МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «Воронежский государственный
технический университет»

С.Ю. Белецкая

МЕТОДЫ ОПТИМИЗАЦИИ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

Утверждено учебно-методическим советом
университета в качестве учебного пособия

Воронеж 2017

УДК 681.3
ББК 32.965
Б 431

Белецкая С.Ю. Методы оптимизации в автоматизированных системах: учеб. пособие / С.Ю. Белецкая. – Воронеж: ФГБОУ ВО «Воронежский государственный технический университет», 2017. – 154 с.

В учебном пособии рассматриваются основные классы задач оптимизации и методы их решения, использующиеся в автоматизированных системах

Издание соответствует требованиям Федерального государственного стандарта высшего образования по направлениям 09.03.01 «Информатика и вычислительная техника», 09.03.02 «Информационные системы и технологии» 09.04.01 «Информатика и вычислительная техника»

Табл. 11. Ил. 26. Библиогр.: 8 назв.

Рецензенты: кафедра вычислительной математики
и прикладных информационных технологий
Воронежского государственного
университета (зав. кафедрой д-р техн. наук,
проф. Т.М. Леденева);
д-р техн. наук, проф. О.Ю. Макаров

ВВЕДЕНИЕ

Проблемы, связанные с поиском оптимальных вариантов, занимают важное место при решении практических задач проектирования и управления в автоматизированных системах. Поэтому важным является изучение моделей и методов оптимального выбора, а также рассмотрение их прикладных аспектов.

В учебном пособии последовательно рассматриваются методы и алгоритмы решения задач оптимизации различных классов. Особое внимание уделяется рассмотрению основных приемов построения математических моделей оптимизационных задач. Обсуждаются прикладные оптимизационные модели, использующиеся в автоматизированных системах.

Пособие состоит из восьми глав. Первая глава посвящена вопросам формализации и классификации оптимизационных задач. Во второй главе рассмотрены основные методы одномерной оптимизации. Третья глава ориентирована на рассмотрение методов линейной оптимизации. В четвертой главе дается изложение особенностей задач нелинейной оптимизации и методов их решения. Пятая глава посвящена рассмотрению алгоритмов решения задач дискретной оптимизации. В шестой главе излагаются основы построения генетических алгоритмов. В седьмой главе кратко обсуждаются базовые подходы к решению задач многокритериальной оптимизации. В восьмой главе рассматриваются прикладные оптимизационные модели, использующиеся при поиске оптимальных проектных и управленческих решений в автоматизированных системах.

Представленный теоретический материал иллюстрируется примерами. Алгоритмы, рассмотренные в пособии, могут быть доведены до программной реализации. Это позволяет использовать учебное пособие не только в лекционных курсах, но и в курсовом и дипломном проектировании.

1. ФОРМАЛИЗАЦИЯ ЗАДАЧ ПОИСКА ОПТИМАЛЬНЫХ РЕШЕНИЙ

1.1. Математическое описание и классификация задач оптимизации

Решение широкого круга задач автоматизированного проектирования и управления связано с принятием оптимальных решений, что требует использования соответствующих процедур оптимизации. Под *оптимизацией* понимают процедуру определения в соответствии с установленными критериями наилучшего (оптимального) варианта из множества допустимых вариантов. При этом объектами оптимизации могут являться различные технические, производственные, социально-экономические, информационно-телекоммуникационные системы, транспортные и вычислительные сети, механические конструкции, информационные потоки, маршруты и. т. д. Будем в дальнейшем называть их просто *объектами,* оптимальный вариант которых необходимо определить.

Каждый объект характеризуется набором параметров. При этом различают *внешние*, *внутренние* и *выходные* параметры.

Выходные параметры – величины, характеризующие свойства объекта и определяющие степень выполнения им своего функционального назначения (мощность, коэффициент усиления, производительность, надежность, стоимость и т. д.).

Внешние параметры – величины, характеризующие свойства внешней по отношению к объекту среды и оказывающие влияние на его функционирование (температура окружающей среды, параметры входных сигналов, напряжение источников питания и т. д.).

Внутренние параметры – величины, характеризующие свойства отдельных элементов объекта (геометрические размеры, электрические параметры и т. д.).

Например, для вычислительной системы выходные параметры – производительность системы, коэффициенты за-

грузки оборудования, время решения поступающих задач, вероятность отказа в обслуживании и т. д. ; внутренние параметры – характеристики оборудования (емкости запоминающих устройств, быстродействие процессоров, объем оперативной памяти); внешние параметры – интенсивность поступления заявок на обслуживание.

Совокупность внутренних и внешних параметров называется *входными* параметрами объекта. При этом входные параметры играют роль независимых переменных, а выходные параметры являются зависимыми от них величинами. Входные параметры делятся на *управляемые* (варьируемые) и *неуправляемые*. Варьируемые параметры в процессе оптимизации могут изменяться, что вызывает, в свою очередь, изменение выходных параметров. Варьируемые параметры в зависимости от решаемой задачи могут иметь различную интерпретацию. В дальнейшем будем говорить только о варьируемых параметрах объекта и обозначать их следующим образом:

$$X = (x_1,..., x_n).$$

Различают задачи *параметрической* и *структурной* оптимизации. *Структурная оптимизация* связана с определением оптимальной структуры объекта (т. е. входящих в него элементов и их взаимосвязей). При *параметрической оптимизации* определяются значения параметров объекта заданной структуры, при которых достигается наилучшее сочетание его свойств. Например, при структурной оптимизации вычислительной системы осуществляется выбор количества и типов оборудования, определение топологии сети и т. д. Параметрическая оптимизация может быть связана с определением оптимальной интенсивности поступления задач на обслуживание. В этом случае соответствующие интенсивности будут являться варьируемыми параметрами при оптимизации.

Решение задачи оптимизации начинается с формирования математической оптимизационной модели. Обобщенно такая модель может быть записана следующим образом:

$$
\begin{aligned}
& f(X) \to \min(\max) \\
& g_i(X) \geq (\leq) b_i, \quad i = \overline{1,m} \\
& h_k(X) = d_k, \quad k = \overline{1,s} \\
& x_j^{\min} \leq x_j \leq x_j^{\max}, \quad j = \overline{1,n}
\end{aligned}
\tag{1.1}
$$

Здесь $X = (x_1,..., x_n)$ – вектор варьируемых параметров объекта; f(X) – *целевая функция*, т. е. функция, характеризующая наиболее важные и существенные свойства объекта и являющаяся критерием оценки качества каждого из вариантов. Целевая функция в оптимизационной модели может стремиться к максимуму или к минимуму. Ее еще называют *критерием оптимальности*, *критерием эффективности*, *показателем качества объекта* и т. д. На основании вычисления значений целевой функции f(X) анализируются и сравниваются между собой различные варианты объектов.

В процессе оптимизации объект должен удовлетворять различным требованиям (конструктивным, техническим, экономическим), которые формализуются в виде системы *ограничений*. Ограничения на варьируемые параметры $x_j^{\min} \leq x_j \leq x_j^{\max}$ называют *прямыми ограничениями*. Здесь $x_j^{\min}$ и $x_j^{\max}$ – нижние и верхние допустимые значения для управляемого параметра x_j, которые зависят от решаемой оптимизационной задачи.

Ограничения вида $g_i(X) \geq (\leq) b_i$, $h_k(X) = d_k$ называют *функциональными ограничениями*. Функциональные ограничения являются какими-либо функциями от варьируемых параметров и формулируются в виде системы равенств и неравенств.

Совокупность функциональных и прямых ограничений образует *допустимую область* или *область допустимых решений* D.

Таким образом, решение задачи оптимизации сводится к определению значений управляемых параметров $X = (x_1,..., x_n)$, обеспечивающих оптимальное значение (максимум или минимум) целевой функции f(X) и удовлетворяющих системе ограничений. Такой вектор $X = (x_1,..., x_n)$ называется *оптимальным решением*. Задачи определения оптимальных значений параметров на допустимом множестве D называются еще задачами *математического программирования.*

Рассмотренная математическая оптимизационная модель является обобщенной и конкретизируется при решении практических задач оптимизации. При этом в зависимости от вида оптимизационной модели задачи оптимизации и методы их решения делятся на классы.

Классификация задач оптимизации

1. В зависимости от числа управляемых параметров различают задачи *одномерной* и *многомерной* оптимизации. В задачах одномерной оптимизации имеется один варьируемый параметр, а в задачах многомерной оптимизации – несколько параметров.

2. По характеру искомого оптимума различают задачи *локальной* и *глобальной* оптимизации (унимодальные и многоэкстремальные задачи).

3. В зависимости от наличия ограничений выделяют задачи *безусловной* и *условной* оптимизации. В задачах безусловной оптимизации ограничения отсутствуют, и варьируемые параметры могут изменяться в любых пределах.

4. В зависимости от количества критериев оптимальности различают задачи *однокритериальной* (скалярной) и *многокритериальной* (векторной) оптимизации.

5. По виду целевой функции и ограничений различают задачи *линейной* и *нелинейной* оптимизации. В задачах линейной оптимизации целевая функция и все функции-ограничения ли-

нейны. Если хотя бы одна из функций является нелинейной, задача относится к классу задач нелинейной оптимизации.

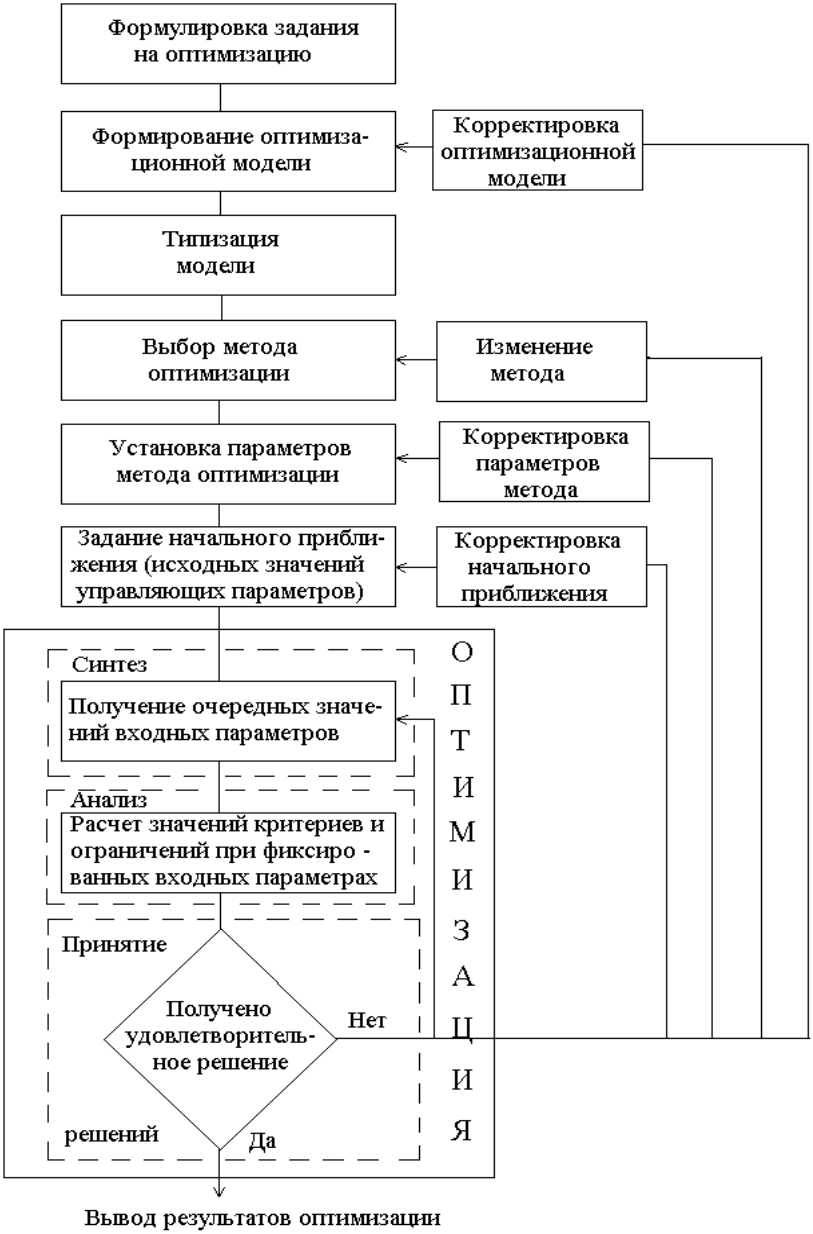
6. По характеру изменения варьируемых параметров различают задачи *непрерывной* и *дискретной* оптимизации. В задачах непрерывной оптимизации параметры изменяются непрерывно в пределах, установленных функциональными ограничениями. В задачах дискретной оптимизации варьируемые параметры принимают дискретные значения.

Частными случаями задач дискретной оптимизации являются задачи *целочисленной* и *булевой* оптимизации. В задачах целочисленной оптимизации варьируемые параметры могут принимать только целочисленные значения. В задачах булевой оптимизации $x_j \in \{0,1\}$. Если D – дискретное конечное множество точек, то такая дискретная задача называется *комбинаторной*. Если при оптимизации часть параметров дискретна, а часть имеет непрерывный характер, то рассматривается задача *частично дискретной* (непрерывно-дискретной) оптимизации.

Необходимо заметить, что одна и та же оптимизационная задача может относиться к нескольким типам (например, являться однокритериальной, линейной и непрерывной). Для различных типов оптимизационных задач разработаны соответствующие методы их решения, классификация которых является аналогичной. При этом данные методы, как правило, инвариантны к предметной области их приложения.

1.2. Обобщенная схема поиска оптимальных решений

Несмотря на то, что задачи оптимизации решаются в разных предметных областях и характеризуются многообразием постановок и методов решения, можно выделить некоторую обобщенную последовательность типовых этапов поиска оптимальных вариантов (рисунок).



Обобщенная схема оптимизационного процесса

На этапе формирования задания на оптимизацию осуществляется содержательная постановка задачи, т. е. определяется набор варьируемых параметров и требования к ним, выбираются критерии оптимальности. Исходной информацией при этом являются данные о назначении, условиях применения и режимах функционирования объекта оптимизации, а также его описание. Эти данные позволяют определить цели оптимизации и в дальнейшем формализовать технические требования, предъявляемые к объекту, в виде оптимизационной модели.

Формирование оптимизационной модели связано с формализацией содержательной постановки задачи, т. е. с ее математической записью в форме (1.1). При этом необходимо определить взаимосвязи между входными и выходными параметрами и сформулировать критерии оптимальности и функциональные ограничения в аналитической или алгоритмической форме. Этот этап связан также с нормированием (нормализацией) варьируемых параметров, если они имеют различную физическую размерность. При этом различают линейную и логарифмическую нормализацию:

$$x_i^{н} = x_i / \xi_i \quad \text{(линейная нормализация)},$$

$$x_i^{н} = \ln(x_i / \xi_i) \quad \text{(логарифмическая нормализация)},$$

где ξ_i – константа, равная единице измерения параметра x_i.

Типизация модели связана с ее преобразованием и приведением к типовому виду, допускающему использование стандартных оптимизационных процедур.

Одним из важнейших этапов поиска оптимальных вариантов является рациональный выбор метода оптимизации. Как уже было отмечено в п. 1.1, задачи оптимизации делятся классы, для каждого из которых разработаны свои методы и алгоритмы решения. Отнесение задачи к определенному классу и определение соответствующего класса методов осуществляется на основании оценки свойств задачи (числа кри-

териев и ограничений, характера изменения параметров и т. д.) и, как правило, сложности не представляет. Однако в рамках каждого класса методов существуют множество различных оптимизационных процедур, выбор которых является достаточно сложной проблемой, так как один и тот же алгоритм при решении различных задач может показывать разную эффективность. Поэтому на практике часто используется многометодный режим решения задач оптимизации.

Далее в зависимости от выбранного метода оптимизации определяются его параметры, а также выбирается начальное приближение. От рационального выбора этих величин также во многом зависят результат и трудоемкость решения оптимизационной задачи.

Сам процесс оптимизации представляет собой итерационную последовательность процедур синтеза, анализа и принятия решений. На этапе синтеза в соответствии с выбранным алгоритмом определяются очередные значения варьируемых параметров (т. е. синтезируется очередной вариант сочетания значений параметров). Этап анализа связан с оценкой полученного варианта, т. е. с вычислением значений критериев и ограничений, соответствующих полученным значениям варьируемых параметров. На этапе принятия решений определяется, является ли данный вариант удовлетворительным. Если получен приемлемый вариант, процесс оптимизации заканчивается, в противном случае – продолжается, т. е. осуществляется переход к следующей итерации. Обычно в вычислительных процедурах оптимизации проверка качества получаемого варианта осуществляется автоматически. При этом на начальном этапе оптимизации определяется требуемая точность вычислений, при достижении которой алгоритм заканчивает работу.

Если при завершении оптимизационного процесса полученный вариант не устраивает проектировщика, можно выполнить следующие действия:

- выбрать другой метод оптимизации;
- скорректировать оптимизационную модель.

После внесения соответствующих изменений оптимизационный процесс продолжается до получения оптимального результата.

Необходимо заметить, что все этапы рассмотренного итерационного процесса, кроме непосредственно оптимизации, выполняются проектировщиком. Таким образом, решение каждой практической задачи, связанной с поиском оптимальных вариантов, требует индивидуального подхода и во многом зависит от опыта и интуиции проектировщика. Поэтому при разработке современных систем оптимизации большое внимание уделяется вопросам принятия оптимальных решений в интерактивном режиме, когда пользователь имеет возможность оперативно взаимодействовать с ЭВМ на любом этапе решения своей задачи.

2. МЕТОДЫ ОДНОМЕРНОЙ ОПТИМИЗАЦИИ

2.1. Метод Ньютона

Метод Ньютона является одним из самых эффективных методов одномерной оптимизации.

Рассмотрим задачу безусловной оптимизации с одним варьируемым параметром x:

$$f(x) \to \min_{x \in R}.$$

Метод Ньютона основан на численном решении уравнения:

$$\varphi(x) = f'(x) = 0. \qquad (2.1)$$

Разложение функции $\varphi(x)$ в окрестности некоторой точки x_k в ряд Тейлора позволяет переписать уравнение (2.1) в виде:

$$\varphi(x) = \varphi(x_k) + (x - x_k)\varphi'(x_k) + o(|x - x_k|) = 0. \qquad (2.2)$$

Отбрасывая в (2.2) малые высшего порядка, получим линейное уравнение относительно неизвестной величины x, решением которого будет

$$x = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)}.$$

Значение аргумента x, определяемое по этой формуле, дает новое приближение корня уравнения (2.1). Таким образом, метод Ньютона отыскания решения уравнения (2.1) описывается следующей итерационной схемой:

$$x_{k+1} = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)} \quad \text{или} \quad x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}. \qquad (2.3)$$

Метод Ньютона называют также *методом касательных* из-за его геометрической интерпретации (рис. 2.1).

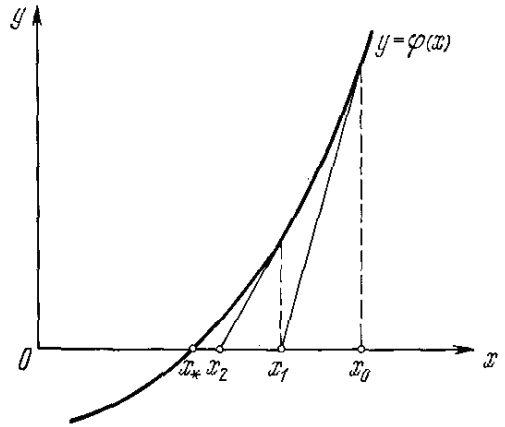


Рис. 2.1. Геометрическая интерпретация метода Ньютона

Проведем касательную к кривой $\varphi(x)$ в точке x_k. Точка пересечения этой касательной с осью абсцисс x_{k+1}, определяется по формуле (2.3). Таким образом, метод Ньютона для решения уравнения $\varphi(x)=0$ состоит в последовательном построении точек пересечения касательных к кривой $\varphi(x)$ с осью абсцисс.

Итерации могут быть продолжены до выполнения условия: $|x_{k+1}-x_k|<\varepsilon$, где ε - требуемая точность.

Основные шаги алгоритма :

1. Задать начальное значение x_0 и точность ε. Положить k=0.

2. Найти значения первой и второй производных целевой функции в точке x_k: $f'(x_k)$ и $f''(x_k)$.

3. Определить: $x_{k+1}=x_k-\dfrac{f'(x_k)}{f''(x_k)}$.

4. Если $|x_{k+1}-x_k|<\varepsilon$, работа алгоритма заканчивается. В противном случае положить k=k+1 и перейти к шагу 2.

2.2. Метод дихотомии

Пусть известен интервал [a,b], содержащий точку минимума, который называется *интервалом неопределенности*. В методе дихотомии этот интервал делится пополам и определяются две точки испытаний x_1 и x_2 по формулам:

$$x_1 = c - \frac{\delta}{2} \quad \text{и} \quad x_2 = c + \frac{\delta}{2}.$$

Здесь c – середина отрезка [a;b]: $c = \frac{a+b}{2}$; δ – минимально допустимый сдвиг ($\delta>0$) между точками испытаний x_1 и x_2, при котором можно точно определить знак разности $(f(x_1) - f(x_2))$. При этом параметр δ имеет такое значение, что если $|x_1 - x_2| \geq \delta$, то $f(x_1) = f(x_2)$ только в том случае, когда x_1 и x_2 находятся по разные стороны от точки локального минимума x^*.

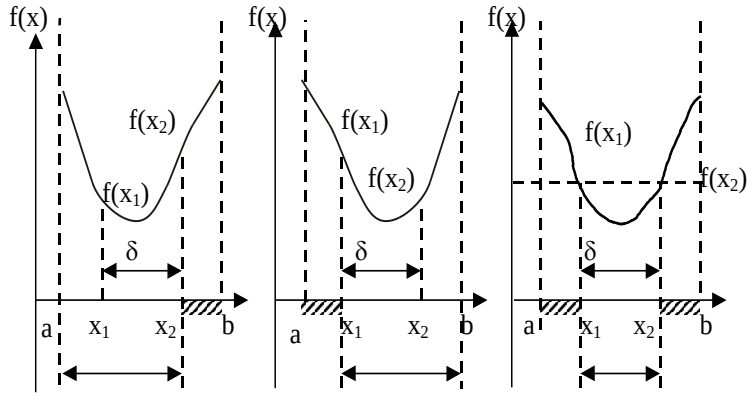
Следовательно, точки x_1 и x_2 располагаются симметрично относительно середины априорного интервала неопределенности [a;b]. Из свойства унимодальности минимизируемой функции f(x) после отбрасывания подынтервала, где точка локального минимума x^* отсутствует, для дальнейшего рассмотрения остается один из текущих интервалов неопределенности $\left[a; c + \frac{\delta}{2}\right]$ или $\left[c - \frac{\delta}{2}; b\right]$, длина которого определяется выражением : $L = \frac{b-a}{2} + \frac{\delta}{2}$.

При этом возможна одна из следующих ситуаций:

а) если $f(x_1) < f(x_2)$ (рис. 2.2, а), то отрезком локализации точки x^* является подынтервал $[a;x_2]$;

б) если $f(x_1) > f(x_2)$ (рис. 2.2, б), то отрезком локализации точки x^* является подынтервал $[x_1;b]$;

в) если $f(x_1) = f(x_2)$ (рис. 2.2, в), то отрезком локализации точки x^* является подынтервал $[x_1;x_2]$.



а) б) в)

Рис. 2.2

Следующая пара испытаний, разнесенная между собой на величину минимально допустимого сдвига δ, проводится симметрично относительно середины текущего интервала неопределенности, полученного на предыдущем шаге.

Процесс поиска заканчивается, если $|x_{k+1} - x_k| < \varepsilon$, где ε - требуемая точность вычисления оптимума, которая является входным параметром алгоритма.

Основные шаги алгоритма :

1. Задать интервал [a;b], минимально допустимый сдвиг δ>0, точность ε.

2. Определить $x_1 = \frac{a+b-\delta}{2}$ и $x_2 = \frac{a+b+\delta}{2}$.

3. Вычислить значения функции $f_1 = f(x_1)$ и $f_2 = f(x_2)$.

