms in Search, Optimization and Machine Learning. - Reading: Addison Wesley, 1989. - 412 p.

64. Goldberg D., Thierens D. Elitist recombination: an itegrated selection recombination GA // Proc. first IEEE World Congress on Computational Intelligence. – Piscataway, New Jersey: IEEE Service Center, 1994. Vol. 1. P. 508 – 512.

65. Goldberg D. E. and Deb K. A comparative analysis of selection schemes used in genetic algorithms. In Foundations of Genetic Algorithms, pages 69–93. Morgan Kaufmann, 1991.

66. E.Goncharov and Yu.Kochetov, Behavior of a Probabilistic Tabu Search Algorithm for the Multi Stage Uncapacitated Facility Location Problem // Proceedings INFORMS-KORMS, Seoul 2000. `http://math.nsc.ru/LBRT/k5/Kochetov/seoul.doc`

67. Halliday D. and Resnick R. Fundamentals of Physics, 3rd ed. extended. New York: Wiley, 1988, 544.

68. S. Hampson, and D. Kibler, Plateaus and plateau search in Boolean satisfiability problems: When to give up searching and start again. In: Proceedings of the second DIMACS Implementation Challenge “Cliques, Coloring and Satisfiability”, pp. 437–456. American Mathematical Society (1996)

69. Hartshorn A., Chang S. MWISO hubs development, Technical report. Cambridge, MA, LECG, 2003.

70. Harvey S. M., Hartshorn A. P., Bertagnolli D. and Kowalski R. Derivation of a trading hub in the New England Market, Holyoke, NEPOOL Joint CMS/MSS Group, 1999.

71. Hausman E., Fagan R., White D., Takahashi K., Napoleon A. LMP electricity markets: Market operations, market power, and value for consumers. Synapse Energy Economics. 2006.

72. Holland J.H. 1975. Adaptation in Natural and Artificial systems, University of Michigan Press, Ann Arbor, MI.

73. Howard R. Classifying a population into homogeneous groups. In: Lawrence J.R. (ed.) Operational research in the social sciences. London, Tavistock Publ. 1966.

74. S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi. Optimization by Simulated Annealing. Science, 1983, Vol. 220, Issue 4598, pp. 671-680.

75. Koza J.R. Genetic programming. MIT Press, 1992.

76. Lehre P.K., Witt C. Black-Box Search by Unbiased Variation. Algorithmica (2012) Vol. 64, pp. 623–642.

77. MacQueen J. Some methods for classification and analysis of multivariate observations. In: Lucien Le Cam, M., Neyman, J. (eds.) 5-th Berkeley Symposium on Mathematical Statistics and Probability, Vol. 2, Berkeley, Calif., University of California Press, 1967, pp. 281–297.

78. Langdon W.B. Size fair and homologous tree crossovers for tree genetic programming // Genetic Programming and Evolvable Machines. – 2000. – Vol. 1. P. 95-119.

79. Maulik U., Bandyopadhyay S. Genetic Algorithm-Based Clustering Technique // Pattern Recognition. 2000. Vol. 33, No. 9. P. 1455–1465.

80. Moraglio A., Krawiec K., Johnson C.G. Geometric semantic genetic programming // Proceedings of International Conference on Parallel Problem Solving from Nature – PPSN XII. – 2012. – LNCS, vol. 7491. – P. 21-31.

81. Nix A., Vose M. D. Modeling genetic algorithms with Markov chains // Annals of Math. and Artificial Intelligence. – 1992. – Vol. 5. – P. 79–88.

82. P. S. Oliveto and C. Witt, On the runtime analysis of the simple genetic algorithm, Theor. Comput. Sci., vol. 545, pp. 2–19, 2014.

83. P. S. Oliveto and C. Witt, "Improved time complexity analysis of the simple genetic algorithm," Theor. Comput. Sci., vol. 605, pp. 21–41, 2015.

84. Poli R., Page J. Solving high-order Boolean parity problems with smooth uniform crossover, sub-machine code GP and demes // Genetic Programming and Evolvable Machines. – 2000. Vol. 1. P. 37-56.

85. Pirlot M. General local search methods. EJOR, v. 92, 1996, pp. 493-511.

86. Pradella S., Hans A., Sproer C., Reichenbach H., Gerth K., Beyer S. Characterisation, genome size and genetic manipulation of the myxobacterium Sorangium cellulosum So ce56. Archives of Microbiology. 178 (6). pp. 484–492.

87. Rechenberg I. 1973. Evolutionsstrategie:Optimerung Technischer Systeme nach Prinzipen der Biologischen Evolution, Formann-Holzboog Verlag, Stuttgart.

88. Radcliffe N. J. Forma analysis and random respectful recombination // Proc. of the Fourth International Conference on Genetic Algorithms. – San Francisco, CA: Morgan Kaufmann, 1991. – P. 31–38.

89. Radcliffe, N. J., The Algebra of Genetic Algorithms, Annals of Maths and Artificial Intelligence, 10, 1994.

90. Radcliffe, N.J., Surry, P.D., Fitness Variance of Formae and Performance Prediction, in "Foundations of Genetic Algorithms III", (Ed: L.D. Whitley and M.D. Vose, Morgan Kaufmann), 1994.

91. Rastrigin L.A. 1996. Random Search in Evolutionary Computations. In Proceedings of the First International Conference on Evolutionary Computation and Its Applications, Moscow, 135-142.

92. Reeves C.R. Genetic Algorithms for the Operations Researcher// INFORMS Journal on Computing. - 1997. - Vol. 9, N 3. - P.231-250.

93. Reeves C.R. Genetic Algorithms: No panacea, but a Valuable Tool for the Operations Researcher. INFORMS Journal on Computing. Vol. 9, N.3, 263-265.

94. Reeves C. R., Rowe J. E. Genetic algorithms: principles and perspectives. – Norwell, MA: Kluwer Acad. Pbs., 2002. – 333 p.

95. Rudolph, G. "Finite Markov Chain Results in Evolutionary Computation: A Tour d'Horizon"// Fundamenta Informaticae. Vol. 35. N 1-4. 1998. -pp. 67-89.

96. Smith A. E. and Tate D. M. Genetic optimization using a penalty function. In Forrest S. (ed.) Proceedings of the 5th International Conference on Genetic Algorithms. Morgan Kaufmann Publishers, San Mateo, CA, 1993., pages 499-505.

97. Vose M. D. Modeling simple genetic algorithms // Evolutionary Comput. – 1995. – Vol. 3. – N 4. – P. 453–472.

98. Vose M. D., Wright A. H. Stability of vertex fixed points and applications // Foundations of Genetic Algorithms 3 / Ed. by Whitley D., Vose M. – San Mateo, CA: Morgan Kaufmann, 1995. – P. 103–114.