4. Если $f_1 < f_2$, то $b = x_2$;

если $f_1 > f_2$, то $a = x_1$;

если $f_1 = f_2$, то $a = x_1$ и $b = x_2$.

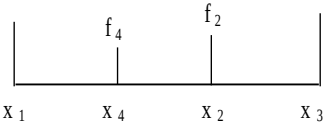
5. Если $|b-a|<\varepsilon$, работа алгоритма заканчивается. В противном случае необходимо вернуться к шагу 2.

2.3. Метод Фибоначчи

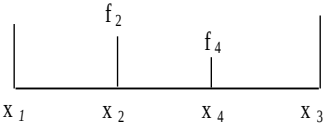
Предположим, что имеется интервал неопределенности $[x_1;x_3]$ и известна точка x_2 внутри этого интервала. В методе Фибоначчи следующая точка x_4 помещается внутри текущего интервала симметрично относительно уже находящейся там точке. Затем одна из частей интервала неопределенности отбрасывается.

При этом если $f(x_2)=f_2$ и $f(x_4)=f_4$, то можно рассмотреть четыре случая (рис. 2.3):

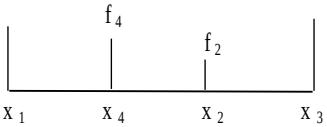
а) $x_4 < x_2$
$f_4 < f_2$
новый интервал (x_1, x_2)
содержащий точку x_4



б) $x_4 > x_2$
$f_4 < f_2$
новый интервал (x_2, x_3)
содержащий точку x_4



в) $x_4 < x_2$
$f_4 > f_2$
новый интервал (x_4, x_3)
содержащий точку x_2



г) $x_4 > x_2$
$f_4 > f_2$
новый интервал (x_1, x_4)
содержащий точку x_2

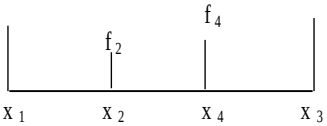


Рис. 2.3

Таким образом, для работы метода Фибоначчи необходимо найти положение первой точки, после чего итерационная процедура может быть продолжена с использованием рассмотренного правила симметрии.

Пусть L_1 – длина начального интервала неопределенности. При этом первая точка помещается на расстоянии L_2 от одного из концов начального интервала. Величина L_2 определяется следующим образом:

$$L_2 = \frac{F_{n-1}}{F_n} L_1 + \frac{(-1)^n \varepsilon}{F_n}. \qquad (2.4)$$

Здесь n – число испытаний, которое задается перед началом работы алгоритма; F_n и F_{n-1} – числа Фибоначчи. Последовательность чисел Фибоначчи определяется следующим образом:

$$F_0 = 1,\ F_1 = 1,\ F_k = F_{k-1} + F_{k-2} \qquad k=2, \ldots,n$$

Таким образом, каждое последующее число Фибоначчи, начиная с третьего, определяется как сумма двух предыдущих: 1,1,2,3,5,8,…

После того как найдено положение первой точки, числа Фибоначчи больше не нужны. Используемое значение параметра ε задается пользователем.

Основные шаги алгоритма :

1. Задать число испытаний n, параметр ε и начальный интервал неопределенности [a;b]. Положить x_1=a и x_3=b.

2. Найти положение первой точки по правилу:

$$x_2 = a + \frac{F_{n-1}}{F_n}(b-a) + \frac{(-1)^n \varepsilon}{F_n}.$$

3. Точку x_4 поместить по правилу симметрии на расстоянии x_2 от другого конца отрезка: $x_4 = x_1 - x_2 + x_3$.

4. Вычислить значения функции $f(x_2) = f_2$ и $f(x_4) = f_4$.

5. Установить новые границы интервала неопределенности:

если $x_4<x_2$ и $f_4<f_2$, то $x_3=x_2$, $x_2=x_4$, $f_2=f_4$;
если $x_4>x_2$ и $f_4<f_2$, то $x_1=x_2$, $x_2=x_4$, $f_2=f_4$;
если $x_4<x_2$ и $f_4>f_2$, то $x_1=x_4$;
если $x_4>x_2$ и $f_4>f_2$, то $x_3=x_4$.

6. Вернуться к шагу 3. Выполнять до тех пор, пока число испытаний не станет равным n.

2.4. Метод «золотого сечения»

Не всегда можно заранее определить, сколько раз придется вычислять функцию. В методе Фибоначчи это нужно знать для определения L_2, т.е. положения начальной точки.

В методе «золотого сечения» не требуется знать n – количество вычислений функции, определяемое вначале. После того как выполнено j вычислений, исходя из тех же соображений, что и ранее, записываем $L_{j-1} = L_j + L_{j+1}$. Однако если n неизвестно, то мы не можем использовать условие $L_{n-1} = 2L_n - \varepsilon$. Если отношение последующих интервалов будет постоянным, т.е.

$$\frac{L_{j-1}}{L_j} = \frac{L_j}{L_{j+1}} = \frac{L_{j+1}}{L_{j+2}} = \ldots = \tau, \qquad (2.5)$$

то $\frac{L_{j-1}}{L_j} = 1 + \frac{L_{j+1}}{L_j}$, т.е. $\tau = 1 + 1/\tau$.

Таким образом, $\tau^2 - \tau - 1 = 0$.

Получим $\tau = (1+\sqrt{5})/2 \approx 1.618033989$. Тогда

$\frac{L_{j-1}}{L_{j+1}} = \tau^2$, $\frac{L_{j-2}}{L_{j+1}} = \tau^3$ и т.д. Следовательно, $L_n = \frac{L_1}{\tau^{n-1}}$.

В результате анализа двух рассмотренных значений функций будет определен тот интервал, который должен исследоваться в дальнейшем. Этот интервал будет содержать одну из предыдущих точек и следующую точку, помещаемую симметрично ей. Первая точка находится на расстоянии L_1 / τ от одного конца интервала, вторая – на таком же расстоянии от другого. Поскольку $\lim\limits_{n\to\infty} \frac{F_{n-1}}{F_n} = \frac{1}{n}$, то поиск методом «золотого сечения» является предельной формой поиска методом Фибоначчи. Название «золотое сечение» произошло от названия отношения в уравнении (2.5). Видно, что L_{j-1} делится на две части так, что отношение целого к большей части равно отношению большей части к меньшей, т.е. равно так называемому «золотому отношению».

Таким образом, если ищется интервал $(x_0;x_3)$ и имеются два значения функции f_1 и f_2 в точках x_1 и x_2, то следует рассмотреть два случая (рис. 2.4).

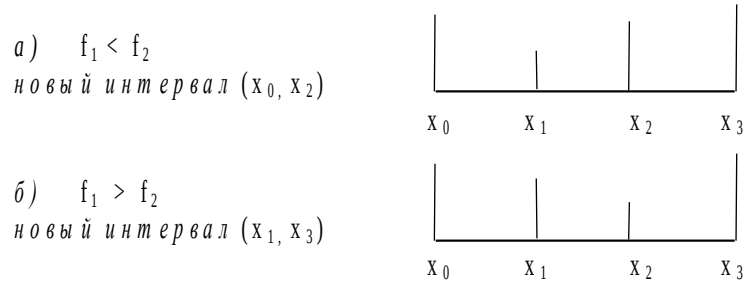


Рис. 2.4

Основные шаги алгоритма:

Шаг 1. Задать интервал (A;B), точность ε.

Шаг 2. Положить x_0=A и x_3=B. Определить первую точку $x_1 = B - \frac{L_1}{\tau}$, где $L_1 = (x_0 - x_3)$; $x_2 = x_0 + x_3 - x_1$.

Шаг 3. Вычислить значения функции в точках x_1 и x_2 – f_1 и f_2.

Шаг 4. Установить новые значения координат точек в зависимости от соотношения f_1 и f_2:

а) если $f_1 < f_2$, то $I = x_2 - x_0$; $x_3 = x_2$; $x_2 = x_1$; $x_1 = x_0 + \frac{I}{\tau}$; $f_2 = f_1$.

б) если $f_1 > f_2$, то $I = x_3 - x_1$; $x_0 = x_1$; $x_1 = x_2$; $x_2 = x_0 + \frac{I}{\tau}$; $f_1 = f_2$.

Шаг 5. Если $I > \varepsilon$, возврат к шагу 3. Иначе, конечным результатом поиска считается точка x_1.

3. ЛИНЕЙНАЯ ОПТИМИЗАЦИЯ

3.1. Форма записи задач линейной оптимизации

Задачи линейной оптимизации заключаются в определении оптимального (максимального или минимального) значения линейной целевой функции при линейных ограничениях. Ограничения в задачах линейной оптимизации представлены в виде системы линейных равенств и неравенств. Задачи линейной оптимизации называются еще задачами *линейного программирования* (ЗЛП). Эту терминологию будем использовать в дальнейшем.

Модель задачи линейного программирования (ЗЛП) может быть записана в одной из следующих форм:

Общая ЗЛП:

$$F = \sum_{j=1}^{n} c_j x_j \to \max(\min),$$

$$\sum_{j=1}^{n} a_{ij} x_j \le (\ge, =) b_i, i = \overline{1,m}, \qquad (3.1)$$

$$x_k \ge 0, k = \overline{1,s},\ s \le n.$$

Здесь $x_1,..., x_n$ – варьируемые параметры модели. Будем предполагать, что переменные x_j неотрицательны.

Стандартная ЗЛП:

$$F = \sum_{j=1}^{n} c_j x_j \to \max(\min),$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, i = \overline{1,m}, \qquad (3.2)$$

$$x_i \ge 0, i = \overline{1,n}$$

Каноническая ЗЛП:

$$F = \sum_{j=1}^{n} c_j x_j \to \min,$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = \overline{1, m}, \qquad (3.3)$$

$$x_i \geq 0, i = \overline{1, n}.$$

Все формы записи ЗЛП являются эквивалентными и могут быть приведены друг к другу с помощью равносильных преобразований.

При разработке вычислительных методов решения ЗЛП особое значение имеет *каноническая форма* задачи (3.3). Перед решением ЗЛП ее необходимо привести к канонической форме. Каноническая форма записи ЗЛП характеризуется следующими условиями:

1. Целевая функция подлежит минимизации.

2. Все переменные должны быть неотрицательны.

3. Правые части всех ограничений задачи должны быть неотрицательны.

4. Все ограничения задачи должны быть записаны в виде равенств.

При приведении ЗЛП к канонической форме используются следующие *основные приемы*:

1. Для перехода от задачи максимизации к задаче минимизации целевую функцию необходимо умножить на (–1).

2. Переменные, на которые не наложено ограничение неотрицательности, представляются в виде разности двух неотрицательных переменных

$$x_j = u_j - w_j, \quad \text{где} \quad u_j, w_j \geq 0.$$

Такая подстановка делается во всех ограничениях, где есть x_j, а также в выражении для целевой функции.

3. Переход к ограничениям с неотрицательными правыми частями осуществляется умножением левой и правой

частей ограничений с отрицательными правыми частями на (–1). При этом знаки соответствующих неравенств меняются на противоположные.

4. Переход от ограничений-неравенств к ограничениям-равенствам осуществляется путем введения дополнительных неотрицательных переменных. При этом, если знак неравенства ≤, дополнительная переменная прибавляется к левой части ограничения, а если ≥, то вычитается. Таким образом, ограничение-неравенство

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \le b_i$$

преобразуется в ограничение-равенство

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n + x_{n+1} = b_i \ (x_{n+1} \ge 0),$$

а ограничение-неравенство

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \ge b_i \ ,$$

в ограничение-равенство

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n - x_{n+1} = b_i \ (x_{n+1} \ge 0).$$

В каждое неравенство вводится своя дополнительная переменная. Число вводимых дополнительных неотрицательных переменных равно числу преобразуемых неравенств.

ЗЛП в векторной форме:

$$F = \sum_{j=1}^{n} c_j x_j \to \min;$$

$$\sum_{j=1}^{n} A_j x_j = B$$

$$x_j \ge 0, \ j = \overline{1,n},$$

где $A_1,\ldots,A_n$ – m-мерные вектор-столбцы, составленные из коэффициентов при переменных в системе ограничений:

$$A_1 = \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix} \quad \cdots \quad A_n = \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{pmatrix},$$

B – вектор-столбец свободных членов системы ограничений:

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}.$$

ЗЛП в матричной форме:

$$\begin{aligned} & C^T X \to \min \\ & AX = B \\ & X \geq 0 \end{aligned},$$

где X – вектор-столбец варьируемых параметров оптимизационной модели:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix},$$

C – вектор-столбец коэффициентов при переменных в целевой функции:

$$C = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}, \qquad C^T = (c_1, ..., c_n),$$

A – матрица, составленная из коэффициентов при переменных в системе ограничений:

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$$

3.2. Решение задач линейной оптимизации симплекс-методом

Рассмотрим ЗЛП в канонической форме:

$$F = \sum_{j=1}^{n} c_j x_j \to \min$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \ i = \overline{1,m}$$

$$x_j \geq 0, \quad j = \overline{1,n}$$

Вектор X = ($x_{1,\dots,}$ x_n), удовлетворяющий системе ограничений ЗЛП, называется *допустимым решением* или *планом.*

План $x^* = (x_1^*, \dots, x_n^*)$, при котором целевая функция принимает оптимальное значение, называется *оптимальным*.

Перепишем ЗЛП в векторной форме:

$$F = \sum_{j=1}^{n} c_j x_j \to \min,$$

$$\sum_{j=1}^{n} A_j x_j = B,$$

$$x_j \geq 0, \quad j = \overline{1,n},$$

где $A_1,\ldots,A_n$ – m-мерные вектор-столбцы, составленные из коэффициентов при переменных в системе ограничений;

$$A_1 = \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix} \quad \cdots \quad A_n = \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{pmatrix},$$

B – вектор-столбец свободных членов системы ограничений:

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}.$$

План $X = (x_{1,\ldots,} x_n)$ называется *опорным планом* ЗЛП, если система векторов A_j, входящих в разложение $\sum_{j=1}^{n} A_j x_j = B$ с положительными коэффициентами x_j , линейно независима. Так как векторы A_j являются m-мерными, то число положительных компонент опорного плана не может быть больше m.

Симплексный метод решения ЗЛП основан на переходе от одного опорного плана к другому, при котором значение целевой функции убывает. Указанный переход возможен, если известен какой-нибудь исходный опорный план. Такой план можно легко указать, если ЗЛП имеет следующий вид:

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n \to \min$$
$$x_1 + a_{1m+1}x_{m+1} + \ldots + a_{1n}x_n = b_1$$
$$x_2 + a_{2m+1}x_{m+1} + \ldots + a_{2n}x_n = b_2$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$x_m + a_{mm+1}x_{m+1} + \ldots + a_{mn}x_n = b_m$$
$$x_j \geq 0, j = \overline{1,n}$$

Векторная форма данной задачи имеет вид:

$$F = \sum_{j=1}^{n} c_j x_j \to \min$$
$$A_1x_1 + A_2x_2 + \ldots + A_{m+1}x_{m+1} + \ldots + A_nx_n = B$$
$$x_j \geq 0, j = \overline{1,n}$$

где

$$A_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix},\ A_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix},\ A_m = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix},\ A_{m+1} = \begin{pmatrix} a_{1m+1} \\ a_{2m+1} \\ \vdots \\ a_{mm+1} \end{pmatrix},\ A_n = \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{pmatrix}$$

Переменные, которые входят только в одно уравнение системы ограничений с коэффициентом 1, называют *базисными переменными* (в рассматриваемом примере это переменные $x_1 \ldots x_m$). Остальные переменные называются *свободными*. Тогда приравнивая базисные переменные соответствующим правым частям ограничений, а свободные переменные нулю, получим опорный план $X = (b_1, b_2, \ldots, b_m, 0, \ldots, 0)$, определяемый системой единичных векторов, $A_1, \ldots, A_n$, которые образуют базис n-мерного пространства.

Для удобства расчетов в симплексном методе все исходные данные задачи заносят в симплекс-таблицу:

Таблица 3.1

			c_1	…	c_n
$x_Б$	$c_Б$	B	x_1	…	x_n
$x_{Б1}$	$c_{Б1}$	b_1	a_{11}	…	a_{1n}
…	…	…	…	…	…
$x_{Бm}$	$c_{Бm}$	b_m	a_{m1}	…	a_{mn}
F			Δ_1	…	Δ_n

В верхней строке таблицы записываются коэффициенты при всех переменных в целевой функции. В первом столбце таблицы записываются базисные переменные в той последовательности, в которой они входят в систему ограничений, во втором – коэффициенты целевой функции при базисных переменных, в третьем – правые части всех ограничений, в последующих столбцах – коэффициенты при соответствующих переменных в системе ограничений. В нижней строке таблицы записываются оценки по каждой переменной, определяемые следующим образом: $\Delta_j = \sum_{i=1}^{m} c_{Бi} a_{ij} - c_j$. Под столбцом правых частей обычно записывается текущее значение целевой функции.

На каждой итерации симплекс-метода осуществляется вывод из базиса какой-либо переменной и включение в него другой переменной с соответствующим пересчетом элементов таблицы. Перед решением задачи ее необходимо привести к канонической форме.

Основные шаги симплекс - метода:

1. Определение начального опорного плана.
2. Составление симплекс-таблицы.

3. Вычисление оценок $\Delta_j = \sum_{i=1}^{m} c_{Бi} a_{ij} - c_j$.

4. Анализ оценок.

4.1. Если $\Delta_j \le 0 \ \forall j = \overline{1,n}$, то получено оптимальное решение.

4.2. Если существует хотя бы одна оценка $\Delta_j > 0$, для которой $a_{ij} < 0 \ \forall j = \overline{1,m}$, то целевая функция не ограничена снизу на множестве допустимых решений (ЗЛП не имеет решения).

4.3. Из всех оценок $\Delta_j > 0$ выбирается максимальная

$$\Delta_k = \max_{j:\Delta j > 0} \Delta_j$$

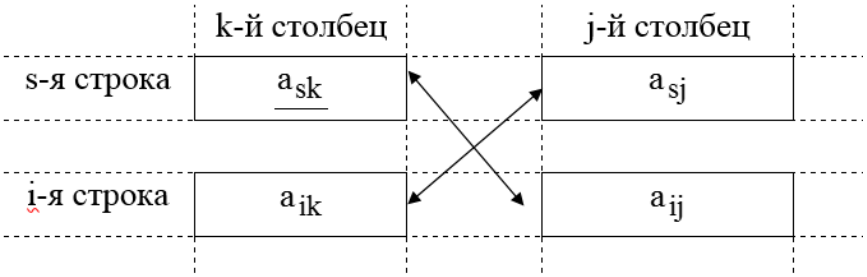
Переменная x_k, которой соответствует максимальная оценка, становится на текущей итерации базисной, а k-й столбец объявляется *ведущим столбцом*.

5. Определение *ведущей строки*. Для этого находятся попарные отношения правых частей ограничений к положительным элементам ведущего столбца и среди них выбирается минимальное:

$$\frac{b_s}{a_{sk}} = \min_{i:a_{ik}>0} \frac{b_i}{a_{ik}}$$

Строка s объявляется ведущей строкой. Элемент a_{sk}, находящийся в симплекс-таблице на пересечении s-й строки и k-го столбца, становится ведущим элементом.

6. Пересчет элементов симплекс-таблицы. При этом элементы ведущей строки $a_{s1},...,a_{sn},b_s$ делятся на ведущий элемент a_{sk}. Пересчет остальных элементов удобно осуществлять по правилу прямоугольника.



Пусть a_{sk} – ведущий элемент, и необходимо пересчитать элемент a_{ij}. Мысленно соединяем эти элементы и достраиваем конструкцию до прямоугольника. Элемент a_{ij} пересчитывается следующим образом:

$$a'_{ij} = \frac{a_{sk}a_{ij} - a_{ik}a_{sj}}{a_{sk}}$$

При этом сначала вычисляется произведение ведущего элемента на пересчитываемый, из него вычитается произведение элементов противоположной диагонали и результат делится на ведущий элемент. По аналогичной схеме пересчитываются и правые части ограничений:

$$b'_i = \frac{a_{sk}b_i - a_{ik}b_s}{a_{sk}}$$

7. Переход к шагу 3.

По окончательной симплекс-таблице оптимальное решение определяется следующим образом: базисные переменные приравниваются соответствующим правым частям ограничений, остальные переменные равны нулю.

Пример. Решить задачу определения оптимального производственного плана.

Требуется определить, в каком количестве необходимо выпускать продукцию четырех типов П1, П2, П3, П4, для из-

готовления которой требуются ресурсы трех видов: трудовые ресурсы, сырье, финансы. Нормы расхода ресурсов каждого вида для выпуска единицы продукции, а также прибыль, получаемая от реализации единицы каждого типа продукции, приведены в табл. 3.2 Количество расходуемых ресурсов не должно превышать имеющихся запасов.

Таблица 3.2

Ресурсы	Виды продукции				Запасы ресурсов
	П1	П2	П3	П4	
Трудовые	3	1	2	4	440
Сырье	1	8	6	2	200
Финансы	1	4	7	2	320
Прибыль	7	3	6	12	

Математическая модель для решения данной задачи будет иметь следующий вид:

$$F=7x_1+3x_2+6x_3+12x_4 \to \max;$$

$$3x_1+x_2+2x_3+4x_4 \le 440;$$

$$x_1+8x_2+6x_3+2x_4 \le 200;$$

$$x_1+4x_2+7x_3+2x_4 \le 320;$$

$$x_j \ge 0,\ j=\overline{1,4}.$$

Здесь x_j – количество продукции $П_j$, которое необходимо выпустить по плану. Целевая функция – общая прибыль от производства всей продукции. Каждое i-е ограничение характеризует общее количество ресурса S_i, которое не должно превышать величины b_i.

Перед решением задачи приведём ее к канонической форме:

$$F=-7x_1-3x_2-6x_3-12x_4\rightarrow\min;$$

$$3x_1+x_2+2x_3+4x_4+x_5=440;$$

$$x_1+8x_2+6x_3+2x_4+x_6=200;$$

$$x_1+4x_2+7x_3+2x_4+x_7=320;$$

$$x_j\geq 0,\ j=\overline{1.7}.$$

Базисными переменными в данном случае будут переменные x_5, x_6, x_7. Решение задачи иллюстрируется симплексной табл. 3.3. Ведущие элементы на каждой итерации в таблице выделяются.

Таблица 3.3

$x_Б$	$c_Б$	B	-7	-3	-6	-12	0	0	0
			x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_5	0	440	3	1	2	4	1	0	0
x_6	0	200	1	8	6	**2**	0	1	0
x_7	0	320	1	4	7	2	0	0	1
		0	7	3	6	12	0	0	0
x_5	0	40	**1**	-15	-10	0	1	-2	0
x_4	-12	100	1/2	4	3	1	0	1/2	0
x_7	0	120	0	-4	1	0	0	-1	1
		-1200	1	-45	-30	0	0	-6	0
x_1	-7	40	1	-15	-10	0	1	-2	0
x_4	-12	80	0	23/2	8	1	-1/2	3/2	0
x_7	0	120	0	-4	1	0	0	-1	1
		-1240	0	-30	-20	0	-1	-4	0

Так как после третьей итерации все оценки $\Delta_j \leq 0$, получено оптимальное решение. Оптимальный план имеет следующий вид:

$x_1^*=40$; $x_2^*=0$; $x_3^*=0$; $x_4^*=80$; $x_5^*=0$; $x_6^*=0$; $x_7^*=120$; $F^*=1240$.

Таким образом, для того, чтобы прибыль от производства всей продукции была максимальна при имеющихся ограниченных ресурсах, необходимо выпустить 40 единиц продукции 1-го вида и 80 единиц продукции 4-го вида. Продукция 2-го и 3-го видов при этом не выпускается. Прибыль при этом равна 1240. Так как значения дополнительных переменных в оптимальном решении $x_5^*=0$ и $x_6^*=0$, то трудовые ресурсы и сырье полностью используются при производстве продукции. Финансовые ресурсы при производстве используются не полностью (имеется запас в 120 единиц).

3.3. Метод искусственного базиса

Для задачи линейной оптимизации, записанной в канонической форме, можно непосредственно указать опорный план, если среди векторов A_j, компонентами которых служат коэффициенты при переменных в системе ограничений данной задачи, имеются m единичных. Однако для многих задач, записанных в канонической форме и имеющих опорные планы, среди векторов A_j не всегда есть m единичных. Рассмотрим такую задачу.

$$F = \sum_{j=1}^{n} c_j x_j \to \min,$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = \overline{1,m},$$

$$x_j \geq 0, j = \overline{1,n}.$$

Предположим, что в данной задаче среди векторов

$$A_{m+1} = \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix},\ A_n = \begin{pmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{pmatrix}$$

нет единичных.

В данном случае для получения первоначального опорного плана в систему ограничений вводятся неотрицательные переменные, называемые *искусственными* переменными. Тогда система ограничений рассматриваемой задачи примет вид:

$$\sum_{j=1}^{n} a_{ij} x_j + x_{n+i} = b_i, i = \overline{1, m},$$

где x_{n+i} – искусственные переменные.

Так как введение искусственных переменных меняет множество решений задачи, они вводятся и в выражение для целевой функции с очень большим коэффициентом $M > 0$:

$$F = \sum_{j=1}^{n} c_j x_j + M \sum_{j=n+1}^{n+m} x_j \to \min$$

Такая задача называется *расширенной* по отношению к исходной. Тогда в процессе решения задачи минимизации искусственные переменные будут стремиться к нулю. Оценки Δ_j в данном случае вычисляются следующим образом:

$$\Delta_j = \sum_{j=1}^{m} c_{Бi} a_{ij} - c_j = M \sum_{i=1}^{m} a_{ij} - c_j$$

Таким образом, оценки можно представить в виде: $\Delta_j = \alpha_j + M\beta_j$. Каждая j-я оценка состоит из двух частей, одна из которых зависит от M, а другая не зависит.

Исходные данные расширенной задачи заносят в симплекс-таблицу, которая содержит на одну строку больше, чем

обычная. В (m+2)-ю строку оценок таблицы записывают коэффициенты при М, а в (m+1)-ю заносят слагаемые, не содержащие М.

Так как знак оценки Δ_j определяется знаком коэффициента, стоящего при М, ведущий столбец и базисную переменную выбирают по (m+2)-й строке таблицы. Пересчет симплекс-таблицы при переходе от одного опорного плана к другому производят по общим правилам симплексного метода. Если в процессе решения задачи какая-либо искусственная переменная выводится из базиса, то соответствующий столбец симплекс-таблицы можно вычеркнуть и не пересчитывать (т. к. эту переменную не имеет смысла вводить ни в один из последующих базисов).

Итерационный процесс по (m+2)-й строке симплекс-таблицы ведут до тех пор, пока:

1) либо все искусственные переменные не будут исключены из базиса;

2) либо не все искусственные переменные исключены, но (m+2)-я строка не содержит больше положительных элементов.

В первом случае базис отвечает некоторому опорному плану исходной задачи и определение ее оптимального плана продолжают по (m+1)-й строке.

Во втором случае, если элемент, стоящий в (m+2)-й строке столбца В положителен, задача не имеет решения, если он равен нулю, то базис содержит по крайней мере один из векторов искусственного базиса.

Таким образом, процесс нахождения решения ЗЛП методом искусственного базиса включает следующие этапы:

1. Составление расширенной задачи.

2. Определение опорного плана расширенной задачи.

3. Вывод искусственных переменных из базиса с использованием симплекс-метода. В результате либо находят опорный план исходной задачи, либо устанавливают ее неразрешимость.

4. Определение оптимального плана исходной задачи с использованием симплекс-метода.

Пример. Решить задачу линейной оптимизации:

$$F=6x_1+x_2+x_3\rightarrow\min;$$

$$2x_1+4x_2+5x_3\le 26;$$

$$2x_1+x_2\ge 8;$$

$$12x_1+3x_2+x_3\le 60;$$

$$x_j\ge 0,\ j=\overline{1,3}.$$

Приведем задачу к канонической форме.

$$F=6x_1+x_2+x_3\rightarrow\min;$$

$$2x_1+4x_2+5x_3+x_4=26;$$

$$2x_1+x_2-x_5=8;$$

$$12x_1+3x_2+x_3+x_6=60;$$

$$x_j\ge 0,\ j=\overline{1,6}.$$

Составим расширенную задачу, вводя искусственную переменную x_7 во второе ограничение. Эту же переменную вводим в целевую функцию с большим коэффициентом М. Расширенная задача имеет вид:

$$F=6x_1+x_2+x_3+Mx_7\rightarrow\min;$$

$$2x_1+4x_2+5x_3+x_4=26;$$

$$2x_1+x_2-x_5+x_7=8;$$

$$12x_1+3x_2+x_3+x_6=60;$$

$$x_j\ge 0,\ j=\overline{1,7}.$$

Решение задачи приведено в таблице 3.4.

Таблица 3.4

$X_б$	$c_б$	B	6	1	1	0	0	0	M
			x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_4	0	26	2	4	5	1	0	0	0
X_6	0	8	**2**	1	0	0	-1	0	1
X_7	M	60	12	3	1	0	0	1	0
		0	-5	-1	-1	0	0	0	0
		60	12	3	1	0	0	0	0

$x_б$	$c_б$	B	6	1	1	0	0	0
			x_1	x_2	x_3	x_4	x_5	x_6
x_4	0	18	0	**3**	5	1	1	0
x_1	6	4	1	1/2	0	0	-1/2	0
x_6	0	12	0	-3	1	0	6	1
		20	0	2	-1	0	-3	0

$x_б$	$c_б$	b	6	1	1	0	0	0
			x_1	x_2	x_3	X_4	x_5	x_6
x_2	1	6	0	1	5/3	1/3	1/3	0
x_1	6	1	1	0	-5/6	-1/6	-2/3	0
x_4	0	30	0	0	6	1	7	1
		12	0	0	-10/3	-2/3	-11/3	0

В результате решения получен оптимальный план:

$x_1^*=1$; $x_2^*=6$; $x_3^*=0$; $x_4^*=30$; $x_5^*=0$; $x_6^*=11$; $F(X^{*)}=12$.

3.3. Двойственный симплекс-метод

Двойственный симплекс-метод используется при нахождении решения ЗЛП, записанной в канонической форме, для которой среди векторов A_j , составленных из коэффициентов при неизвестных в системе ограничений, имеются m единичных. Вместе с тем этот метод можно применять при решении ЗЛП, правые части ограничений которой могут быть отрицательными числами.

Рассмотрим ЗЛП, записанную в канонической форме и предположим, что выбран такой базис $(A_1,...,A_m)$, при котором хотя бы одна из правых частей ограничений отрицательна, но для всех переменных x_j оценки $\Delta_j \le 0$.

$$F = \sum_{j=1}^{n} c_j x_j \to \min$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = \overline{1,m}$$

$$x_j \ge 0, j = \overline{1,n}.$$

Тогда решение $X=(b_1,\ldots,b_m,0,\ldots,0)$ системы линейных алгебраических уравнений, задающей ограничения, называется *псевдопланом* ЗЛП. Псевдоплан не является планом исходной ЗЛП, т. к. некоторые его компоненты отрицательны.

Очевидно, что для получения оптимального плана исходной ЗЛП необходимо исключить из правых частей ограничений симплекс-таблицы все отрицательные элементы. При этом должно выполняться условие

$$\Delta_j \le 0 \quad \forall j = \overline{1,n}.$$

Основные шаги двойственного симплекс-метода.

1. Нахождение псевдоплана задачи.
2. Составление симплекс- таблицы.

3. Проверка псевдоплана на оптимальность. Если псевдоплан оптимален (все правые части $b_i \geq 0$, а все оценки $\Delta_j \leq 0$), решение задачи найдено. В противном случае необходимо перейти к следующему шагу.

4. Выбор ведущей строки. При этом в столбце правых частей ограничений выбирается наименьший отрицательный элемент, и строка, которой он соответствует, объявляется ведущей строкой.

$$b_s = \min b_i \quad \forall i : b_i < 0$$

или

$$b_s = \max |b_i| \quad \forall i : b_i < 0$$

Если в ведущей строке s все элементы неотрицательны, то целевая функция двойственной задачи не ограничена на множестве допустимых значений (задача не имеет решения).

Если в s-й строке существуют элементы $a_{sj} < 0$, то осуществляется выбор ведущего столбца.

5. Выбор ведущего столбца. При этом для s-й ведущей строки определяются отношения $\frac{\Delta_j}{a_{sj}}$ для $a_{sj} < 0$, среди них выбираются минимальное, и столбец, которому оно соответствует, объявляется ведущим

$$\frac{\Delta_k}{a_{sk}} = \min_{a_{sj}<0} \frac{\Delta_j}{a_{sj}}$$

6. Пересчет элементов симплекс-таблицы по правилу прямоугольника. Переход к шагу 3.

Двойственный симплекс-метод может применяться:

- для решения задач линейной оптимизации, правые части ограничений которых могут быть отрицательны (при этом двойственный симплекс-метод позволяет в ряде случаев упростить решение исходной задачи и обойтись без использования искусственных переменных);
- для решения задач линейной оптимизации с возрастающим числом ограничений.

Пример. Решить задачу линейной оптимизации:

$$F = x_1 + x_2 + 2x_3 \to \max$$
$$\begin{cases} x_1 + x_2 + x_3 = 8 \\ x_1 - x_2 \geq 4 \\ x_1 + 2x_2 \geq 6 \end{cases}$$
$$x_1, x_2, x_3 \geq 0$$

Решение. Приведем задачу к канонической форме, введя три дополнительные переменные: x_3, x_4, x_5.

$$F = -x_1 - x_2 - 2x_3 \to \min$$
$$\begin{cases} x_1 + x_2 + x_3 = 8 \\ x_1 - x_2 - x_4 = 4 \\ x_1 + 2x_2 - x_5 = 6 \end{cases}$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

В системе ограничений данной задачи имеется только одна базисная переменная x_3. Умножаем второе и третье ограничения на (–1), в результате чего в системе ограничений появятся еще 2 базисные переменные x_4 и x_5

$$-x_1 - x_2 - 2x_3 \to \min$$
$$\begin{cases} x_1 + x_2 + x_3 = 8 \\ -x_1 + x_2 + x_4 = -4 \\ -x_1 - 2x_2 + x_5 = -6 \end{cases}$$
$$x_1, x_2, x_3, x_4, x_5 \geq 0$$

Так как правые части второго и третьего ограничений отрицательны, для решения данной задачи используем двойственный симплекс-метод.

Составим симплекс-таблицу:

$x_Б$	$c_Б$	B	-1	-1	-2	0	0
			x_1	x_2	x_3	x_4	x_5
x_3	-2	8	1	1	1	0	0
x_4	0	-4	-1	1	0	1	0
x_5	0	-6	-1	**-2**	0	0	1
		-16	-1	-1	0	0	0

Псевдоплан задачи X = (0, 0, 8, -4, -6) не является оптимальным, т. к. имеет отрицательные компоненты. Определяем ведущую строку. Для этого в столбце правых частей выбираем отрицательные элементы и среди них определяем минимальный. Следовательно, ведущей строкой будет третья строка. Для определения ведущего столбца определяем отношения оценок к отрицательным элементам ведущей строки и среди них выбираем минимальное.

Для первого столбца отношение $\frac{-1}{-1}=1$. Для второго столбца $\frac{-1}{-2}=\frac{1}{2}$. Следовательно, ведущим столбцом будет второй столбец. Элементы таблицы пересчитываем по правилу прямоугольника.

$x_Б$	$c_Б$	B	-1	-1	-2	0	0
			x_1	x_2	x_3	x_4	x_5
x_3	-2	5	1/2	0	1	0	1/2
x_4	0	-7	**-3/2**	0	0	1	1/2
x_2	-1	3	1/2	1	0	0	-1/2
		-13	-1/2	0	0	0	-1/2

В качестве ведущей строки выбирается вторая (т. к. в столбце находится единственный отрицательный элемент –7), в качестве ведущего столбца – первый.

В результате симплекс-таблица примет вид:

$x_Б$	$c_Б$	B	-1	-1	-2	0	0
			x_1	x_2	x_3	x_4	x_5
x_3	-2	8/3	0	0	1	1/3	2/3
x_1	-1	14/3	1	0	0	-2/3	-1/3
x_2	-1	2/3	0	1	0	1/3	-1/3
		-32/3	0	0	0	-1/3	-2/3

Так как условия оптимальности выполнены, получен оптимальный план исходной задачи:

$$x_1^* = 14/3;\ \ x_2^* = 2/3;\ \ x_3^* = 8/3;\ \ x_4^* = 0;\ \ x_5^* = 0.$$

4. НЕЛИНЕЙНАЯ ОПТИМИЗАЦИЯ

4.1. Постановка задачи нелинейной оптимизации

В общем виде задача нелинейной оптимизации может быть сформулирована следующим образом:

$$\begin{aligned} & f(X) \to \min \\ & g_i(X) \geq (\leq, =) b_i, \ i = \overline{1,m} \\ & x_j^{\min} \leq x_j \leq x_j^{\max}, \ \ j = \overline{1,n} \end{aligned} \qquad (4.1)$$

где хотя бы одна из функций f(x), $g_i(x)$ является нелинейной. Задача без ограничений называется задачей *безусловной* оптимизации, задача с ограничениями – задачей *условной* оптимизации.

4.2. Методы учета ограничений в задачах нелинейной оптимизации

Одним из наиболее распространенных приемов решения задач нелинейной оптимизации с ограничениями (4.1) является их преобразование к задачам безусловной оптимизации. При этом для учета прямых и функциональных ограничений могут быть использованы различные подходы.

Прямые ограничения на варьируемые параметры можно исключить из данной задачи с помощью замены переменных:

$$x_j = V_j(z_j), \qquad j = \overline{1,n}, \qquad -\infty \leq z_j \leq \infty \ ,$$

где $Z = (z_1, ..., z_n)$ – вектор новых варьируемых параметров.

Тогда целевая функция задачи может быть переформулирована следующим образом:

$$f(X) = f(V_1(Z),..., V_n(Z)) = \hat{f}(Z) \to \min .$$

Аналогичным образом переформулируются и функциональные ограничения:

$$g_i(X) = g_i(V_1(Z),..., V_n(Z)) = \hat{g}_i(Z) \geq (\leq, =) b_i$$

Переформулированная в новых переменных задача примет вид:

$$\hat{f}(Z) \to \min$$

$$\hat{g}_i(Z) \geq (\leq, =) b_i, \; i = \overline{1,m}$$

При этом новые переменные $Z = (z_1,..., z_n)$ могут изменяться в любых пределах. В случае, если границы изменения всех варьируемых параметров одинаковы, замена переменных может иметь вид:

$$x_j = V(z_j), \qquad j = \overline{1,n}, \qquad -\infty \leq z_j \leq \infty$$

Некоторые виды прямых ограничений параметры и стандартные функции преобразования, позволяющие исключить их из задач оптимизации, представлены в таблице:

Тип ограничений	Функция преобразования
$x_j \geq 0$	$x_j = z_j^2$
$x_j > 0$	$x_j = \exp(z_j)$
$x_j^{min} \leq x_j$	$x_j = x_j^{min} + z_j^2$
$x_j^{min} < x_j < x_j^{max}$	$x_j = x_j^{max} + (x_j^{min} - x_j^{max}) / \pi \cdot \mathrm{arcctg} z_j$

Рассмотрим теперь основные подходы к учету *функциональных ограничений.* Пусть решается задача минимизации с функциональными ограничениями, представленными в виде системы неравенств (4.2) и равенств (4.3):

$$f(X) \to \min$$

$$g_i(X) \ge b_i,\ i = \overline{1,m} \tag{4.2}$$

$$h_j(X) = d_j,\ j = \overline{1,r} \tag{4.3}$$

Для учета функциональных ограничений обычно используется *метод штрафных функций*. При этом осуществляется переход от исходной целевой функции f(X) к следующей функции:

$$P(X,\gamma) = f(X) + \gamma \cdot S(X).$$

Здесь S(X) – *штрафная функция (функция штрафа)*, отличная от нуля вне допустимой области D; $\gamma > 0$ – коэффициент штрафа, значение которого может быть постоянным или меняться на различных итерациях. Во втором случае параметр $\gamma_k > 0$ настраивается в ходе оптимизационного процесса (k – номер итерации).

В результате решение задач с ограничениями сводится к решению последовательности задач безусловной оптимизации вспомогательной функции P(X). При этом штрафная функция S(X) формируется таким образом, чтобы при нарушении ограничений задачи производился некоторый “штраф” за их нарушение. При решении задачи минимизации “штраф” заключается в том, что к целевой функции прибавляется некоторое положительное число, “отбрасывая” тем самым оптимизационный процесс от оптимальной точки.

Существует два метода построения штрафных функций:

- метод *внутренних* штрафных функций (барьерных функций);

- метод *внешних* штрафных функций.

Метод *внутренних штрафных функций* предназначен для учета *только ограничений-неравенств* и характеризуется следующей основной функцией штрафа:

$$S(X) = \sum_{i=1}^{m} \frac{1}{g_i(X) - b_i}.$$

При этом предполагается, что ограничения-равенства (4.3) в задаче отсутствуют. Тогда целевая функция оптимизационной задачи примет вид:

$$P(X, \gamma) = f(X) + \gamma \cdot \sum_{i=1}^{m} \frac{1}{g_i(X) - b_i} \quad \text{или}$$

$$P(X, \gamma) = f(X) + \gamma_k \cdot \sum_{i=1}^{m} \frac{1}{g_i(X) - b_i}.$$

При этом параметр $\gamma_k > 0$ выбирается таким образом, чтобы его значения стремились к нулю при $k \to \infty$ (k – номер итерации).

При использовании внутренних штрафных функций поиск оптимума следует начинать с внутренней точки допустимой области, то есть с точки, в которой все ограничения выполнены как строгие неравенства. При выходе на границу допустимой области значение штрафной функции S(X) (штраф) будет бесконечным, следовательно, оптимизационный процесс никогда не выйдет за пределы допустимой области. Недостатком данного метода является то, что он не позволяет решать задачи с ограничениями-равенствами. Кроме того, для их использования необходимо знать начальную допустимую точку. В этой связи более целесообразным является использование внешних штрафных функций.

С помощью метода *внешних штрафных функций* учитываются как ограничения-неравенства $g_i(X) \geq b_i$, так и огра-

ничения-равенства $h_j(X) = d_j$. Наибольшее распространение получили штрафные функции следующего вида:

$$S(X) = \sum_{i=1}^{m} (\min\{0; g_i(X) - b_i\})^2 + \sum_{j=1}^{r} (h_j(X) - d_j)^2, \qquad (4.4)$$

$$S(X) = \sum_{i=1}^{m} |\min\{0; g_i(X) - b_i\}| + \sum_{j=1}^{r} |h_j(X) - d_j|. \qquad (4.5)$$

При этом если ограничения не нарушаются, т. е. $g_i(X) \geq b_i$ и $h_j(X) = d_j$, то S(X)=0. Если ограничения нарушаются, то величина "штрафа" зависит от степени нарушения ограничения.

Может быть использована также комбинированная штрафная функция, в которой ограничения-неравенства учитываются с помощью внешних штрафных функций, а ограничения-равенства – с помощью внутренних (барьерных) функций.

$$S(X) = \sum_{i=1}^{m} \frac{1}{g_i(x) - b_i} + \sum_{j=1}^{r} (h_j(X) - d_j)^2.$$

Таким образом, задачи с ограничениями с помощью замены переменных и методов штрафных функций могут быть преобразованы к задачам безусловной оптимизации. После соответствующих преобразований полученные задачи решаются с использованием методов безусловной оптимизации.

4.3. Принципы построения и классификация методов безусловной оптимизации

Рассмотрим задачу безусловной оптимизации:

$$f(X) = f(x_1, ..., x_n) \to \min_{X \in R^n}$$

Алгоритмы безусловной оптимизации составляют основу алгоритмического обеспечения, предназначенного для

решения оптимизационных задач. Они представляют собой итерационные процедуры, реализующие последовательное приближение к искомому экстремуму:

$$X^{k+1} = X^k + \alpha_k H^k,$$

где k – номер итерации, H^k – направление движения на k-й итерации, α_k – величина шага в данном направлении. При этом

$$X^{k+1} = \begin{pmatrix} x_1^{k+1} \\ ... \\ x_n^{k+1} \end{pmatrix}, \quad X^k = \begin{pmatrix} x_1^k \\ ... \\ x_n^k \end{pmatrix}, \quad H^k = \begin{pmatrix} h_1^k \\ ... \\ h_n^k \end{pmatrix}.$$

Поиск поиска начинается с некоторой начальной точки X^0, которая называется *начальным приближением* и задается пользователем. Получаемая в процессе поиска последовательность точек $X^0, X^1, ..., X^k$ называется *траекторией поиска* и должна сходиться к оптимальной точке X*. Таким образом, алгоритмы безусловной оптимизации различаются способами выбора направления поиска H^k и величины шага α_k.

В зависимости от способа выбора направления поиска методы безусловной оптимизации делятся на методы 0, 1 и 2-го порядка.

Методы нулевого порядка – методы, в которых для определения направления поиска используются только значения целевой функции. Производные в данном случае не вычисляются. Этот класс алгоритмов называется еще *методами прямого поиска* или *поисковыми методами оптимизации*. К ним относятся метод переменного многогранника и различные алгоритмы покоординатной оптимизации и случайного поиска. Особенностью методов поисковой оптимизации является то, что они могут быть использованы при алгоритмическом задании критериев оптимальности и ограничений.

Методы первого порядка – методы, в которых для определения направления поиска используются первые производные целевой функции по управляемым параметрам. Эти методы называют также *градиентными*. В качестве направления поиска на каждой итерации в них выбирается градиент (в задачах максимизации) или антиградиент (в задачах минимизации) оптимизируемой функции.

Методы второго порядка – методы, в которых для определения направления используются вторые производные целевой функции. К этому классу относят метод Ньютона и его модификации.

Необходимо заметить, что все рассматриваемые методы являются *локальными* и способны находить только локальный оптимум.

4.4. Методы нулевого порядка

Методы оптимизации нулевого порядка (поисковые методы оптимизации) используют в процессе оптимизации *только значения целевой функции*. Все они являются эвристическими и реализуют различные стратегии направленного перебора.

В зависимости от способа выбора направления поиска методы нулевого порядка делятся на следующие группы:

1. *Методы покоординатной оптимизации.* В данных методах поиск осуществляется вдоль координатных направлений. К этому классу относятся:

- алгоритм покоординатного спуска с постоянным шагом;
- релаксационный метод Гаусса-Зейделя;
- метод конфигураций Хука-Дживса;
- процедура вращения осей Розенброка;
- алгоритм сопряженных направлений Пауэлла.

2. *Алгоритмы переменного многогранника.* В эту группу входит метод переменного многогранника Нелдера-Мида и его модификации. Методы базируются на стратегии поиска,

заключающейся в построении в n-мерном пространстве многогранника из n+1-й вершины и перемещении его в направлении оптимальной точки с помощью трех основных операций: отражения, растяжения и сжатия.

3. *Методы случайного поиска.* При использовании данных методов направление поиска на каждой итерации выбирается случайным образом. Методы отличаются друг от друга способами генерации случайных векторов, а также стратегиями анализа текущей информации и перестройки шага поиска. Достоинством этой группы методов является возможность определения глобального оптимума, так как использование случайного механизма повышает вероятность выхода поискового процесса из зон локальных экстремумов.

4.4.1. Простейшие алгоритмы покоординатного спуска

В методах покоординатной оптимизации в качестве направлений поиска выбираются координатные оси пространства варьируемых параметров. При этом на каждой k-й итерации выполняется n шагов циклического покоординатного спуска из текущей точки X^k вдоль каждой из n координатных осей. Покоординатный спуск сводится к поочередному изменению переменных вдоль одной из осей:

$$X^{k+1} = X^k + \lambda_i^k \cdot e_i,\ i = 1 \ldots n,$$

где e_i – i-й координатный n-мерный вектор, у которого i-й элемент равен 1, а остальные элементы равны 0:

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ \ldots \\ 0 \end{pmatrix} \quad e_2 = \begin{pmatrix} 0 \\ 1 \\ \ldots \\ 0 \end{pmatrix} \quad \ldots \quad e_n = \begin{pmatrix} 0 \\ 0 \\ \ldots \\ 1 \end{pmatrix},$$

λ_i^k – длина шага поиска по i-й переменной на k-ой итерации.

Таким образом, в координатной форме шаги циклического спуска имеют следующий вид:

$$\begin{pmatrix} x_1^{k+1} \\ \dots \\ x_n^{k+1} \end{pmatrix} = \begin{pmatrix} x_1^{k} \\ \dots \\ x_n^{k} \end{pmatrix} + \lambda_1^k \begin{pmatrix} 1 \\ \dots \\ 0 \end{pmatrix} \quad (\text{i=1 – первая координатная ось}),$$

…

$$\begin{pmatrix} x_1^{k+1} \\ \dots \\ x_n^{k+1} \end{pmatrix} = \begin{pmatrix} x_1^{k} \\ \dots \\ x_n^{k} \end{pmatrix} + \lambda_n^k \begin{pmatrix} 0 \\ \dots \\ 1 \end{pmatrix} \quad (\text{i=n – n-я координатная ось}).$$

Геометрической интерпретацией траектории покоординатного поиска является ломаная, состоящая из отрезков, параллельных осям координат (рис. 4.1).

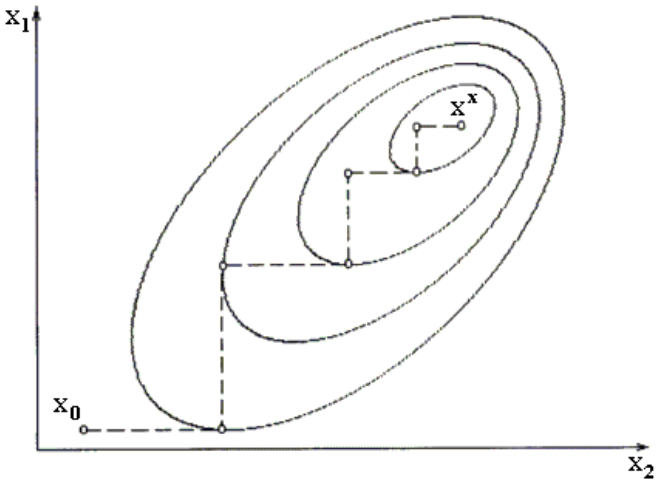


Рис. 4.1

На рис. 4.1 изображены также *линии уровня целевой функции* двух переменных. Каждая линия уровня соответствует некоторому постоянному значению целевой функции $f(X) = \text{const}$ (в случае большего числа переменных речь идет о *поверхностях уровня* целевой функции). Таким образом, процесс минимизации заключается в последовательном переходе от одной линии уровня $f(X) = C_i$ к линии уровня с меньшим значением целевой функции $f(X) = C_{i+1}$, $(C_{i+1} < C_i)$.

Необходимо заметить, что для нелинейной целевой функции линии уровня часто имеют овражный характер. При этом целевая функция сильно меняется по одним направлениям и слабо – по другим. Такие овражные ситуации создают значительные трудности для оптимизации.

К простейшим алгоритмам покоординатной оптимизации можно отнести *алгоритм покоординатного спуска с постоянным шагом* и *релаксационный метод Гаусса-Зейделя.* Различия между этими двумя алгоритмами заключается в способе выбора шага поиска λ_i^k.

Покоординатный спуск с дроблением шага

В этом методе на каждой k-й итерации циклический покоординатный спуск заключается в следующем. Сначала осуществляется движение из точки X^k с шагом λ_1^k вдоль 1-й координатной оси:

$$X^k + \lambda_1^k e_1.$$

Если это приводит к уменьшению значения целевой функции: $f(X^k + \lambda_1^k e_1) < f(X^k)$, осуществляется переход в точку $X^k + \lambda_1^k e_1$. В противном случае производится пробный шаг в противоположном направлении:

$$X^k - \lambda_1^k e_1 .$$

В случае уменьшения значения целевой функции осуществляется переход в точку $X^k - \lambda_1^k e_1$, в противном случае переход в новую точку не производится. Затем рассматривается следующая координатная ось. Аналогичные действия повторяются для всех координатных осей. Если в результате движения вдоль всех координатных осей значение целевой функции не уменьшилось, осуществляется дробление шага, и циклический покоординатный спуск повторяется из точки X^k с уменьшенной длиной шага. Итерационный процесс заканчивается, когда длины шагов вдоль координатных направлений уменьшатся до некоторого числа ε.

Основные шаги алгоритма:

1. Задать начальную точку $X^0 = (x_1^0,..., x_n^0)$, точность вычислений $\varepsilon > 0$, начальные величины шагов по координатным направлениям $\lambda_1^0,..., \lambda_n^0$, коэффициент уменьшения шага $\beta > 1$. Положить $k = 0$, $X^k = X^0$, i=1.

2. Осуществить циклический покоординатный поиск по каждому координатному направлению:

2.1. Произвести движение из точки X^k с шагом λ_i^k вдоль i-й координатной оси: $X^k + \lambda_i^k e_i$. Если $f(X^k + \lambda_i^k e_i) < f(X^k)$, шаг считается удачным. В этом случае положить $X^k = X^k + \lambda_i^k e_i$ и перейти к шагу 2.3.

2.2. Если в п. 2.1. шаг неудачен, произвести поиск в противоположном направлении: $X^k - \lambda_i^k e_i$. Если при этом значение целевой функции уменьшилось: $f(X^k - \lambda_i^k e_i) < f(X^k)$ положить $X^k = X^k - \lambda_i^k e_i$ и перейти к шагу 2.3.

2.3. Если не все координатные оси рассмотрены (i<n), положить i=i+1 и перейти к шагу 2.1 (продолжить покоординатный спуск по оставшимся направлениям). Если i=n, положить $X^{k+1} = X^k$ и перейти к шагу 3.

3. Проверить условие окончания поиска. Если все $\lambda_i^k < \varepsilon$, $i = \overline{1,n}$, закончить поиск. При этом $X^* = X^{k+1}$. В противном случае перейти к шагу 4.

4. Если $f(X^{k+1}) < f(X^k)$, положить k=k+1, i=1 и перейти к шагу 2 (повторить циклический покоординатный поиск из новой точки). В противном случае перейти к шагу 5.

5. Так как цикл покоординатного поиска не привел к уменьшению значения функции, уменьшить длины шагов вдоль каждого направления: $\lambda_i^k = \lambda_i^k / \beta$ и перейти к шагу 2.

Релаксационный алгоритм Гаусса-Зейделя

Отличие данного алгоритма от предыдущего заключается в том, что в нем на каждой k-й итерации определяются оптимальные длины шагов λ_i^k, $i = \overline{1,n}$ в результате решения вспомогательных задач одномерной оптимизации. При этом процедура поиска точки минимума X^* сводится к следующей последовательности действий.

1. Задается начальное приближение $X^0 = (x_1^0, ..., x_n^0)$ и точность вычислений $\varepsilon > 0$. При этом k=0, $X^k = X^0$.

2. Осуществляется циклический покоординатный спуск из точки X^k по итерационной схеме $X^{k+1} = X^k + \lambda_i^k \cdot e_i$, $i = 1 \ldots n$ с выбором оптимальной длины поискового шага λ_i^k вдоль каждого направления $i = 1...n$. Эта процедура образует внутренний цикл, в процессе которого осуществляется одномерная минимизация функции $f(x_1, ..., x_n)$ по каждой переменной x_i, $i = 1...n$:

$$f(x_1^{k+1}, x_2^k, ..., x_n^k) = \min_{x_1} f(x_1, x_2^k, ..., x_n^k),$$

$$f(x_1^{k+1}, x_2^{k+1}, ..., x_n^k) = \min_{x_2} f(x_1^{k+1}, x_2, ..., x_n^k),$$

….

$$f(x_1^{k+1}, x_2^{k+1}, ..., x_n^{k+1}) = \min_{x_n} f(x_1^{k+1}, x_2^{k+1}, ..., x_n).$$

В результате каждого i-го шага внутреннего цикла определяется очередная координата x_i^{k+1} точки X^{k+1}. После окончания внутреннего цикла в качестве нового приближения принимается точка X^{k+1}.

3. Проверяется выполнение критерия окончания итерационного процесса. Если $|x_i^{k+1} - x_i^k| \le \varepsilon$ для всех $i = 1...n$, работа алгоритма заканчивается. В противном случае осуществляется переход к шагу 2 (при этом k=k+1).

4.4.2. Метод конфигураций Хука-Дживса

Дальнейшим развитием методов покоординатной оптимизации является метод конфигураций Хука-Дживса. В данном методе этап циклического покоординатного поиска вдоль координатных осей чередуется с этапом экстраполяции, т.е. движения в направлении, соединяющем две перспективные точки X^k и X^{k+1} на двух последовательных шагах (рис. 4.2). Первый этап называется в методе Хука-Дживса *исследующим поиском вокруг базисной точки*, а второй этап – *поиском по образцу*.

На этапе исследующего поиска из базисной точки X^k осуществляется движение вдоль координатных осей в соответствии с общей схемой покоординатного спуска, в результа-

те чего будет получена точка X^{k+1}. Если этот этап не приводит к уменьшению значения целевой функции, длина шага уменьшается и исследующий поиск повторяется.

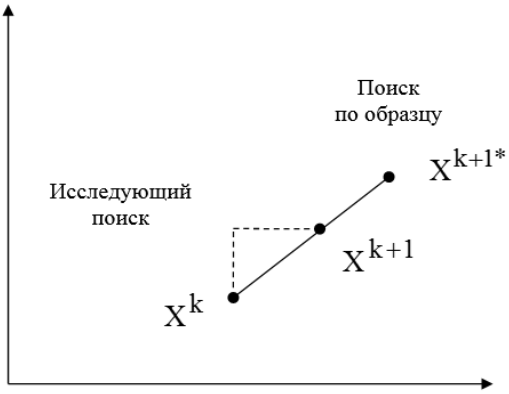


Рис. 4.2

Если этап исследующего поиска оказывается успешным, т.е. $f(X^{k+1}) < f(X^{k})$, реализуется этап поиска по образцу. При этом производится движение из точки X^{k+1} в направлении $X^{k+1} - X^{k}$, после чего осуществляется исследующий поиск вокруг полученной точки X^{k+1*}. Если полученное при этом значение целевой функции меньше, чем значение в точке X^{k+1}, поиск по образцу считается удачным. В противном случае осуществляется возврат в точку X^{k+1} и исследующий поиск продолжается вокруг нее. Алгоритм заканчивает работу, когда текущая длина шага становится меньше заданной точности ε.

Основные шаги алгоритма :

1. Выбирается начальная базисная точка $X_b^0 = X^0$ и шаг длиной λ_j для каждой переменной x_j, $j = 1...n$. Вы-

числяется значение целевой функции в базисной точке $f(X_b^0)$. При этом необходимо положить k=0.

2. *Исследующий поиск*

2.1. Осуществляется циклический покоординатный спуск по всем координатным осям с целью получения новой базисной точки. При этом каждая переменная изменяется по очереди прибавлением длины шага. Сначала вычисляется значение функции $f(X_b^k + \lambda_1 e_1)$, где e_1 – единичный вектор в направлении оси x_1. Если это приводит к уменьшению значения целевой функции, точка X_b^k заменяется на $X_b^k + \lambda_1 \mathbf{e}_1$. В противном случае вычисляется значение функции $f(X_b^k - \lambda_1 e_1)$, и если ее значение уменьшилось, то X_b^k заменяется на $X_b^k - \lambda_1 \mathbf{e}_1$. Если ни один из проделанных шагов не приводит к уменьшению значения функции, то точка X_b^k остается неизменной, и рассматриваются изменения в направлении оси x_2, т. е. находится значение функции $f(X_b^k + \lambda_2 e_2)$ и т.д. Когда будут рассмотрены все n переменных, мы будем иметь новую базисную точку X_b^{k+1}.

2.2. Если $X_b^k = X_b^{k+1}$, то есть уменьшения функции не было достигнуто, проверяется выполнение критерия окончания итерационного процесса. Если текущие длины шагов вдоль всех направлений меньше заданной точности ε ($\lambda_i < \varepsilon,\ i = 1...n$), работа алгоритма заканчивается.

Если критерий останова не выполнен, то исследование повторяется вокруг той же базисной точки X_b^k, но с уменьшенной длиной шага. На практике целесообразным считается уменьшение шагов в десять раз от начальной длины, т.е. $\lambda_i = \lambda_i / 10$ для всех $i = 1...n$.

2.3. Если $X_b^k \neq X_b^{k+1}$, то производится поиск по образцу.

3. *Поиск по образцу*

3.1. Осуществляется движение из новой базисной точки X_b^{k+1} в направлении $X_b^{k+1} - X_b^k$, поскольку поиск в этом направлении уже привел к уменьшению значения функции. Определяется точка образца:

$$X^{k+1*} = X_b^{k+1} + (X_b^{k+1} - X_b^k) = X_b^k + 2(X_b^{k+1} - X_b^k)$$

3.2. Производится исследующий поиск вокруг точки образца X^{k+1*}.

3.3. Если наименьшее значение целевой функции на этапе 3.2. меньше значения в точке X_b^{k+1}, то соответствующая точка будет являться новой базисной точкой X_b^{k+2}. В этом случае необходимо положить k=k+1 и вернуться к шагу 3 (повторить этап поиска по образцу).

В противном случае шаг поиска по образцу считается неудачным и его результаты не используются. При этом точка X^{k+1*} отбрасывается, осуществляется возврат в точку X_b^{k+1} и вокруг нее осуществляется исследующий поиск. Для этого необходимо положить k=k+1 и перейти к шагу 2.

Недостатком описанных выше методов является то, что при минимизации функций f(X), имеющих овраг, дно которого не ориентировано вдоль одной из координатных осей, процесс поиска сильно замедляется и может вообще остановиться вдали от точки локального минимума. В таких ситуациях предпочтительным является использование овражно-ориентированных методов покоординатной оптимизации, к которым, в частности, относится метод Розенброка.

4.4.3. Метод вращения осей Розенброка

Идея метода состоит во вращении системы координат в соответствии с изменением скорости убывания целевой функции. Новые направления координатных осей определяются таким образом, чтобы одна из них соответствовала направлению наиболее быстрого убывания целевой функции, а остальные находятся из условия ортогональности.

Напомним, что линейно независимые векторы $p_1,...,p_n$, по норме равные единице, называются *взаимно ортогональными*, если для всех $i, j = 1,..., n$ справедливо условие:

$$p_i^T p_j = 0, \quad j \neq i .$$

Работа метода проиллюстрирована на рис. 4.3. Из начальной точки X[0] осуществляют спуск в точку X[1] по направлениям, параллельным координатным осям. На следующем шаге одна из осей должна проходить в направлении y_1 = X[1] -X[0], а другая – в направлении, перпендикулярном к y_1.

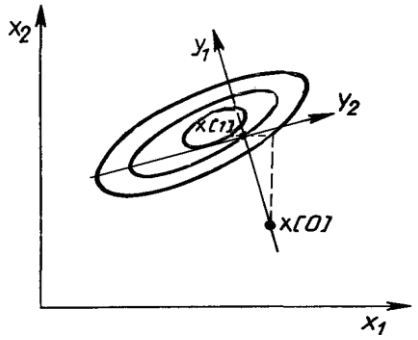


Рис. 4.3

Основные шаги алгоритма:

1. Задать начальную точку $X^0 = (x_1^0,..., x_n^0)$, точность вычислений $\varepsilon > 0$, начальные величины шагов по координат-

ным направлениям $\lambda_1^0,...,\lambda_n^0$, коэффициенты растяжения (увеличения шага) $\alpha > 1$ и сжатия (уменьшения шага) $-1 < \beta < 0$. В качестве начальных ортогональных направлений $p_1,...,p_n$ выбрать координатные направления:

$$p_1 = e_1 = \begin{pmatrix} 1 \\ 0 \\ ... \\ 0 \end{pmatrix}, \qquad p_2 = e_2 = \begin{pmatrix} 0 \\ 1 \\ ... \\ 0 \end{pmatrix}, \ \dots, \qquad p_n = e_n = \begin{pmatrix} 0 \\ 0 \\ ... \\ 1 \end{pmatrix}.$$

Положить $k = 0$, $X^k = X^0$, i=1.

2. Осуществить циклический покоординатный поиск по каждому ортогональному направлению:

2.1. Произвести движение из точки X^k с шагом λ_i^k вдоль i-го направления: $X^k + \lambda_i^k p_i$. Если при этом $f(X^k + \lambda_i^k p_i) < f(X^k)$, то умножить текущий шаг поиска на коэффициент растяжения $\lambda_i^{k+1} = \alpha\lambda_i^k$ и перейти к шагу 2.3.

2.2. Если в п. 2.1. движение оказалось неудачным, т.е. $f(X^k + \lambda_i^k p_i) > f(X^k)$, уменьшить шаг, умножив его на коэффициент сжатия: $\lambda_i^{k+1} = \beta\lambda_i^k$ и перейти к шагу 2.3.

2.3. Произвести движение из точки X^k вдоль i-го направления с новым шагом $X^k + \lambda_i^{k+1} p_i$. Вычислить значение целевой функции $f(X^k + \lambda_i^{k+1} p_i)$.

2.3.1. Если $f(X^k + \lambda_i^{k+1} p_i) < f(X^k)$, движение с новым шагом оказалось успешным. В этом случае перейти в новую точку поиска с новым шагом: $X^{k+1} = X^k + \lambda_i^{k+1} p_i$. Положить $\gamma_i^k = \lambda_i^{k+1}$, где γ_i^k – результирующий шаг поиска при переходе в новую точку на данной итерации.

2.3.2. Если $f(X^k + \lambda_i^{k+1} p_i) > f(X^k)$, но $f(X^k + \lambda_i^k p_i) < f(X^k)$ (движение на этапе 2.1. со старым шагом λ_i^k привело к уменьшению значения функции), перейти в новую точку поиска со старым шагом: $X^{k+1} = X^k + \lambda_i^k p_i$. При этом положить $\gamma_i^k = \lambda_i^k$, $\lambda_i^{k+1} = \lambda_i^k$.

2.3.3. Если $f(X^k + \lambda_i^{k+1} p_i) > f(X^k)$ и $f(X^k + \lambda_i^k p_i) > f(X^k)$ (движение по данному направлению оказалось неудачным даже после корректировки шага), переход в новую точку поиска не осуществляется. При этом необходимо положить $\gamma_i^k = 0$.

2.4. Если не все координатные оси рассмотрены (i<n), положить i=i+1 и перейти к шагу 2.1. (продолжить покоординатный спуск по оставшимся направлениям). Если i=n, положить $X^{k+1} = X^k$ и перейти к шагу 3.

Таким образом, в результате выполнения шага 2 будет получена новая точка $X^{k+1} = X^k + \sum_{i=1}^{n} \gamma_i^k p_i$.

3. Проверить условие окончания поиска. Если все $\left|\lambda_i^{k+1}\right| < \varepsilon$, $i = \overline{1,n}$, закончить поиск. При этом $X^* = X^{k+1}$. В противном случае перейти к шагу 4.

4. Если $f(X^{k+1}) = f(X^k)$, т. е. этап 2 не привел к уменьшению функции, положить k=k+1, i=1 и перейти к шагу 2 (повторить циклический покоординатный поиск с уменьшенной длиной шага).

5. Если $f(X^{k+1}) < f(X^k)$, построить новую систему ортогональных направлений. При этом в качестве первого

направления принимается вектор $X^{k+1} - X^k = \sum_{i=1}^{n} \gamma_i^k p_i$, а остальные направления строят ортогональными к первому. Построение ортогональных направлений осуществляется с использованием процедуры ортогонализации Грама-Шмидта:

$$a_i = \begin{cases} p_i, & \gamma_i^k = 0, \\ \sum_{j=i}^{n} \gamma_j^k p_j, & \gamma_j^k \neq 0, \end{cases} \qquad b_i = \begin{cases} a_i & i = 1, \\ a_i - \sum_{j=1}^{i-1} (a_i^T p_j) p_j, & i \geq 2, \end{cases}$$

$$p_i = \frac{b_i}{\|b_i\|}, \qquad i = 1,..., n.$$

После построения новых направлений следует положить: k=k+1; $\lambda_i^k = \lambda_i^0$ для всех $i = 1,..., n$; i=1 и перейти к шагу 2.

4.4.4. Метод переменного многогранника Нелдера-Мида

Метод переменного многогранника является одним из наиболее эффективных методов поисковой оптимизации. Этот метод основан на построении последовательности n+1 точек, которые являются вершинами выпуклого многогранника, и последующем перемещении полученного многогранника в направлении оптимальной точки с помощью трех основных операций: *отражения, растяжения и сжатия.* В результате многогранники деформируются в зависимости от структуры поверхностей уровня целевой функции.

Основные шаги алгоритма :

1. Задаются исходные данные для работы алгоритма:
   - начальное приближение $X^1 = (x_1^1,..., x_n^1)$;
   - требуемая точность ε;

- длина шага λ для построения исходного многогранника;

- коэффициенты отражения α, растяжения β и сжатия γ.

2. Строится многогранник, состоящий из n+1 точек. При этом в качестве первой вершины многогранника выбирается начальная точка X^1, а остальные вершины формируются по следующей схеме:

$$X^{i+1} = X^1 + \lambda e_i, \quad i = 1...n$$

или в координатной форме

$$\begin{pmatrix} x_1^2 \\ ... \\ x_n^2 \end{pmatrix} = \begin{pmatrix} x_1^1 \\ ... \\ x_n^1 \end{pmatrix} + \lambda \begin{pmatrix} 1 \\ ... \\ 0 \end{pmatrix} \quad , \ldots , \quad \begin{pmatrix} x_1^{n+1} \\ ... \\ x_n^{n+1} \end{pmatrix} = \begin{pmatrix} x_1^1 \\ ... \\ x_n^1 \end{pmatrix} + \lambda \begin{pmatrix} 0 \\ ... \\ 1 \end{pmatrix}$$

После построения многогранника вычисляются значения целевой функции в его вершинах: $f(X^1),..., f(X^{n+1})$.

3. Осуществляется сортировка значений целевой функции. При этом находится наибольшее значение целевой функции f_h, следующее за наибольшим значением функции f_g, наименьшее значение функции f_m и соответствующие им точки X_h, X_g и X_m.

4. Вычисляется центр тяжести X_c всех точек, за исключением X_h,

$$X_c = \frac{1}{n} \sum_{i \neq h} X^i$$

и значение целевой функции в этой точке $f(X_c) = f_c$.

5. *Процедура отражения.* "Наихудшая" точка X_h отражается относительно центра тяжести X_c по следующей схеме:

$$X_r = X_c + \alpha(X_c - X_h),$$

где X_r – точка, полученная в результате отражения; $\alpha > 0$ - коэффициент отражения (рекомендуемое значение $\alpha = 1$).

Операция отражения иллюстрируется на рис. 4.4.

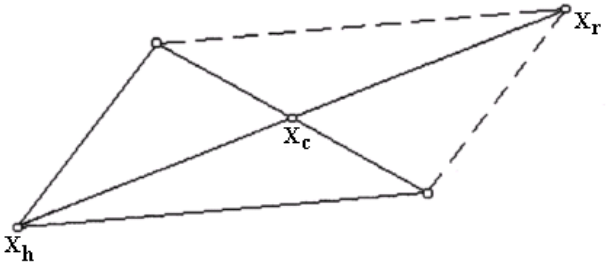


Рис. 4.4

После получения точки X_r находится значение целевой функции $f_r = f(X_r)$.

6. *Анализ результатов отражения.* Сравниваются значения функции f_r и f_m.

6.1. Если $f_r < f_m$, то получено наименьшее значение функции. Направление из точки X_c в точку X_r является перспективным и делается попытка движения в данном направлении с большим шагом. При этом производится *растяжение* многогранника по следующей схеме:

$$X_e = X_c + \beta(X_r - X_c),$$

где X_e – точка, полученная в результате растяжения; $\beta > 1$ - коэффициент растяжения (рекомендуемое значение $\beta = 2$).

Операция растяжения иллюстрируется на рис. 4.5.

После получения точки X_e находится значение целевой функции $f_e = f(X_e)$. Далее производится *анализ результатов растяжения*.

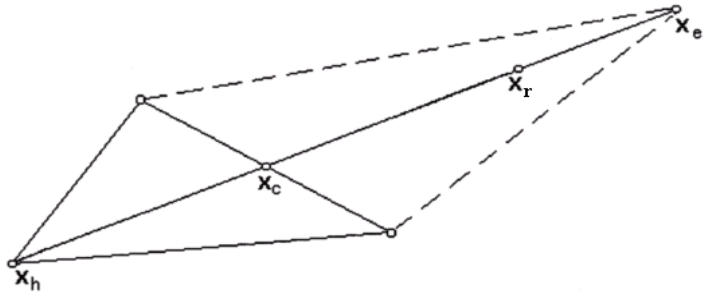


Рис. 4.5

6.1.1. Если $f_e < f_m$, то этап растяжения признается удачным. “Наихудшая” точка X_h заменяется на точку X_e, после чего осуществляется переход к шагу 10 (проверка сходимости).

6.1.2. Если $f_e \geq f_m$, точка X_e отбрасывается (этап растяжения признается неудачным). При этом “наихудшая” точка X_h заменяется на точку X_r, полученную в результате отражения. Далее осуществляется переход к шагу 10.

6.2. Если $f_m < f_r \leq f_g$, то точка X_r является лучшей точкой по сравнению с двумя “наихудшими” вершинами многогранника. При этом точка X_h заменяется на точку X_r, после чего осуществляется переход к шагу 10.

6.3. Если $f_r > f_g$, то очевидно, что мы переместились слишком далеко от точки X_h в направлении X_c. Этап отражения признается неудачным и производится сжатие многогранника (шаг 7).

7. *Процедура сжатия.*

7.1. Если $f_r \geq f_h$, то процедура сжатия проводится по следующей схеме:

$$X_p = X_c + \gamma(X_h - X_c),$$

где X_p – точка, полученная в результате сжатия; $0 < \gamma < 1$ – коэффициент сжатия (рекомендуемое значение $\gamma = 0.5$).

7.2. Если $f_r < f_h$, то перед сжатием осуществляется замена точки X_h на X_r, а значения f_h на f_r. При этом процедура сжатия проводится следующим образом:

$$X_p = X_c + \gamma(X_r - X_c).$$

Обе процедуры иллюстрируются на рис. 4.6.

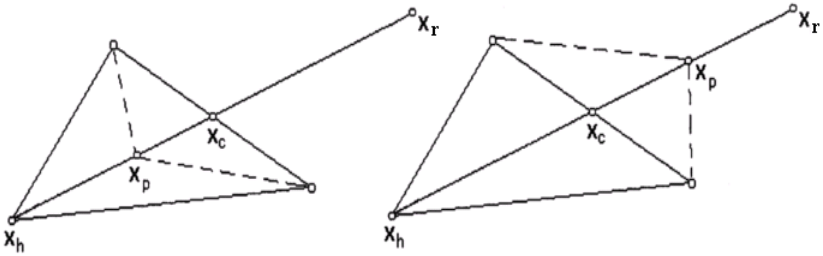


Рис. 4.6

После получения точки X_p находится значение $f_p = f(X_p)$.

8. *Анализ результатов сжатия.* Сравниваются значения функция f_p и f_h.

8.1. Если $f_p < f_h$, то процедура сжатия признается удачной. При этом точка X_h заменяется точкой X_p и осуществляется переход к шагу 10.

8.2. Если $f_p \geq f_h$, то очевидно, что все попытки улучшить значение f_h закончились неудачей. В данном случае происходит переход к шагу 9.

9. *Редукция (усечение) многогранника.* На этом шаге осуществляется уменьшение размерности многогранника де-

лением пополам расстояния от каждой точки многогранника до X_m – точки, определяющей наименьшее значение функции.

Таким образом, каждая точка X^i заменяется на точку

$$X^i + \frac{1}{2}(X^i - X_m), \quad i = \overline{1,m} .$$

Затем вычисляются значения $f_i = f(X^i)$ для всех $i = 1...m$ и осуществляется переход к шагу 10.

10. *Проверка сходимости.* Она основана на том, чтобы среднее квадратическое отклонение значений функции было меньше некоторого заданного малого значения ε. В этом случае вычисляется значение σ по формуле :

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n+1}(f_i - \bar{f})^2}{n+1}} \quad , \text{ где } \bar{f} = \frac{\sum f_i}{n+1} .$$

Если $\sigma \le \varepsilon$, то все значения функции очень близки друг к другу, и поэтому они, возможно, лежат вблизи точки минимума X^*.

Если сходимость не достигнута ($\sigma > \varepsilon$), осуществляется возврат к шагу 3.

4.5. Методы первого порядка

В методах первого порядка используются первые производные целевой функции f(X). В качестве направления поиска на каждой итерации выбирается градиент целевой функции (в задачах максимизации) или антиградиент (в задачах минимизации) Поэтому эти методы называют также *градиентными.* Итерационный процесс реализуется по схеме:

$$X^{k+1} = X^k + \alpha_k \nabla f(X^k) \quad \text{(максимизация);} \qquad (4.6)$$

$$X^{k+1} = X^k - \alpha_k \nabla f(X^k) \quad \text{(минимизация).} \qquad (4.7)$$

Здесь $\nabla f(X)$ – градиент (вектор частных производных) целевой функции:

$$\nabla f(X) = \begin{pmatrix} \dfrac{\partial f(X)}{\partial x_1} \\ \dots \\ \dfrac{\partial f(X)}{\partial x_n} \end{pmatrix}.$$

Таким образом, координатной форме градиентная процедура имеет вид (для задачи минимизации):

$$\begin{pmatrix} x_1^{k+1} \\ \dots \\ x_n^{k+1} \end{pmatrix} = \begin{pmatrix} x_1^{k} \\ \dots \\ x_n^{k} \end{pmatrix} - \alpha_k \begin{pmatrix} \dfrac{\partial f(X^k)}{\partial x_1} \\ \dots \\ \dfrac{\partial f(X^k)}{\partial x_n} \end{pmatrix}.$$

Градиентные методы различаются в зависимости от способов выбора и корректировки величины шага α_k. Наиболее распространены два метода. Первый называется *методом с дроблением шага* и связан с проверкой на каждой итерации некоторого неравенства, обеспечивающего убывание целевой функции. Второй метод называется *методом наискорейшего спуска* и обеспечивает выбор на каждой итерации оптимальной длины шага в результате решения вспомогательной задачи одномерной оптимизации.

4.5.1. Градиентный метод с дроблением шага

В данном методе перед началом поиска пользователем задается начальная величина шага α_0. На каждой итерации проверяется выполнение условия убывания целевой функции: $f(X^{k+1}) < f(X^k)$. Если данное условие не выполняется, осу-

ществляется дробление шага до тех пор, пока оно не окажется истинным. Для регулировки шага может быть использовано также более жесткое условие:

$$f(X^{k+1}) - f(X^k) \le -\gamma\alpha_k \left\| \nabla f(X^k) \right\|^2, \qquad (4.8)$$

где $0 < \gamma < 1$ - произвольно выбранная константа, неизменная на всех итерациях.

Геометрическая интерпретация градиентного метода с дроблением шага представлена на рис. 4.7. Здесь изображены линии уровня целевой функции, причем $C_1 > C_2 > C_3$, и зигзагообразная траектория поиска $x_0, x_1, x_2 ... x_k$. В качестве направления поиска в каждой точке x_k выбирается направление антиградиента, ортогональное касательной к линии уровня целевой функции в этой точке.

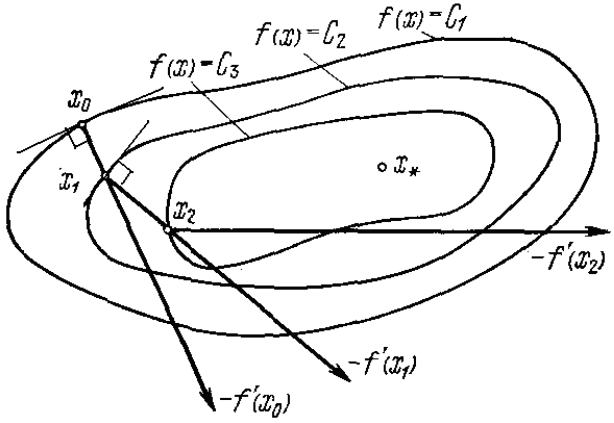


Рис. 4.7

Основные шаги алгоритма:

1. Задать исходные данные для работы алгоритма:
   - начальное приближение $X^0 = (x_1^0, ..., x_n^0)$;

- начальную величину шага α_0;

- коэффициент для дробления шага β (рекомендуемое значение $\beta = 2$);

- требуемую точность ε.

Положить k=0.

2. Вычислить градиент $\nabla f(X^k)$ в точке X^k.

3. Вычислить $X^{k+1} = X^k - \alpha_k \nabla f(X^k)$.

4. Если $f(X^{k+1}) < f(X^k)$, перейти к шагу 5. В противном случае положить $\alpha_k = \frac{\alpha_k}{\beta}$ и вернуться к шагу 3.

5. Проверить выполнение условий

$$\left\| X^{k+1} - X^k \right\| < \varepsilon, \qquad \left\| f(X^{k+1}) - f(X^k) \right\| < \varepsilon.$$

Если оба условия выполнены, работа алгоритма закончена (при этом $X^* = X^{k+1}$). Если хотя бы одно из условий не выполнено, положить k=k+1 и перейти к шагу 2.

4.5.2. Метод наискорейшего спуска

В методе наискорейшего спуска на каждой итерации определяется оптимальная длина шага в результате решения вспомогательной задачи одномерной оптимизации

$$\min_{\alpha \geq 0} f(X^k - \alpha \nabla f(X^k)).$$

Таким образом, этап градиентного метода (переход в новую точку поиска) чередуется с этапом определения оптимальной длины шага.

Геометрическая интерпретация метода наискорейшего спуска представлена на рис. 4.8. В этом методе, в отличие от обычного градиентного спуска, направление движения из точ-

ки X^k касается линии уровня в точке X^{k+1}. Последовательность точек $X^0, X^1, ..., X^k$ зигзагообразно приближается к точке минимума X_*, причем звенья этого ломаной ортогональны между собой. Действительно, шаг α выбирается из условия минимизации по α функции

$$\varphi(\alpha) = f(X^k - \alpha \nabla f(X^k)).$$

Поэтому должно выполняться условие:

$$\frac{d\varphi(\alpha_k)}{d\alpha} = -\nabla f(X^{k+1}) \nabla f(X^k) = 0.$$

Таким образом, направления поиска на двух последовательных итерациях ортогональны между собой.

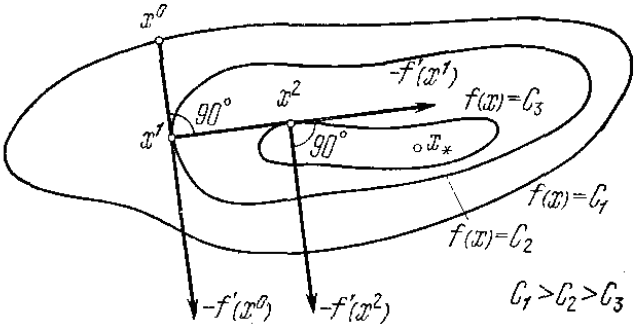


Рис. 4.8

Основные шаги алгоритма :

1. Задать исходные данные для работы алгоритма:

- начальное приближение $X^0 = (x_1^0, ..., x_n^0)$;
- требуемую точность ε.

Положить k=0.

2. Вычислить градиент $\nabla f(X^k)$ в точке X^k.

3. Вычислить величину шага α_k^* из условия

$$\varphi(\alpha_k) = f(X^k - \alpha_k \nabla f(X^k)) \to \min_{\alpha_k} .$$

4. Вычислить $X^{k+1} = X^k - \alpha_k^* \nabla f(X^k)$.

5. Проверить выполнение условий

$$\left\| X^{k+1} - X^k \right\| < \varepsilon, \qquad \left\| f(X^{k+1}) - f(X^k) \right\| < \varepsilon .$$

Если оба условия выполнены, работа алгоритма закончена (при этом $X^* = X^{k+1}$). Если хотя бы одно из условий не выполнено, положить k=k+1 и перейти к шагу 2.

Градиентные методы достаточно просты в реализации и обладают более высокой скоростью сходимости, чем методы нулевого порядка, так как используют информацию о первых производных целевой функции. Однако их сходимость значительно снижается в случаях, когда поверхности уровня целевой функции имеют овражный характер. Это бывает, если матрица вторых производных целевой функции (матрица Гессе) является плохо обусловленной, т. е. ее максимальное M и минимальное m собственные числа сильно отличаются друг от друга. В таких случаях применяются методы оптимизации второго порядка, которые используют вторые производные целевой функции и позволяют преодолевать овражные ситуации в ходе оптимизационного процесса.

4.6. Методы второго порядка

Методы второго порядка используют в процессе поиска вторые производные целевой функции. К этому классу относят метод Ньютона и его модификации. Основная стратегия данных методов заключается в следующем:

$$X^{k+1} = X^k - \alpha_k [\nabla^2 f(X^k)]^{-1} \nabla f(X^k), \qquad (4.9)$$

где $\nabla f(X^k)$ – градиент целевой функции f(X) в точке X^k, $[\nabla^2 f(X^k)] = G(X^k)$ – матрица вторых производных (матрица Гессе) целевой функции f(X) в точке X^k.

Методы данного класса отличаются друг от друга способами выбора и настройки шага α_k.

4.6.1. Метод Ньютона

В методе Ньютона величина шага на каждой итерации выбирается равной единице: $\alpha_k = 1$. Итерационная схема поиска в данном случае имеет вид:

$$X^{k+1} = X^k - [\nabla^2 f(X^k)]^{-1} \nabla f(X^k)$$

или

$$X^{k+1} = X^k - G^{-1}(X^k) \nabla f(X^k).$$

Основная сложность реализации метода Ньютона заключается в необходимости обращения матрицы Гессе, так как процедура вычисления обратной матрицы является достаточно трудоемкой. Поэтому на практике часто используется подход, при котором направление поиска

$$H^{k+1} = -G^{-1}(X^k) \nabla f(X^k)$$

определяется в результате решения системы линейных уравнений:

$$G(X^k) H^{k+1} = -\nabla f(X^k).$$

После определения направления поиска H^k новая ночка находится по схеме:

$$X^{k+1} = X^k + H^k.$$

Основные шаги алгоритма :

1. Задать исходные данные для работы алгоритма:

- начальное приближение $X^0 = (x_1^0, ..., x_n^0)$;

- требуемую точность ε.

Положить k=0.

2. Вычислить градиент $\nabla f(X^k)$ в точке X^k.

3. Определить матрицу Гессе $G(X^k)$ в точке X^k.

4. Определить направление поиска $H^k = -G^{-1}(X^k)\nabla f(X^k)$.

5. Найти новую точку $X^{k+1} = X^k + H^k$.

6. Проверить выполнение условий

$$\left\| X^{k+1} - X^k \right\| < \varepsilon, \qquad \left\| f(X^{k+1}) - f(X^k) \right\| < \varepsilon.$$

Если оба условия выполнены, работа алгоритма закончена (при этом $X^* = X^{k+1}$). Если хотя бы одно из условий не выполнено, положить k=k+1 и перейти к шагу 2.

Метод Ньютона обладает более высокой скоростью сходимости по сравнению с градиентными методами при минимизации овражных функций. Недостатки метода заключаются в том, что он, во-первых, предполагает вычисление вторых производных, что является достаточно трудоемкой процедурой, а во-вторых, может расходиться, если целевая функция не является сильно выпуклой и начальное приближение находится достаточно далеко от минимума.

4.6.2. Метод Ньютона-Рафсона

Метод Ньютона-Рафсона отличается от метода Ньютона тем, что в нем шаг поиска не выбирается равным единице на всех итерациях, а настраивается в ходе оптимизационного процесса. Итерационная схема поиска имеет вид:

$$X^{k+1} = X^k - \alpha_k G^{-1}(X^k)\nabla f(X^k).$$

Методы с регулировкой шага сходятся независимо от начального приближения.

На практике обычно используются два способа выбора длины шага. Первый из них связан с проверкой неравенства, аналогичного неравенству (4.8):

$$f(X^{k+1}) - f(X^k) \le \gamma \alpha_k (\nabla f(X^k), H^k),$$

где $H^k = -G^{-1}(X^k)\nabla f(X^k)$ - направление поиска на k-й итерации; $0 < \varepsilon < \frac{1}{2}$ - некоторая константа, неизменная на всех итерациях. Если это неравенство выполняется при $\alpha_k = 1$, то шаг принимается равным единице и осуществляется следующая итерация. В противном случае шаг дробится до тех пор, пока неравенство не выполнится. В случае, если проверка данного неравенства оказывается слишком трудоемкой, может быть использовано упрощенное условие: $f(X^{k+1}) < f(X^k)$.

Второй способ определения длины шага, как и в методе наискорейшего спуска, состоит определении на каждой итерации оптимальной длины шага в результате решения задачи одномерной минимизации:

$$\min_{\alpha} f(X^k - \alpha G^{-1}(X^k)\nabla f(X^k)) \quad .$$

При этом этап перехода в новую точку поиска чередуется с этапом определения оптимальной длины шага.

Далее рассмотрим алгоритм, реализующий второй способ регулировки шага.

Основные шаги алгоритма:

1. Задать исходные данные для работы алгоритма:

- начальное приближение $X^0 = (x_1^0, ..., x_n^0)$;
- требуемую точность ε.

Положить k=0.

2. Вычислить градиент $\nabla f(X^k)$ в точке X^k.

3. Определить матрицу Гессе $G(X^k)$ в точке X^k.

4. Определить направление поиска $H^k = -G^{-1}(X^k)\nabla f(X^k)$.

5. Вычислить величину шага α_k^* из условия

$$\varphi(\alpha_k) = f(X^k - \alpha_k G^{-1}(X^k)\nabla f(X^k)) \to \min_{\alpha_k} .$$

6. Найти новую точку $X^{k+1} = X^k + \alpha_k^* H^k$.

7. Проверить выполнение условий

$$\left\| X^{k+1} - X^k \right\| < \varepsilon, \qquad \left\| f(X^{k+1}) - f(X^k) \right\| < \varepsilon .$$

Если оба условия выполнены, работа алгоритма закончена (при этом $X^* = X^{k+1}$). Если хотя бы одно из условий не выполнено, положить k=k+1 и перейти к шагу 2.

5. МЕТОДЫ ДИСКРЕТНОЙ ОПТИМИЗАЦИИ

5.1. Классификация задач дискретной оптимизации

При решении задач автоматизированного проектирования и управления широкий круг задач формализуется в виде задач дискретной оптимизации. *Задачей дискретной оптимизации* называется задача оптимизации, в которой на варьируемые параметры наложено требование дискретности:

$$\begin{aligned} & f(X) \to \min(\max) \\ & g_i(X) \geq (\leq, =) b_i, \quad i = \overline{1,m} \\ & x_j \in D_j, \quad j = \overline{1,n}, \end{aligned} \tag{5.1}$$

где D_j – множество допустимых значений каждого параметра x_j. При этом предполагается, что хотя бы один параметр x_j может принимать дискретные значения.

Задачи дискретной оптимизации можно разделить на следующие основные классы:

1. В зависимости от вида целевой функции и ограничений задачи дискретной оптимизации делятся на *линейные* и *нелинейные*. Наиболее изученным в настоящее время является класс линейных задач дискретной оптимизации.

2. В зависимости от характера изменения варьируемых параметров различают задачи:

- *полностью дискретные*;
- *частично дискретные* (дискретно-непрерывные);
- *целочисленные задачи*;
- *задачи с булевыми переменными*. При этом если целевая функция принимает действительные значения, то задачи оптимизации с булевыми переменными называют задачами *псевдобулевой оптимизации*.

3. В зависимости от физического смысла варьируемых параметров выделяются следующие классы задач:

3.1. *Задачи с неделимостями*, в которых переменные представляют собой физические неделимые величины (например, количество выпускаемых изделий). К ним относятся все рассмотренные выше задачи оптимизации, в которых на переменные дополнительно наложены условия целочисленности.

3.2. Задачи с альтернативными переменными (или с логическими условиями). В этих задачах вводятся альтернативные переменные, отражающие логические условия задачи. Как правило, к этому классу относятся задачи булевой оптимизации.

Наиболее распространенными являются задачи целочисленной (в частности, булевой) оптимизации, так как любую дискретную задачу с помощью д преобразований можно привести к целочисленной. Такое преобразование может быть произведено, например, следующим образом.

Пусть $x_j = \{q_{1j}, \dots, q_{mj}\}$, где $\{q_{1j}, \dots, q_{mj}\}$ – дискретный ряд значений переменной x_j. Введем бинарную переменную y_{ij}, принимающую значения 0, 1 и представим каждую переменную x_j следующим образом:

$$x_j = \sum_{i=1}^{m} q_{ij} y_{ij} \quad \text{при условиях:}$$

$$\sum_{i=1}^{m} y_{ij} = 1, \quad y_{ij} \in \{0, 1\}.$$

Таким образом, можно перейти от произвольных дискретных переменных к бинарным переменным, которые являются частным случаем целочисленных переменных. Поэтому в дальнейшем будем рассматривать задачи целочисленной оптимизации.

5.2. Методы решения задач дискретной оптимизации

Методы решения задач дискретной оптимизации делятся на следующие основные группы:

- методы отсечений (предназначены для решения только *линейных* целочисленных и частично целочисленных задач);
- комбинаторные методы;
- приближенные методы.

5.2.1. Метод отсечений Гомори

Рассмотрим задачу целочисленной линейной оптимизации:

$$
\begin{aligned}
&F = \sum_{j=1}^{n} c_j x_j \to \min \\
&\sum_{j=1}^{n} a_{ij} x_j = b_i,\ i = \overline{1,m} \\
&x_j \ge 0,\ j = \overline{1,n} \\
&x_j - \text{целые},\ j = \overline{1,k}
\end{aligned}
\qquad (5.2)
$$

Если все переменные задачи должны принимать целочисленные значения (k = n), задача называется *полностью целочисленной*. Если требование целочисленности предъявляется не ко всем переменным (k < n), задача называется *частично целочисленной.*

Обозначим задачу (5.2) через Z, а область ее допустимых решений D_z. Обозначим задачу (5.2) без учета условий целочисленности через P, а ее допустимую область D. Оптимальное решение целочисленной задачи будем обозначать X_z^*, оптимальное решение соответствующей непрерывной задачи P обозначим X^*. Область D_z представляет собой дис-

кретное множество точек с целочисленными координатами, которые принадлежат D: $D_z \subset D$ (рис. 5.1).

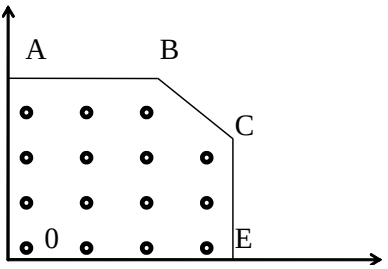


Рис. 5.1

Задачи Z и P характеризуются свойствами:

1. Минимальное значение целевой функции в задаче Z не превышает минимального значения этой же целевой функции в задаче P: $F(X_Z^*) \geq F(X^*)$. Таким образом, значение целевой функции в точке непрерывного оптимума является нижней границей значения целевой функции в точке дискретного оптимума.

2. Если оптимальное решение задачи P оказывается целочисленным, то оно является оптимальным решением задачи Z.

3. Если не имеет решений задача P, то не имеет решений и соответствующая целочисленная задача Z.

Решение задач целочисленной линейной оптимизации с использованием метода отсечений сводится к решению специальным образом построенных задач линейной оптимизации: P_0 , P_1 ,…, P_s. Первая задача P_0 образуется из исходной задачи целочисленной оптимизации Z путем отбрасывания ограничений целочисленности переменных. Каждая последующая задача P_1 ,…, P_s получается из предыдущей путем добавления к ее условиям дополнительного линейного ограничения-неравенства, называемого *правильным отсечением* (или сечением). При этом r-м сечением называется линейное огра-

ничение, вводимое в задачу P_{r-1} и соответствующее следующим двум условиям:

1. Любое целочисленное решение задачи P_{r-1} ему удовлетворяет.

2. Хотя бы одно из нецелочисленных решений задачи P_{r-1} ему не удовлетворяет (отсекается).

Таким образом, к исходной задаче P_0 последовательно добавляются дополнительные ограничения (отсечения), не исключающее целочисленных допустимых точек и отсекающие нецелочисленные решения решаемой задачи целочисленной оптимизации (рис. 5.2).

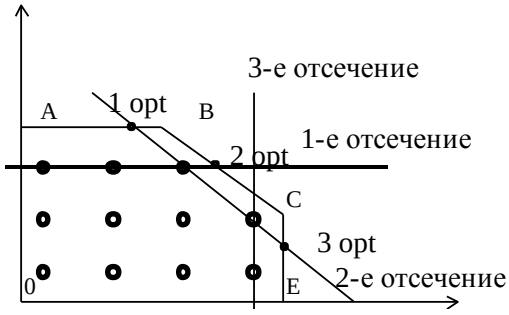


Рис. 5.2

Методы отсечения различаются между собой способами формирования дополнительных ограничений. Наиболее распространенным является *метод отсечения Гомори.*

Пусть задача P_{r-1} решена симплексным методом и ее решение не удовлетворяет условиям целочисленности.

Введем обозначения:

[a] – целая часть числа a, т.е. наибольшее целое число, не превосходящее a;

{a} = a-[a] – дробная часть числа a;

Bas – множество индексов базисных переменных из последней симплекс-таблицы;

S – номер строки из последней симплексной таблицы, которой соответствует переменная с наибольшей дробной ча-

стью из оптимального решения. При определении данного номера S в столбце правых частей ограничений выбирается наибольшее значение $\{b_S\}$.

Тогда *I сечение Гомори* для решения полностью целочисленной задачи запишется в виде:

$$\sum_{j \notin Bas} \{a_{Sj}\} x_j \geq \{b_S\}$$

II сечение Гомори, применяемое для решения частично целочисленной задачи, определяется следующим образом:

$$\sum_{j \notin Bas} \gamma_{Sj} x_j \geq \{b_S\},$$

где γ_{Sj} – коэффициенты, определяемые из следующих соотношений:

1. Для x_j, подчиненных требованиям целочисленности:

$$\gamma_{Sj} = \begin{cases} \{a_{Sj}\} \text{ при } \{a_{Sj}\} \leq \{b_S\} \\ \dfrac{\{b_S\}}{1-\{b_S\}}\left(1-\{a_{Sj}\}\right) \text{при } \{a_{Sj}\} > \{b_S\} \end{cases}$$

2. Для x_j, не подчиненных требованиям целочисленности

$$\gamma_{Sj} = \begin{cases} a_{Sj} \text{ при } a_{Sj} \geq 0 \\ \dfrac{\{b_S\}}{1-\{b_S\}}\left|a_{Sj}\right|, \text{если } a_{Sj} < 0 \end{cases}$$

Основные шаги метода Гомори:

1. Определяется оптимальное решение задачи (5.2) без учета условий целочисленности. При этом ограничения на целочисленность переменных отбрасываются, и задача решается обычным симплекс-методом.

2. Если полученное решение является целочисленным, работа алгоритма заканчивается (оптимальное решение

найдено), в противном случае осуществляется переход к шагу 3.

3. На основании последней симплекс-таблицы, полученной при решении ЗЛП, формируется сечение Гомори (I или II).

4. Дополнительное ограничение, построенное на предыдущем этапе, добавляется к условиям решаемой задачи. Для этого данное ограничение преобразуется в уравнение

$$\sum_{j \neq Bas} \{a_{Sj}\} x_j - x_d = \{b_S\} \text{ (для I сечения)}$$

$$\sum_{j \neq Bas} \gamma_{Sj} x_j - x_d = \{b_S\} \text{ (для II сечения),}$$

где x_d – дополнительная неотрицательная переменная.

Затем полученное уравнение умножается на –1

$$-\sum_{j \neq Bas} \{a_{Sj}\} x_j + x_d = -\{b_S\} \text{ (для I сечения)}$$

$$-\sum_{j \neq Bas} \gamma_{Sj} x_j + x_d = -\{b_S\} \text{ (для II сечения)}$$

и добавляется к последней симплекс-таблице. Далее полученная задача решается двойственным симплекс-методом.

5. Переход к шагу 1.

Пример. Решить задачу целочисленной линейной оптимизации методом Гомори:

$$-x_1 - 4x_2 \to \min$$

$$\begin{cases} -x_1 + 2x_2 + x_3 = 2 \\ 3x_1 + 2x_2 + x_4 = 6 \end{cases}$$

$$x_1, x_2, x_3, x_4 \geq 0$$

$$x_1, x_2, x_3, x_4 \text{ целые.}$$

Решение. Решаем данную задачу симплекс-методом без учета условий целочисленности.

$x_Б$	$c_Б$	B	-1	-4	0	0
			x_1	x_2	x_3	x_4
x_3	0	2	-1	**2**	1	0
x_4	0	6	3	2	0	1
		0	1	4	0	0
x_2	-4	1	-1/2	1	1/2	0
x_4	0	4	**4**	0	-1	1
		-4	3	0	-2	0
x_2	-4	3/2	0	1	3/8	1/8
x_1	-1	1	1	0	-1/4	1/4
		-7	0	0	-5/4	-3/4

Полученное решение ($x_1^* = 1;\ x_2^* = 3/2;\ x_3^* = 0;\ x_4^* = 0$) не является целочисленным. Поэтому переходим к построению дополнительного ограничения. Для этого выберем ту строку таблицы, которой соответствует переменная с наибольшей дробной частью. В данном случае единственной такой строкой является первая строка (S = 1). Записываем первое сечение Гомори:

$$\left\{\frac{3}{8}\right\}x_3 + \left\{\frac{1}{8}\right\}x_4 \geq \left\{\frac{3}{2}\right\} \quad \text{или} \quad \frac{3}{8}x_3 + \frac{1}{8}x_4 \geq \frac{1}{2}$$

Умножив левую и правую часть на 2, получим:

$$\frac{3}{4}x_1 + \frac{1}{4}x_4 \geq 1$$

Введем в левую часть дополнительную переменную x_5 и умножим полученное равенство на (–1). В результате будем иметь следующее равенство:

$$-\frac{3}{4}x_3 - \frac{1}{4}x_4 + x_5 = -1$$

Добавим полученное ограничение к последней симплекс-таблице и решим полученную задачу с использованием двойственного симплекс-метода

$x_Б$	$c_Б$	B	-1	-4	0	0	0
			x_1	x_2	x_3	x_4	x_5
x_2	-4	3/2	0	1	3/8	1/8	0
x_1	-1	1	1	0	-1/4	1/4	0
x_5	0	-1	0	0	**-3/4**	-1/4	1
		-7	0	0	-5/4	-3/4	0
x_2	-4	1	0	1	0	0	½
x_1	-1	4/3	1	0	0	1/3	-1/3
x_3	0	4/3	0	0	1	1/3	-4/3
		-16/3	0	0	0	-1/3	-5/3

В результате получено следующее решение:

$$x_1^* = 4/3;\ x_2^* = 1;\ x_3^* = 4/3;\ x_4^* = 0$$

Полученное решение не является целочисленным, поэтому опять переходим к построению сечения Гомори. Так как значения дробных частей у переменных x_1 и x_3 одинаковы, строку для построения дополнительного ограничения выбираем произвольно (например, S = 2). Тогда ограничение записывается в виде

$$\left\{\frac{1}{3}\right\}x_4 + \left\{-\frac{1}{3}\right\}x_5 \geq \left\{\frac{4}{3}\right\} \text{ или } \frac{1}{3}x_4 + \frac{2}{3}x_5 \geq \frac{1}{3}$$

После соответствующих преобразований получим

$$-x_4 - 2x_5 + x_6 = -1$$

Добавим данное ограничение к последней симплекс-таблице и решим полученную задачу двойственным симплекс-методом

$x_Б$	$c_Б$	B	-1	-4	0	0	0	
			x_1	x_2	x_3	x_4	x_5	x_6
x_2	-4	1	0	1	0	0	1/2	0
x_1	-1	4/3	1	0	0	1/3	-1/3	0
x_3	0	4/3	0	0	1	1/3	-4/3	0
x_6	0	-1	0	0	0	**-1**	-2	1
		-16/3	0	0	0	-1/3	-5/3	0
x_2	-4	1	0	1	0	0	-1/2	0
x_1	-1	1	1	0	0	0	-1	1/3
x_3	0	1	0	0	1	0	-6	1/3
x_4	0	1	0	0	0	1	2	-1
		-5	0	0	0	0	-1	-1/3

Полученное целочисленное решение является оптимальным решением исходной задачи: $x_1^* = 1;\ x_2^* = 1;\ x_3^* = 1;\ x_4^* = 1$.

Основным *недостатком метода Гомори* является то, что он может быть использован для решения только линейных задач. Более универсальным методом дискретной оптимизации является метод ветвей и границ.

5.3.2. Метод ветвей и границ

Метод ветвей и границ относится к комбинаторным методам решения задач дискретной оптимизации и может использоваться для решения как *линейных*, так и *нелинейных* задач. Он основан на использовании конечности множества

допустимых вариантов и замене полного перебора сокращенным направленным перебором. При этом полного перебора удается избежать за счет отбрасывания неперспективных множеств вариантов, т. е. тех множеств, где заведомо нет оптимального решения. В методе ветвей и границ все множество допустимых вариантов последовательно делится на все меньшие подмножества. При этом вычисляются оценки (границы), которые позволяют сделать вывод о том, какое из полученных подмножеств может быть отброшено при условии сохранения подмножества, содержащего оптимальное решение. Для иллюстрации работы метода ветвей и границ используется дерево, называемое *деревом перебора* или *деревом вариантов* (рис. 5.3).

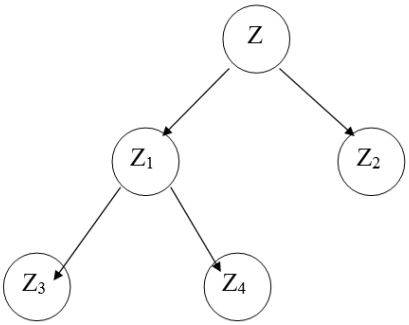


Рис. 5.3

Каждой вершине дерева вариантов соответствует какая-либо подзадача Z_i исходной задачи Z.

Перед рассмотрением метода ветвей и границ введем некоторые определения.

1. Для задачи $\min\limits_{X\in D} f(X)$ *подзадачами* будем называть задачи следующего вида: $\min\limits_{X\in \tilde{D}} f(X)$, где $\tilde{D} \subset D$. Таким образом, подзадача – это задача с той же целевой функцией $f(X)$,

но с меньшей допустимой областью, которая входит в исходную область D.

Так как при сужении допустимой области результат решения задачи ухудшается, то в общем случае выполняется следующее соотношение между оптимальными решениями исходной задачи и ее подзадач:

$$\min_{X \in D} f(X) \le \min_{X \in \underline{D}} f(X).$$

2. *Релаксацией* задачи (или подзадачи) будем называть переход от задачи $\min_{X \in D} f(X)$ к задаче с той же целевой функцией, но с некоторой допустимой областью D', включающей исходную область D в качестве подмножества: $\min_{X \in D'} f(X)$, где $D \subset D'$ (одним из примеров релаксации является, в частности, отбрасывание условий целочисленности в задаче целочисленной оптимизации).

Таким образом, при релаксации задачи происходит расширение ее допустимой области. Поэтому выполняется следующее соотношение между оптимальными решениями исходной задачи и ее релаксации:

$$\min_{X \in D'} f(X) \le \min_{X \in D} f(X).$$

Таким образом, оптимальное решение задачи $\min_{X \in D'} f(X)$ будет являться нижней границей оптимального решения исходной задачи.

3. *Ветвлением* будем называть разбиение допустимой области на непересекающиеся подмножества и получение таким образом новых оптимизационных подзадач. При этом каждой подзадаче будет соответствовать вершина дерева вариантов.

4. *Нижней оценкой* (нижней границей) μ_i оптимального значения целевой функции в вершине i будем называть

оптимальное решение задачи, полученной в результате релаксации подзадачи Z_i.

5. *Верхней оценкой* (верхней границей) ν_i оптимального решения в вершине i будем называть значение целевой функции, обеспечиваемое некоторым допустимым решением подзадачи Z_i (например, для задачи целочисленной оптимизации это целочисленное допустимое решение).

6. Значение минимальной верхней оценки в вершинах дерева вариантов называется *рекордом.*

7. Прозондированные вершины – это вершины дерева вариантов, в которых дальнейшее ветвление невозможно или нецелесообразно.

Пусть решается задача целочисленной минимизации целевой функции $\min\limits_{X \in D_Z} f(X)$, которую в дальнейшем обозначим Z. При этом D_Z – область допустимых решений исходной задачи, включающая условия целочисленности переменных.

Рассмотрим основные этапы метода ветвей и границ при решении данной задачи.

1. Вычисление нижней оценки оптимального решения исходной задачи Z. С этой целью осуществляется релаксация задачи Z (например, в результате отбрасывания условий целочисленности) и формируется расширенная допустимая область D. Построенная задача решается с использованием одного из методов непрерывной оптимизации (например, с использованием симплекс-метода при линейности целевой функции и ограничений), после чего осуществляется анализ полученного результата X^*.

1.1. Если оптимальное решение X^* целочисленное, то оно является оптимальным решением исходной задачи Z.

1.2. Если X^* нецелочисленное, то необходимо продолжить решение. При этом $\mu_Z = f(X^*)$ – нижняя граница оптимального целочисленного решения исходной задачи.

2. Ветвление, т. е. разбиение области допустимых решений D_Z исходной задачи Z на два подмножества: D_{Z_1} и D_{Z_2}. При этом исходная задача Z разбивается на две подзадачи: Z_1 и Z_2.

Для организации ветвления может быть использована следующая схема. Пусть x_j^* – какая-либо нецелочисленная переменная из полученного на предыдущем этапе оптимального решения X^*. При этом область D_{Z_1} формируется путем добавления к области D_Z следующего дополнительного ограничения:

$$x_j \le [x_j^*], \text{ где } [x_j^*] \text{ – целая часть числа } x_j^*.$$

Область D_{Z_2} образуется путем добавления к D_Z ограничения: $x_j \ge [x_j^*]+1$. При этом формируется задача Z_1 с допустимой областью D_{Z_1} и задача Z_2 с допустимой областью D_{Z_2}.

3. Вычисление нижних оценок μ_{Z_1} и μ_{Z_2} для подзадач Z_1 и Z_2. Для этого осуществляется релаксация подзадач Z_1 и Z_2 и их решение. При этом $\mu_{Z_1} = f(X_1^*)$, $\mu_{Z_2} = f(X_2^*)$, где X_1^* и X_2^* – оптимальные решения полученных задач.

Если какая-либо из подзадач не имеет решения, ее нижняя оценка считается равной бесконечности.

4. Из двух подзадач Z_1 и Z_2 выбирается подзадача для дальнейшего ветвления.

5. Ветвление выбранной подзадачи (аналогично п. 2). После ветвления снова вычисляются нижние оценки для полученных подзадач и т. д.

Процесс ветвления продолжается до получения целочисленного оптимального решения для одной из подзадач, например Z_k. При этом значение целевой функции на полученном целочисленном решении задачи Z_k представляет собой верхнюю границу оптимального решения: $\nu_{z_k} = f(X_k^*)$. Минимальная из полученных верхних границ будет являться рекордом.

После получения верхних границ определяются прозондированные вершины. Вершина является прозондированной, если она удовлетворяет одному из следующих условий:

1) оптимальное решение в этой вершине целочисленно;

2) значение нижней границы в этой вершине больше текущего значения рекорда (дальнейшее ветвление этой вершины нецелесообразно, так как в процессе ветвления значение целевой функции будет только увеличиваться);

3) подзадача, соответствующая данной вершине, не имеет допустимых решений. При этом в случаях 2) и 3) прозондированные вершины отбрасываются.

Процесс продолжается до тех пор, пока все вершины не будут прозондированы. Прозондированная вершина, соответствующая целочисленному решению с наилучшим значением целевой функции, определяет оптимальное решение исходной задачи.

Пример Для иллюстрации основных принципов метода ветвей и границ рассмотрим следующую задачу целочисленной линейной оптимизации:

$$-7x_1 - 3x_2 \to \min$$
$$5x_1 + 2x_2 \le 20$$
$$8x_1 + 4x_2 \le 38$$
$$x_1, x_2 \ge 0, \quad x_1, x_2 - \text{целочисленные}$$

Начальный шаг решения этой задачи методом ветвей и границ состоит в нахождении решения задачи линейного программирования, получаемой при отбрасывании условий целочисленности x_1 и x_2. Обозначим эту задачу через ЛП-1. Решим задачу ЛП-1 с использованием симплекс-метода. Ее оптимальное решение имеет вид: $x_1^* = 1;\ x_2^* = 7.5$, оптимальное значение целевой функции f_1=-29.5. Найденное значение f_1 представляет собой *нижнюю границу* истинного оптимального целочисленного значения, поскольку при выполнении требования целочисленности x_2 значение оптимальное значений целевой функции может лишь увеличиться.

Следующий шаг метода ветвей и границ состоит в разбиении задачи ЛП-1 на две подзадачи, первая из которых (ЛП-2) образуется в результате добавления к задаче ЛП-1 ограничения $x_2 \le 7$, а вторая (ЛП-3) – в результате добавления ограничения $x_2 \ge 8$.

ЛП-2

$$-7x_1 - 3x_2 \to \min$$
$$5x_1 + 2x_2 \le 20$$
$$8x_1 + 4x_2 \le 38$$
$$x_2 \le 7 (\text{новое ограничение})$$
$$x_1, x_2 \ge 0$$

ЛП-3

$$-7x_1 - 3x_2 \to \min$$
$$5x_1 + 2x_2 \le 20$$
$$8x_1 + 4x_2 \le 38$$
$$x_2 \ge 8 (\text{новое ограничение})$$
$$x_1, x_2 \ge 0$$

Оптимальное решение задачи ЛП-3: $x_1^* = 0.75;\ x_2^* = 8$, где f_2=-29.25. Оптимальное решение задачи ЛП-2: $x_1^* = 1.2;\ x_2^* = 7$, где f_3=-29.4.

Дерево возможных вариантов представлено на рис. 5.4.

Для дальнейшего ветвления выберем задачу ЛП-2, т.к. значение целевой функции в ней меньше. К задаче ЛП-2 добавим ограничения $x_1 \le 1$, (задача ЛП-4) и $x_1 \ge 2$ (задача ЛП-5).

Результаты решения задачи ЛП-4: $x_1^* = 1;\ x_2^* = 7$, где f_4=-28.

Результаты решения задачи ЛП-5: $x_1^* = 2;\ x_2^* = 5$, где f_5=-29.

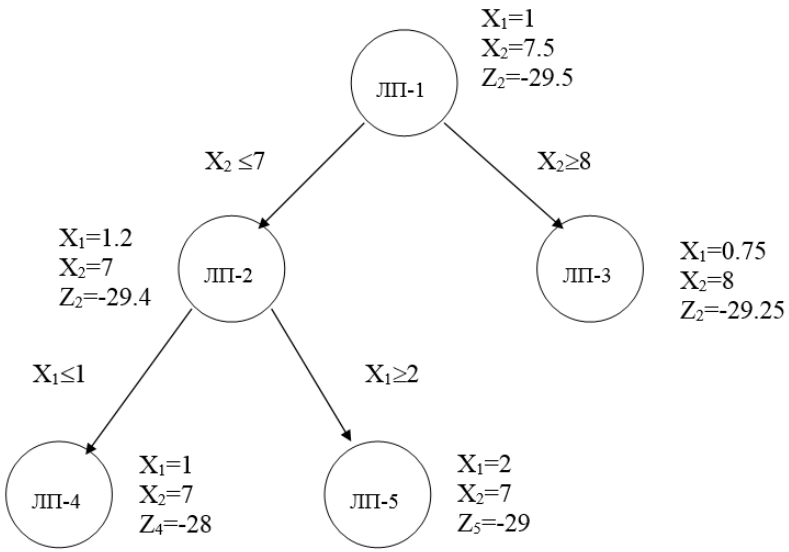


Рис. 5.4

Таким образом, получены два допустимых (целочисленных) решения исходной задачи целочисленного программирования, которые представляют собой *верхние границы* оп-

тимального решения, причем значение f_5=-29 будет являться *рекордом*. Даже если исходная задача имеет другие целочисленные решения, значение целевой функции в них не может быть больше -29.

Вершины ЛП-4 и ЛП-5 являются *прозондированными* (так как в них получено целочисленное решение, а в процессе дальнейшего ветвления оно может лишь ухудшаться). Ветвление вершины ЛП-3 также нецелесообразно в связи с тем, что целевая функция исходной задачи может принимать только целочисленные значения (т. к. все переменные и коэффициенты целочисленны), а при дальнейшем ветвлении ее значения будут увеличиваться (т. е., станут больше -29). Следовательно, оптимальное решение задачи:

$$x_1^* = 2;\ x_2^* = 5,\ \text{где}\ f(x_1^*, x_2^*) = -29.$$

Основные стратегии метода ветвей и границ

В рамках обобщенной схемы ветвей и границ может быть построен широкий класс алгоритмов, отличающихся друг от друга способами реализации отдельных шагов. Эффективность конкретного алгоритма в рамках данного класса зависит от выбранной стратегии его формирования для рассматриваемой задачи. Рассмотрим некоторые стратегии реализации различных этапов метода ветвей и границ.

1. *Выбор вершины (подзадачи) для ветвления* (этап 4). При этом возможны следующие стратегии:

- выбирается вершина с минимальной нижней оценкой;
- поиск в глубину: всегда выбирается одна из новых, только что полученных подзадач (стратегия LIFO);
- поиск в ширину (стратегия FIFO);
- вершина для ветвления выбирается случайным образом.

2. *Выбор нецелочисленной переменной, по которой осуществляется ветвление* (этап 2). Используются следующие стратегии реализации данного этапа:

- выбирается переменная с максимальной дробной частью;

- переменным присваиваются приоритеты и осуществляется выбор переменной с наибольшим приоритетом (данная переменная играет более важную роль в модели по мнению разработчиков);

- переменная для ветвления выбирается произвольным образом, например, переменная с наименьшим номером.

3. *Вычисление нижних оценок*. Самым простым способом вычисления нижних оценок является отбрасывание условий целочисленности и дальнейшее использование какого-либо метода непрерывной оптимизации (если задача линейная, может использоваться симплекс-метод, если нелинейная – соответствующие нелинейные методы и т. д.). Этот способ позволяет получать достаточно точные значения нижних оценок, однако является достаточно трудоемким. Поэтому на практике часто используются приближенные эвристические процедуры определения оценок, трудоемкость и время вычисления которых значительно ниже. Для задач целочисленной оптимизации специального вида (задачи коммивояжера, задачи о ранце, задачи о покрытии) разработаны специальные методы вычисления нижних границ, учитывающие специфику этих задач.

4. *Вычисление верхних оценок.* При этом возможны следующие варианты:

- верхние оценки специально не вычисляются, а определяются в процессе решения задачи при получении целочисленного результата для одной из вершин;

- верхние границы вычисляются в начале работы алгоритма с помощью какой-либо эвристической процедуры.

- вычисление верхних границ с помощью эвристических процедур осуществляется на каждом шаге работы алгоритма (для каждой подзадачи).

Последние два способа требуют дополнительных вычислительных затрат, но позволяет более эффективно сокращать перебор.

6. ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ ОПТИМИЗАЦИИ

В настоящее время при решении практических задач поиска оптимальных решений в различных предметных областях широкое применение находят эволюционно-генетические алгоритмы. Эволюционные вычисления доказали свою эффективность при решении трудноформализуемых задач непрерывной и дискретной оптимизации. К преимуществам эволюционных вычислений относятся адаптивность, способность к обучению, параллелизм, возможность построения гибридных интеллектуальных систем на основе комбинирования с парадигмами искусственных нейронных сетей и нечеткой логики.

Генетические алгоритмы (ГА) образуют класс алгоритмов поисковой оптимизации, основанных на математическом моделировании биологических механизмов и процессов в живой природе с помощью принципов популяционной генетики, которые позволяют находить оптимальные или близкие к ним (субоптимальные) решения. Информационная технология решения задач оптимизации с помощью алгоритмов поиска, реализующих генетический подход, основывается на использовании аналогов с эволюционными процессами скрещивания, кроссовера, мутации и естественного отбора.

К достоинствам эволюционных вычислений можно отнести следующие:

- широкую область применения, возможность решения широкого класса задач дискретной и непрерывной оптимизации;

- возможность проблемно-ориентированного кодирования решений, подбора начальной популяции, комбинирования эволюционных вычислений с неэволюционными алгоритмами, продолжения процесса эволюции до тех пор, пока имеются необходимые ресурсы;

- пригодность для поиска в сложном пространстве решений большой размерности;

- отсутствие ограничений на вид целевой функции, что позволяет использовать алгоритмы данного класса при реше-

нии оптимизационных задач со сложными алгоритмическими моделями;

- ясность схемы и базовых принципов эволюционных вычислений;

- интегрируемость эволюционных вычислений с другими неклассическими парадигмами искусственного интеллекта, такими, как искусственные нейронные сети и нечеткая логика.

6.1. Основные понятия

В задачах оптимизации любой вариант оптимизируемого объекта представляется значениями варьируемых параметров $X = (x_1,..., x_n)$

В генетических алгоритмах вектору варьируемых параметров X соответствует запись, называемая *хромосомой*. Элементы хромосомы соответствуют параметрам x_i, их называют *генами*.

В ГА оперируют подмножеством хромосом, образующих *популяцию*. Имитация генетических принципов — вероятностный выбор родителей среди членов популяции, скрещивание их хромосом, отбор потомков для включения в новые поколения объектов на основе оценки целевой функции — ведет к эволюционному улучшению значений целевой функции (функции полезности) от поколения к поколению.

Для применения генетического алгоритма (ГА) необходимо:

1. Выделить совокупность свойств объекта, характеризуемых варьируемыми параметрами $X = (x_1,..., x_n)$ и влияющих на его полезность. Среди x_i могут быть величины различных типов (real, integer, Boolean, enumeration). При этом могут решаться задачи как параметрической, так и структурной оптимизации. Значения $X = (x_1,..., x_n)$ могут соответствовать различным вариантам параметров или структуры объекта.

2. Сформулировать количественную оценку полезности вариантов объекта – целевую функцию $f(x_1,..., x_n)$.

3. Разработать математическую модель объекта, представляющую собой алгоритм вычисления $f(x_1,..., x_n)$ для заданного вектора X;

4. Представить вектор X в форме хромосомы – записи следующего вида (рис. 6.1).

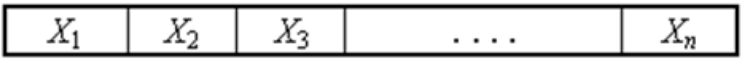


Рис. 6.1. Хромосома

Представление параметров объекта в виде хромосом осуществляется посредством кодирования. В хромосоме кодируются численные параметры решения. Часто используются двоичное кодирование. Тогда хромосома представляет собой битовую строку.

Например, рассмотрим хромосому из 12 бит, соответствующую трем параметрам объекта: 1001 1100 1011. Первый параметр равен 9, второй 12, третий 11.

Может использоваться и вещественное кодирование (рис. 6.2)

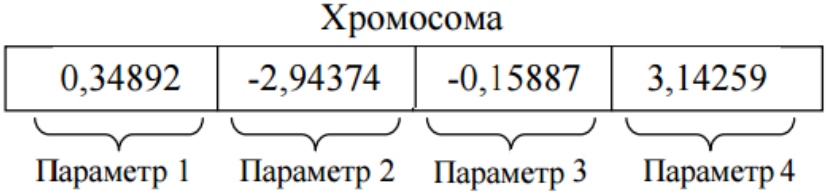


Рис. 6.2. Вещественное кодирование

При построении генетических алгоритмов часто применяется кодирование с использованием *кода Грея.*

Код Грея — непозиционный код с одним набором символов (0 и 1) для каждого разряда. Этот код строится из двоич-

ных цифр таким образом, что соседние в нем числа отличаются всегда только в одном разряде.

Младший разряд в последовательности чисел в коде Грея принимает значения 0 и 1, затем следующий старший разряд становится единичным и младший разряд принимает свои значения уже в обратном порядке (1, 0). Этим и объясняется название кода — отраженный. Соответственно, два младших разряда принимают значения 00, 01, 11, 10, а затем, при единичном следующем старшем разряде, те же значения в обратном порядке (10, 11, 01, 00). Ниже дана таблица, показывающая первые восемь чисел в двоичном коде и в коде Грея.

Таблица 6.1

число	*двоичный код*	*код Грея*
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

В генетических алгоритмах используется следующая терминология:

ген – часть хромосомы. Ген может быть битом, числом или другим объектом. При решении задач многомерной оптимизации ген соответствует закодированному управляемому параметру x_i. Таким образом, хромосома содержит n сцепленных между собой генов, расположенных в определённой последовательности;

аллель – значение гена;

локус (позиция) – местоположение определённого гена в хромосоме;

генотип – экземпляр хромосомы, заполненный конкретными значениями генов. Каждый генотип соответствует одному из решений исходной задачи. В ряде реализаций генотип называют *особью*;

популяция – конечное множество особей (генотипов)

фенотип – совокупность значений признаков, описывающих объект, получаемых после декодирования хромосомы. Иными словами – это реальные (не закодированные) значения признаков объекта;

функция приспособленности f – целевая функция. Её ещё называют фитнес-функцией.

Таким образом, в генетических алгоритмах существует строгое различие между фенотипом (решением, выраженным в терминах поставленной задачи) и генотипом (хромосомой, представлением решения). Генетический алгоритм работает с генотипом, фенотип служит для определения пригодности индивида (оценки качества решения поставленной задачи).

Фенотип определяет, чем является объект в реальном мире, а *генотип* содержит всю информацию об объекте на уровне хромосомного набора. При этом каждый ген, то есть элемент информации генотипа, имеет свое отражение в фенотипе. Таким образом, для решения задач нам необходимо представить каждый признак объекта в форме, подходящей для использования в генетическом алгоритме. Все дальнейшее функционирование механизмов генетического алгоритма производится на уровне генотипа, позволяя обойтись без информации о внутренней структуре объекта, что и обуславливает его широкое применение в самых разных задачах.

6.2. Обобщенная схема генетического алгоритма

Классический генетический алгоритм (также называемый элементарным или простым генетическим алгоритмом) состоит из следующих шагов:

1. Инициализация, или выбор исходной популяции хромосом.

2. Оценка приспособленности хромосом в популяции, т.е. расчет функции приспособленности хромосом.

3. Выбор родительской пары хромосом – тех хромосом, которые будут участвовать в создании потомков для следующей популяции;

4. Применение генетических операторов:

- скрещивание (кроссинговер);
- мутация;

5. Вычисление функции приспособленности потомков.

6. Селекция.

7. Повторение шагов 3-5 до тех пор, пока не будет получено новое поколение, содержащее N особей.

8. Повторение шагов 2-6 до тех пор, пока не будет достигнут критерий окончания процесса.

Таким образом, генетический алгоритм реализует циклический процесс смены поколений. Для каждого витка внешнего цикла генетического алгоритма выполняется внутренний цикл, на котором формируются экземпляры нового (следующего за текущим) поколения. Во внутреннем цикле повторяются операторы выбора родителей, скрещивания родительских хромосом, мутации, оценки приспособленности потомков, селекции хромосом для включения в очередное поколение.

Критерием окончания процесса может служить заданное количество поколений или схождение популяции. Схождением называется такое состояние популяции, когда все строки популяции почти одинаковы и находятся в области некоторого экстремума. В такой ситуации скрещивание практически никак не изменяет популяции, так как создаваемые при нем потомки представляют собой копии родите- лей с перемененными участками хромосом. Вышедшие из этой области за счет мутации особи склонны вымирать, так как чаще имеют меньшую приспособленность, особенно если данный

экстремум является глобальным максимумом. Таким образом, схождение популяции обычно означает, что найдено лучшее или близкое к нему решение.

Обобщенная схема базового генетического алгоритма представлена на рис. 6.3.

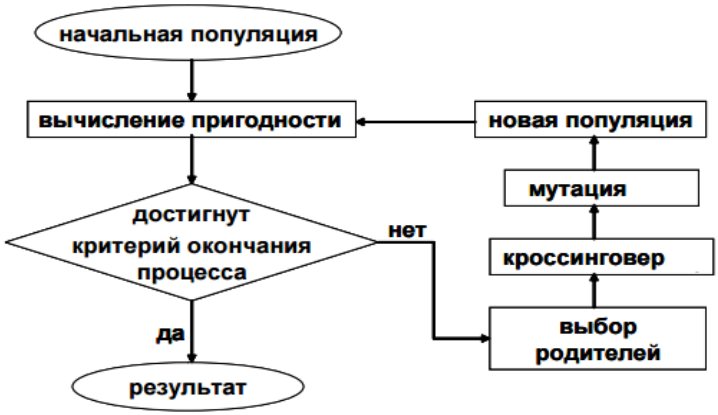


Рис. 6.3. Блок–схема базового генетического алгоритма

6.3. Основные генетические операторы

Основными операторами генетического алгоритма являются скрещивание (кроссинговер), мутация, выбор родителей и селекция (отбор хромосом в новую популяцию). Вид оператора играет важную роль в реализации и эффективности ГА. Существуют основные формы операторов, чистое использование или модернизация которых ведет к получению ГА, пригодного для решения конкретной задачи.

6.3.1. Выбор родительской пары

Важным этапом при реализации генетических алгоритмов является стратегия выбора родительской пары. Наиболее распространенными являются следующие:

1. *Панмиксия* – самый простой оператор отбора, когда обе особи, составляющие родительскую пару, случайным образом выбираются из всей популяции. При этом каждому члену популяции сопоставляется случайное целое число на отрезке [1; n], где n — количество особей в популяции. Будем рассматривать эти числа как номера особей, которые примут участие в скрещивании. При таком выборе какие-то из членов популяции не будут участвовать в процессе размножения, так как образуют пару сами с собой. Какие-то члены популяции примут участие в процессе воспроизводства неоднократно с различными особями популяции. Несмотря на простоту, такой подход универсален для решения различных классов задач. Однако он достаточно критичен к численности популяции, поскольку эффективность алгоритма, реализующего такой подход, снижается с ростом численности популяции.

2. *Инбридинг* – представляет собой такой метод, когда первый родитель выбирается случайным образом, а вторым родителем является член популяции ближайший к первому. Здесь «ближайший» может пониматься, например, в смысле минимального расстояния Хемминга (для бинарных строк) или евклидова расстояния между двумя вещественными векторами. Расстояние Хемминга равно числу различающихся локусов (разрядов) в бинарной строке. Например, расстояние Хемминга между хромосомами 1010001 и 1000000 равно 2.

3. Аутбридинг или кроссбридинг – формирует родительские пары из максимально далеких особей.

Последние два способа по-разному влияют на поведение генетического алгоритма. Так, инбридинг можно охарактеризовать свойством концентрации поиска в локальных узлах, что фактически приводит к разбиению популяции на отдельные локальные группы вокруг подозрительных на экстремум областей. Аутбридинг же направлен на предупреждение сходимости алгоритма к уже найденным решениям, заставляя алгоритм просматривать новые, неисследованные области.

Инбридинг и аутбридинг бывает фенотипным (когда в качестве расстояния берется разность значений целевой функции для соответствующих особей) и генотипным (в качестве расстояния берется расстояние Хемминга между генотипами).

Кроме рассмотренных стратегий для выбора родительских пар могут использоваться стратегии селекции, рассмотренные в п. 6.3.4.

6.3.2. Рекомбинация

Оператор рекомбинации применяют сразу же после оператора отбора родителей для получения новых особей-потомков. Смысл рекомбинации заключается в том, что созданные потомки должны наследовать генную информацию от обоих родителей. Различают дискретную рекомбинацию и кроссинговер.

Дискретная рекомбинация

Дискретная рекомбинация в основном применяется к хромосомам с вещественными генами. Основными способами дискретной рекомбинации являются собственно *дискретная рекомбинация*, *промежуточная,* и *линейная* рекомбинации.

Дискретная рекомбинация соответствует обмену генами между особями. Для иллюстрации данного оператора сравним две особи с тремя генами:

Особь 1	12	25	7
Особь 2	116	4	34

Для создания двух потомков с равной вероятностью случайно выберем номер особи для каждого гена:

Схема 1	2	2	1
Схема 2	1	2	1

Согласно схеме создадим потомков:

Потомок 1	116	4	7
Потомок 2	12	4	7

Дискретная рекомбинация применима для любого типа генов (двоичные, вещественные и символьные).

Промежуточная рекомбинация применима только к вещественным переменным, но не к бинарным. В данном методе предварительно определяется числовой интервал значений генов потомков, который должен содержать значения генов родителей. Потомки создаются по следующему правилу:

$$\text{Потомок} = \text{Родитель 1} + \alpha \cdot (\text{Родитель 2} - \text{Родитель 1}),$$

где множитель α — случайное число на отрезке $[-d, 1 + d]$, $d > 0$. Рекомендуемое значение $d = 0{,}25$. Для каждого гена создаваемого потомка выбирается отдельный множитель α.

Применим промежуточную рекомбинацию и особям из предыдущего примера. Случайно выберем значения $\alpha \in [-0{,}25; 1{,}25]$ для каждого гена обоих потомков

Схема 1	0,5	1,1	−0,1
Схема 2	0,1	0,8	0,5

Вычислим значения генов потомков по предложенной выше формуле:

Потомок 1	$12 + 0{,}5(116 - 12) = 64$	$25 + 1{,}1(4 - 25) = 1{,}9$	4,3
Потомок 2	$12 + 0{,}1(116 - 12) = 22{,}4$	$25 + 0{,}8(4 - 25) = 8{,}2$	20,5

При промежуточной рекомбинации возникают значения генов, отличные от значения генов особей-родителей. Это приводит к возникновению новых особей, пригодность которых может быть лучше, чем пригодность родителей. В литературе такой оператор рекомбинации иногда называется *дифференциальным скрещиванием.*

Линейная рекомбинация отличается от промежуточной тем, что множитель α выбирается для каждого потомка один раз. Рассмотрим гены приведенных выше родителей. Пусть значение α определяется следующим образом:

Схема 1	0,5
Схема 2	0,1

Тогда гены потомков примут следующие значения:

Потомок 1	$12 + 0{,}5(116 - 12) = 64$	$25 + 0{,}5(4 - 25) = 14{,}5$	20,5
Потомок 2	$12 + 0{,}1(116 - 12) = 22{,}4$	$25 + 0{,}1(4 - 25) = 22{,}9$	9,7

Если рассматривать особи популяции как точки в k-мерном пространстве, где k – количество генов в одной особи, то можно сказать, что при линейной рекомбинации генерируемые точки потомков лежат на прямой, заданной двумя точками – родителями.

Кроссинговер

Рекомбинацию бинарных строк принято называть *кроссинговером (кроссовером) (скрещиванием)*. Кроссинговер обычно проводится с вероятностью P_c, которая является входным параметром алгоритма.

Наиболее распространенными являются одноточечный, двухточечный, многоточечный и однородный кроссинговер.

Одноточечный кроссинговер моделируется следующим образом. Пусть имеются две родительские особи с хромосомами $X = \{x_i, i \in [0;L]\}$ и $Y = \{y_i, i \in [0;L]\}$. Случайным образом определяется точка внутри хромосомы (точка разрыва), в которой обе хромосомы делятся на две части и обмениваются ими. Такой тип кроссинговера называется одноточечным, так как при нем родительские хромосомы разделяются только в одной случайной точке.

Тогда гены родительских особей можно представить следующим образом:

$$X_1 = \{x_{11}, x_{12}, \dots, x_{1n}, x_{1n+1}, \dots, x_{1m}\}$$
$$Y_1 = \{y_{11}, y_{12}, \dots, y_{1n}, y_{1n+1}, \dots, y_{1m}\}$$

После скрещивания получим дочерние особи:

$$X_2 = \{y_{11}, y_{12}, \dots, y_{1n}, x_{1n+1}, \dots, x_{1m}\}$$
$$Y_2 = \{x_{11}, x_{12}, \dots, x_{1n}, y_{1n+1}, \dots, y_{1m}\}$$

Пример одноточечного кроссинговера представлен в таблице 6.2. При этом разрыв подразумевается между пятым и шестым локусами.

Таблица 6.2

Хромосома	**Ген 1**	**Ген 2**	**Ген 3**	**Ген 4**	**Ген 5**	**Ген 6**	**Ген 7**	**Ген 8**
Родителя A	f	a	c	d	g	k	v	e
Родителя B	a	b	c	d	e	f	g	h
Потомка C	f	a	c	d	g	f	g	h
Потомка D	a	b	c	d	e	k	v	e

В *двухточечном* алгоритме используются две точки разрыва и, следовательно, хромосомы делятся на три части. Хромосомы обмениваются участками, расположенными между двумя точками оператора кроссинговера. Пример двухточечного кроссинговера представлен на рис. 6.4.

```
P1  : 1 1 1 | 0 1 | 0 0
P2  : 0 0 0 | 1 1 | 1 0

P'1 : 1 1 1 | 1 1 | 0 0
P'2 : 0 0 0 | 0 1 | 1 0
```

Рис. 6.4. Двухточечный кроссинговер

Развитием двухточечного оператора кроссинговера является *многоточечный* оператор кроссинговера или N-точечный оператор кроссинговера. Многоточечный оператор

кроссинговера выполняется аналогично двухточечному хотя большое число «разрезающих» точек может привести к потере «хороших» родительских свойств. На рис. 6.4 представлен пример трехточечного кроссинговера.

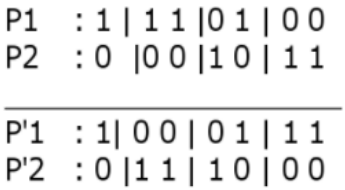


Рис. 6.5. Трехточечный кроссинговер

В *однородном* кроссинговере случайным образом генерируется строка B длиной D с двоичными значениями генов, где D -длина хромосомы. Если bi =0 (bi∈B, i∈[1,D]), то i-е значение гена в первом потомке берется от первого родителя (во втором потомке — от второго родителя), если bi=1, то в первый потомок включается ген второго родителя (во второй потомок — от первого родителя). Пример реализации такой стратегии представлен на рис. 6.5

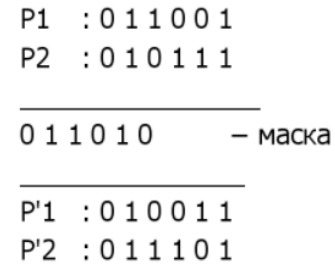


Рис. 6.6. Однородный кроссинговер

Алгоритм однородного кроссинговера для двоичных строк идентичен дискретному воспроизведению для вещественных хромосом. Такой вид кроссинговера еще называют *унифицированным*.

6.3.3. Мутация

После процесса воспроизводства происходит мутация. Данный оператор необходим для «выбивания» популяции из локального экстремума и препятствует преждевременной сходимости. Мутации могут быть точечными (в одном гене), макромутациями (в нескольких генах) и хромосомными (появление новой хромосомы).

Мутация обычно производится с вероятностью P_m для каждого гена. Вероятность мутации P_m (как правило, $P_m << 1$) может являться или фиксированным случайным числом на отрезке [0; 1], или функцией от какой-либо характеристики решаемой задачи. Например, можно положить вероятность мутирования генов, обратно пропорциональную числу всех генов в особи (размерности): $P_m = \frac{1}{n}$. Рассмотрим основные варианты мутации.

Двоичная мутация

Наиболее распространенной стратегией мутации особей, кодированных двоичным кодом, заключается в случайном инвертировании гена (0 заменяется 1 и наоборот).

$$x_1 x_2 \ldots x_{k-1} x_k x_{k+1} \ldots x_n \rightarrow x_1 x_2 \ldots x_{k-1} \overline{x}_k x_{k+1} \ldots x_n$$

Например, $0100101 \rightarrow 0101101$ (инвертируется 4-й бит).

Так же как и кроссинговер, мутации могут проводиться не только по одной случайной точке. Можно выбирать для изменения несколько точек в хромосоме, причем их число также может быть случайным. Используют и мутации с изменением сразу некоторой группы подряд идущих точек.

Кроме рассмотренной стратегии на практике используются также следующие виды мутаций:

1. Обмен местами случайно выбранного гена x_k с соседним геном:

$$x_1x_2...x_{k-1}x_kx_{k+1}...x_n \rightarrow x_1x_2...x_{k-1}x_{k+1}x_k...x_n$$

2. Обмен местами двух соседей случайно выбранного гена x_k:

$$x_1x_2...x_{k-1}x_kx_{k+1}...x_n \rightarrow x_1x_2...x_{k+1}x_kx_{k-1}...x_n$$

3. Присоединение случайного гена s из совокупности всевозможных значений генов к концу последовательности:

$$x_1x_2...x_{k-1}x_kx_{k+1}...x_n \rightarrow x_1x_2...x_{k-1}x_kx_{k+1}...x_n, s$$

4. Вставка случайного гена s из совокупности всевозможных значений генов в случайно выбранную позицию в последовательности

$$x_1x_2...x_{k-1}x_kx_{k+1}...x_n \rightarrow x_1x_2...x_{k-1}x_k, s, x_{k+1}...x_n$$

5. Удаление случайного гена x_k из последовательности:

$$x_1x_2...x_{k-1}x_kx_{k+1}...x_n \rightarrow x_1x_2...x_{k-1}x_{k+1}...x_n$$

Следует заметить, что для особей с фиксированным размером (количество генов в последовательности) мутации 3 и 4 должны применяться в сочетании с мутацией 5.

Мутация особей с вещественными генами

Для мутации особей с вещественными генами необходимо определить величину шага мутации — число, на которое изменится значение гена при мутировании.

Оптимальный размер шага должен меняться в течение всего процесса поиска. Наиболее пригодны маленькие шаги, но иногда большие шаги могут привести к ускорению процесса. Гены могут мутировать согласно следующему правилу:

$$\text{новая переменная} = \text{старая переменная} \pm \alpha \cdot \delta,$$

где знаки + или − выбираются с равной вероятностью; α = 0,5×поисковое пространство,

$$\delta = \sum_{i=1}^{m} a(i)\, 2^{-i},$$

$\alpha(i) = 1$ с вероятностью $\frac{1}{m}$, в противном случае $\alpha(i) = 0$, m – параметр.

Новая особь, получившаяся при такой мутации, в большинстве случаев не намного отличается от старой. Это связано с тем, что вероятность маленького шага мутации выше, чем вероятность большого шага. При m = 20, данный алгоритм мутации пригоден для локализации оптимума с точностью $\alpha \cdot 2^{-19}$.

6.3.4. Селекция

Оператор селекции − это процесс, посредством которого особи с «лучшими» признаками, получают большую возможность для воспроизводства (репродукции) потомков, чем «худшие» особи. Оператор селекции является одним из наиболее важных операторов, помогающих поддерживать генетическое разнообразие популяции

В результате применения селекции особи в соответствии с их пригодностью выбираются для порождения потомков или для перехода в следующее поколение. Наиболее приспособленные особи должны выбираться с большей вероятностью для сохранения своих генов в следующем поколении. Таким образом, оператор селекции позволяет сконцентрировать поиск в перспективных областях.

Существует два основных подхода к селекции:

1. Получившиеся в процессе скрещивания особи (потомки), заменяют собой родительские особи. Лишь затем происходит вычисление приспособленности особей популяции. Таким образом, происходит замена, возможно приспособленных, родителей на потомков неопределенной приспособленности. Это является огромным недостатком данной стратегии.

Достоинством же является высокая производительность, так как на каждой итерации происходит обработка минимального количества особей.

2. Все особи (и родительские и дочерние), после процедуры скрещивания, помещаются в промежуточный массив, который сортируется по приспособленности. Затем, выбираются наиболее приспособленные особи, независимо от того к какому поколению они принадлежат. Этот подход не имеет основного недостатка предыдущего (замена, возможно приспособленных, предков) и это является его наибольшим достоинством. Отрицательные же черты данного подхода

– во-первых, на каждой итерации приходится обрабатывать массив, превышающий размером популяцию в два раза, что, при большом размере популяции может оказаться серьезным недостатком;

– во-вторых, благодаря тому, что на каждой итерации выбирается наилучшее решение, повышается вероятность преждевременной сходимости.

Существует множество различных стратегий селекции. Конкретный тип оператора селекции (а также и других генетических операторов) выбирается исходя из особенностей решаемой задачи. Наиболее распространены следующие базовые стратегии селекции.

1. Селекция по методу рулетки. Каждой хромосоме может быть сопоставлен сектор колеса рулетки, величина которого устанавливается пропорциональной значению функции приспособленности данной хромосомы. Поэтому чем больше значение функции приспособленности, тем больше сектор на колесе рулетки

Каждой i-й особи, $i = 1...N$ соответствует сектор колеса рулетки, выраженный в процентах:

$$V_i = P_i \times 100\%,$$

где P_i - вероятность выбора i-й особи, которая определяется следующим образом:

$$P_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$$

Таким образом, вероятность выбора особи тем выше, чем больше значение её функции приспособленности. Очевидно, что для всей популяции выполняется условие $\sum_{i=1}^{N} P_i = 1$.

В рамках общей стратегии метода рулетки существуют и другие способы вычисления вероятности попадания особи в новую популяцию. В частности, вероятности P_i при решении могут вычисляться следующим образом:

$$P_i = \frac{(f^* - f_i)}{\sum_{j=1}^{N} (f^* - f_j)},$$

где f^* - наихудшее значение целевой функции среди особей текущего поколения. При этом предполагается, что решается задача минимизации. Тогда наихудшее значение f^* - это максимальное значение целевой функции.

2. Турнирная селекция. Из популяции, состоящей из N особей, создается группа из t ($t \geq 2$) особей, выбранных случайным образом. Особь с наибольшей пригодностью в группе отбирается, остальные – отбрасываются. Такая операция повторяется k раз. Затем отобранные особи используются для кроссинговера. Размер группы t часто равен 2, в таких случаях говорят о парных (двоичных) турнирах. Число t называется численностью турнира.

3. Отбор усечением. При отборе усечением используют популяцию, состоящую как из особей-родителей, так и особей потомков, отсортированную по возрастанию значений функции пригодности особей. Число особей для скрещивания выбирается в соответствии с порогом $T \in [0; 1]$. Порог определя-

ет, какая доля особей, начиная с самой первой (самой пригодной), будет принимать участие в отборе. Порог можно задать и числом больше 1, тогда он будет просто равен числу особей из текущей популяции, допущенных к отбору. Среди особей, удовлетворяющих пороговому условию, случайным образом выбирается одна и записывается в новую популяцию. Процесс повторяется N раз, пока размер новой популяции не станет равен размеру исходной популяции. Новая популяция состоит только из особей с высокой пригодностью, причем одна и та же особь может встречаться несколько раз, а некоторые особи, имеющие пригодность выше пороговой, могут не попасть в новую популяцию.

4. Элитарный отбор. Создается промежуточная популя ция, которая включает в себя как родителей, так и их потомков. Члены этой популяции оцениваются, а за тем из них выбираются N самых лучших (пригодных), которые и войдут в следующее поколение. Зачастую данный метод комбинируют с другими — выбирают в новую популяцию, например, 10 % «элитных» особей, а остальные 90 % — одним из традиционных методов селекции. Иногда эти 90 % особей создают случайно, как при создании начальной популяции перед запуском работы генетического алгоритма. Данный метод хорош с той точки зрения, что исключает «случайное блуждание по пространству поиска», поскольку осуществляется переход в следующее поколение самой лучшей особи (найденной на данном этапе поиска или ранее);

5. *Отбор с вытеснением.* В данном отборе выбор особи в новую популяцию зависит не только от величины ее пригодности, но и от того, есть ли уже в формируемой популяции особь с аналогичным хромосомным набором. Отбор проводится из числа родителей и их потомков. Из всех особей с одинаковой приспособленностью предпочтение сначала отдается особям с разными генотипами. Таким образом, достигаются две цели: во-первых, не теряются лучшие найденные решения, обладающие различными хромосомными наборами, во-вторых, в популяции постоянно поддерживается генетиче-

ское разнообразие. Вытеснение в данном случае формирует новую популяцию скорее из удаленных особей, вместо особей, группирующихся около текущего найденного решения. Данный метод наиболее пригоден для многоэкстремальных задач, при этом помимо определения глобальных экстремумов появляется возможность выделить и те локальные максимумы, значения которых близки к глобальным.

6.4. Пример работы генетического алгоритма

Рассмотрим принципы работы генетических алгоритмов на примере. Пусть требуется найти минимум функции

$$y(x) = 5 - 24x + 17x^2 - \frac{11}{3}x^3 + \frac{1}{4}x^4$$

на отрезке [0; 7] (рис. 6.7). На этом отрезке функция принимает минимальное значение в точке x = 1. Очевидно, что в точке x = 6 функция попадает в локальный минимум. Если для нахождения глобального минимума использовать градиентные методы, то в зависимости от начального приближения можно попасть в данный локальный минимум.

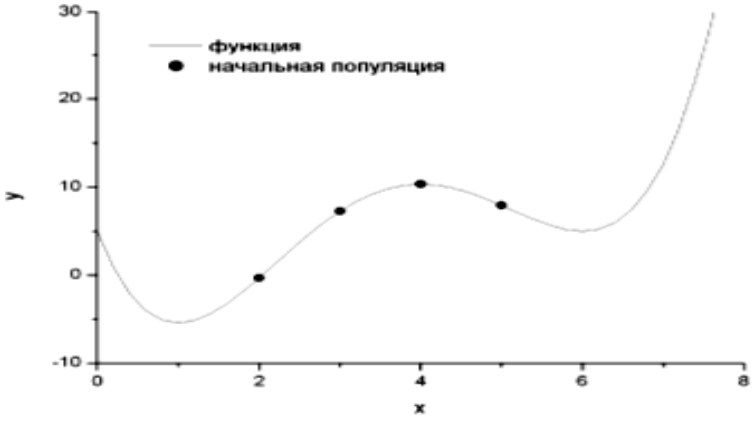


Рис. 6.7. График целевой функции с выбранными значениями пробных решений

Для простоты положим, что x принимает целые значения, т.е. x ∈ {0, 1, 2, 3, 4, 5, 6, 7}. Выберем случайным образом несколько чисел на отрезке [0; 7]: {2, 3, 5, 4}. Будем рассматривать эти числа в качестве пробных решений нашей задачи.

Запишем пробные решения в двоичной форме: {010, 011, 101, 100}. Таким образом, мы имеем популяцию, состоящую из четырех особей (хромосом). Приспособленность особи определяется целевой функцией (функцией приспособленности): чем меньше значение целевой функции, тем более приспособленной является особь, т.е. пробное решение, использовавшееся в качестве аргумента целевой функции. Поставим в соответствие каждой особи исходной популяции случайное целое число из диапазона от 1 до 4. Будем рассматривать эти числа как номера членов популяции.

Исходная популяция представлена в таблице 6.3.

Таблица 6.3

№	Особь		
	Целое число	Двоичное число	Приспособленность
1	2	010	−0,33
2	3	011	7,25
3	5	101	7,92
4	4	100	10,33

Следующим этапом является этап размножения. Для этого сформируем из исходной популяции брачные пары для скрещивания. При таком выборе какие-то из членов популяции не будут участвовать в процессе размножения, так как образуют пару сами с собой. Какие-то члены популяции примут участие в процессе размножения неоднократно с различными особями популяции. Процесс размножения (рекомбинация) заключается в обмене участками хромосом между родителями.

Для нашей популяции процесс создания первого поколения потомков показан в таблице 6.4. В качестве механизма рекомбинации выбран одноточечный кроссинговер.

Таблица 6.4

№	Особь популяции	Выбранный номер	Вторая особь-родитель	Точка кроссинговера	Особи-потомки
1	010	4	100	1	000
2	011	2	011		110
3	101	1	010	2	100
4	100	4	100		011

Следующим шагом в работе алгоритма являются мутации. Пусть вероятность мутации равна 0,3. Для каждого потомка возьмем случайное число на отрезке [0; 1], и если это число меньше 0,3, то инвертируем случайно выбранный ген (заменим 0 на 1 или наоборот) (таблица 6.5).

Таблица 6.5

№	Особи-потомки	Случайное число	Выбранный ген для мутации	Потомок после мутации	Приспособленность потомка до мутации	Приспособленность потомка после мутации
1	000	0,1	3	001	5	−5,42
2	110	0,6	–	110	5	5
3	100	0,5	–	100	10,33	10,33
4	011	0,2	1	111	7,25	12,58

Как видно на примере, мутации способны улучшить (первый потомок) или ухудшить (четвертый потомок) приспособленность особи-потомка. В результате скрещивания хромосомы обмениваются «хвостами», т.е. младшими разрядами в двоичном представлении числа. В результате мутаций изменению может подвергнуться любой разряд, в том числе, старший. Таким образом, если скрещивание приводит к относительно небольшим изменениям пробных решений, то мутации могут привести к существенным изменениям значений пробных решений (рис. 6.8)

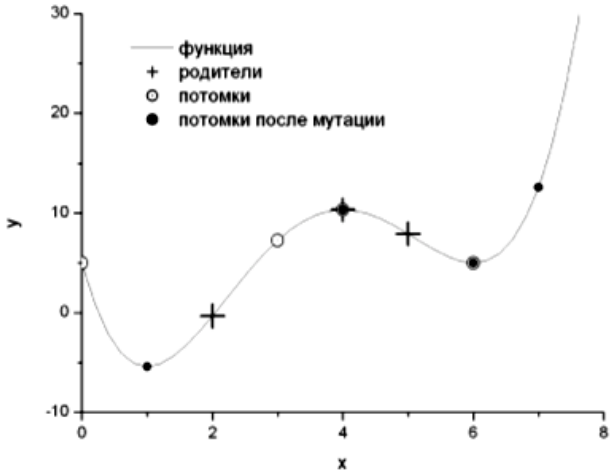


Рис. 6.8. Изменение популяции в процессе естественного отбора

Теперь из четырех особей-родителей и четырех полученных особей потомков необходимо сформировать новую популяцию. В новую популяцию отберем четыре наиболее приспособленных особи из числа «старых» особей и особей-потомков (таблица 6.6).

Таблица 6.6

№	Особи	Приспособленность	Новая популяция	Приспособленность особей в новой популяции
1	010	−0,33	001	−5,42
2	011	7,25	010	−0,33
3	101	7,92	110	5
4	100	10,33	011	7,25
5	001	−5,42		
6	110	5		
7	100	10,33		
8	111	12,58		

В результате получим новое поколение, которое представлено на рис. 6.9.

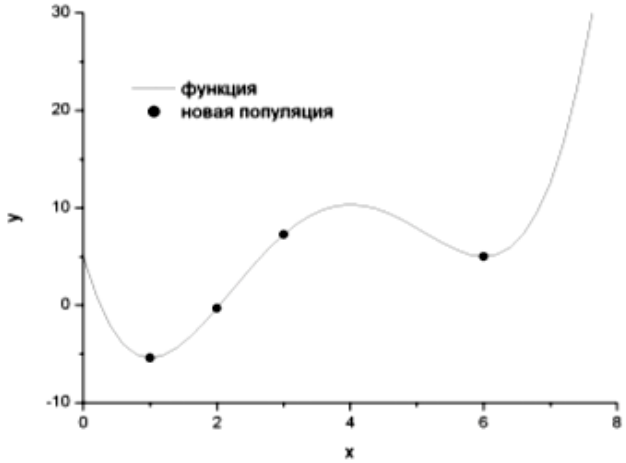


Рис. 6.9. Минимизируемая функция и особи новой популяции

Получившуюся популяцию можно будет вновь подвергнуть кроссинговеру, мутации и отбору особей в новое поколение. Таким образом, через несколько поколений мы получим популяцию из похожих и наиболее приспособленных особей. Значение приспособленности наиболее «хорошей» особи (или средняя приспособленность по популяции) и будет являться решением нашей задачи. Следуя этому, в данном случае, взяв наиболее приспособленную особь 001 во втором поколении, можно сказать, что минимумом целевой функции является значение −5,42, соответствующее аргументу x = 1. Тем самым попадания в локальный минимум удалось избежать. На данном примере разобран вариант простого генетического алгоритма. При дальнейшем использования ГА к разным задачам возможно моделирование основных операторов алгоритма.

7. РЕШЕНИЕ ЗАДАЧ МНОГОКРИТЕРИАЛЬНОЙ ОПТИМИЗАЦИИ

7.1. Парето-оптимальность в задачах многокритериальной оптимизации

При оптимизации сложных систем формализовать требования к оптимизируемому объекту в виде одного критерия оптимальности бывает достаточно трудно, а зачастую и невозможно. В этом случае оптимизацию производят по нескольким частным критериям $f_i(X)$ $(i=1,2,...,s)$, а полученные задачи называются *задачами многокритериальной оптимизации*. Набор частных критериев оптимальности в многокритериальных задачах образует векторный критерий $(f_1(X), f_2(X),..., f_s(X))$, поэтому часто такие задачи называют *задачами векторной оптимизации*.

Обобщенная постановка задачи:

$$f(x) = (f_1(x),..., f_s(x)) \to \min,$$
$$x \in D.$$

Здесь D – множество допустимых решений, представленная системой прямых и функциональных ограничений. Допустимые векторы X**,** принадлежащие допустимой области D, называются *альтернативами* или *вариантами*. Поэтому множество D называют еще *множеством альтернатив, вариантов, планов, стратегий*.

Для всякого решения (альтернативы) $X \in D$ набор его оценок по всем критериям, т.е. набор $(f_1(X), f_2(X),..., f_s(X))$ есть векторная оценка решения X.

Оценка альтернатив по критериям осуществляется либо в автоматическом режиме, либо в режиме диалога с ЛПР. Лицом, принимающим решения (ЛПР), называют человека (или группу людей), имеющего цель, которая служит мотивом постановки задачи и поиска её решения. ЛПР является

компетентным специалистом в своей области и обладает опытом деятельности в ней, наделено необходимыми полномочиями и несёт ответственность за принятое решение.

Сравнение двух работоспособных вариантов с параметрами X^1 и $X^2 \in D$ производится на основании следующих бинарных отношений, определенных на множестве допустимых решений D:

- отношения строгого предпочтения , выполнение которого $X^1 \prec X^2$ (X^1 строго превосходит X^2) содержательно интерпретируется как: вариант с параметрами X^1 лучше варианта с параметрами X^2 (или альтернатива X^1 лучше альтернативы X^2);

- отношения нестрогого предпочтения, выполнение которого $X^1 \le X^2$ (X^1 нестрого превосходит X^2) содержательно интерпретируется как: вариант с параметрами X^1 не хуже варианта с параметрами X^2;

- отношения эквивалентности, выполнение которого $X^1 \approx X^2$ (X^1 эквивалентен X^2) содержательно интерпретируется как: вариант с параметрами X^1 равноценен варианту с параметрами X^2;

Если относительно пары альтернатив нельзя сказать, какая из них лучше, то их называют *несравнимыми.*

Важную роль в многокритериальной оптимизации играет понятие *независимости критериев по предпочтению.* Смысл его состоит в том, что желательные для ЛПР изменения значений каждого из частных критериев (при неизменных значениях остальных критериев) не должны зависеть от конкретных значений остальных критериев.

Рассмотрим многокритериальную задачу с независимыми по предпочтению критериями.

Говорят, что альтернатива $X^1 \in D$ *доминирует (по Парето)* альтернативу $X^2 \in D$, (будем обозначать $X^1 \succ X^2$) если

$$f_i(X^1) \geq f_i(X^2),\ i = \overline{1,m}$$

и $\exists i_0 = \{1,m\}$, такое, что $f_{i_0}(X^1) > f_{i_0}(X^2)$

То есть, альтернатива X^1 доминирует по Парето альтернативу X^2, если по всем критериям оценки альтернативы X^1 не хуже, чем альтернативы X^1, а хотя бы по одному критерию оценка X^1. При этом альтернатива X^2 называется доминируемой.

В случае доминирования при переходе от X^2 к X^1 ничего не будет проиграно ни по одному из частных критериев, но в отношении частного критерия $\exists i_0 = \{1,m\}$ точно будет получен выигрыш. Говорят, что решение X^1 лучше (предпочтительнее) решения X^2.

Доминируемое решение X^2 называется *неэффективным решением*. Множество неэффективных решений образует множество отбрасываемых вариантов

Точка $X^* \in D$ называется *оптимальной по Парето* (*эффективной*), если не существует альтернативы $X \in D$, доминирующей X^*.

Эффективные решения будут несравнимы по векторному критерию оптимальности в смысле рассмотренных бинарных отношений строгого предпочтения, нестрогого предпочтения и эквивалентности. Таким образом, эффективное решение- это такой вариант с параметрами X^*, относительно которого в области допустимых решений D нельзя найти ни одного безусловно лучшего варианта (не существует ни одного доминирующего решения). Оптимальность по Парето означает, что нельзя уменьшить значения некоторых частных кри-

териев оптимальности, не увеличивая при этом хотя бы одного из остальных.

Множество таких точек и называется *множеством точек оптимальных по Парето.* Множество точек оптимальных по Парето лежат между точками оптимумов, полученных при решении задачи для каждого частного критерия.

Множество векторных оценок, соответствующих множеству неэффективных решений, называется *областью согласия.*

Обозначим $D^C \subset D$ *область согласия*, которая не содержит недоминируемых точек (любая точка из D^c может быть улучшена).

Множество векторных оценок, соответствующих множеству эффективных точек, называют областью *компромиссов* (*переговорным множеством*) или множеством Парето в области критериев. D_k. *Эффективная область* (*множество Парето*) $D^э \subseteq D$ содержит все оптимальные по Парето точки.

Значение векторного критерия оптимальности $F(X^*) \in D_k$, соответствующего конкретному эффективному решению X^*, называется *вектором Парето*. Для любых двух эффективных решений X^* и X^{**} соответствующие эм вектора Парето являются противоречивыми критериями. Это приводит к необходимости введения соглашения о компромиссе между векторами Парето.

Оптимальные решения многокритериальной задачи следует искать только среди элементов множества альтернатив $D^э$. В этой области ни один критерий не может быть улучшен без ухудшения хотя бы одного из других. Важным свойством множества Парето $D^э$ является возможность «выбраковывать» из множества альтернатив D заведомо неудачные, уступающие другим по всем критериям. При этом под оптимально-компромиссным решением будем понимать одну из эффективных точек, являющуюся предпочтительней с точки зрения ЛПР.

7.2. Основные подходы к алгоритмизации задач многокритериальной оптимизации

Все известные методы векторной оптимизации непосредственно или косвенно сводят решаемые задачи с несколькими критериями к задачам скалярной оптимизации. При этом используются два основных подхода:

- свертка локальных критериев в обобщенный интегральный показатель;
- сведение многокритериальных задач к последовательности специальным образом сформулированных задач параметрической оптимизации.

7.2.1. Формирование обобщенного критерия оптимальности

При использовании первого подхода частные критерии $f_i(X)$, $i=\overline{1,s}$ объединяются в составной критерий $F(X)=\Phi(F_1(X),...F_s(X))$, который затем минимизируется или максимизируется. Обобщенный критерий оптимальности может быть использован только в том случае, если частные критерии оптимальности удовлетворяют следующим требованиям:

- частные критерии оптимальности соизмеримы по важности, т. е. каждому из них можно поставить в соответствие некоторое неотрицательное число, которое характеризует его относительную важность по отношению к другим частным критериям;
- частные критерии являются однородными (нормализованными) критериями, либо имеют единый масштаб измерения.

При этом в качестве обобщенных критериев используются функции следующего вида:

а) аддитивный критерий оптимальности:

$$F(X)=\sum_{i=1}^{s} w_i f_i(X);$$

б) мультипликативный критерий оптимальности:

$$F(X)=\prod_{i=1}^{s} f_i^{w_i}(X);$$

в) среднестепенной обобщенный критерий оптимальности:

$$F(X)=\left[\frac{1}{S}\sum_{i=1}^{s} w_i f_i^{p}(X)\right]^{1/p}.$$

Здесь w_i – весовые коэффициенты, удовлетворяющие условию:

$$\sum_{i=1}^{s} w_i=1.$$

При этом величина w_i определяет важность i-го критерия и задает в количественном измерении предпочтение i-го показателя над другими.

Наиболее распространенным на практике является аддитивный критерий оптимальности. При этом решение многокритериальной задачи может быть сведено к минимизации аддитивной функции:

$$\min_{X\in D}\sum_{i=1}^{s} w_i f_i(X), \quad \text{где } w_i\geq 0,\ \sum_{i=1}^{s} w_i=1.$$

Этот метод свертки векторного критерия, называемый *методом взвешенных сумм*, позволяет назначать приоритеты более важным частным критериям оптимальности за счет увеличения для них значений w_i.

Таким образом, при выборе обобщенного критерия оптимальности осуществляется переход от субъективизма при выборе предпочтения между частными показателями к субъективизму, связанному с назначением численных значений

весовых коэффициентов $w_i (i=1,2,...,s)$.

Для равноценных критериев, то есть критериев, для которых невозможно установить приоритет по важности, значения весовых коэффициентов выбираются одинаковыми:

$$w_i = \frac{1}{S},\ i=1,2,...,s.$$

Для неравноценных критериев значения весовых коэффициентов определяются в соответствии с важностью критерия.

В настоящее время существует большое количество различных способов и приемов, позволяющих по информации о значениях частных критериев оптимальности определять значения весовых коэффициентов w_i. Наиболее простой в реализации подход заключается в вычислении для каждого частного критерия оптимальности $f_i(X),\ i=1,...,s$ коэффициента относительного разброса:

$$\delta_i = \frac{f_i^+ - f_i^-}{f_i^+} = 1 - \frac{f_i^-}{f_i^+},$$

где $f_i^- = \min\limits_{X \in D} f_i(X),\ \ f_i^+ = \max\limits_{X \in D} f_i(X)$.

Параметр δ_i определяет максимально возможное отклонение по i-му частному критерию. При этом весовые коэффициенты w_i получают наибольшее значение для тех критериев, относительный разброс которых в допустимой области D наиболее значителен:

$$w_i = \frac{\delta_i}{\sum\limits_{k=1}^{s} \delta_k}\ \ (i=1,...,s).$$

Если все $f_i^- \neq 0,\ \ i=1,2,...,s$, можно рассматривать коэффициенты

$$\beta_i(X) = \frac{f_i(X) - f_i^-}{f_i^-},$$

которые характеризуют отклонение частного критерия оптимальности от его наименьшего значения.

Широкое распространение при определении весовых коэффициентов критериев получили также экспертные методы, основанные на направленном опросе ЛПР с последующей обработкой полученных результатов. При этом используются различные процедуры ранжирования, алгоритмы логического упорядочивания и т. д. Особую роль при агрегировании критериев играет использование аппарата нечетких множеств и нечетких отношений, который используется для формализации нечеткой и неполной информации, а также суждений экспертов, заданных в лингвистической форме. Большое количество методов настройки весовых коэффициентов критериев представлено в работах [2,7].

7.2.2. Методы последовательной оптимизации

Использование обобщенного интегрированного показателя, как правило, приводит к трудностям при оптимизации, так как значительно усложняется рельеф целевой функции. Поэтому на практике зачастую используются другие соглашения о компромиссе, которые позволяют многокритериальную задачу свести к одной или нескольким задачам параметрической оптимизации, не вводя в рассмотрение тот или иной обобщенный критерий оптимальности.

Одним из наиболее распространенных подходов является *метод главного критерия*. При этом ЛПР (лицо, принимающее решение) среди частных критериев оптимальности $f_i(X)$, $i = \overline{1,s}$ выделяет наиболее предпочтительный показатель (например, $f_t(X)$). По остальным критериям $f_i(X)$, $i \neq t$ ЛПР устанавливает пороговые значения $f_i^p(X)$, которые не должны превышаться, после чего эти критерии переводятся в

систему ограничений. В результате оптимизация проводится по одному главному критерию $f_t(X)$, а к системе ограничений задачи добавляются следующие ограничения:

$$f_i(X) \le f_i^p(X), \quad i \ne t, \quad i = 1,...,s.$$

В процессе оптимизации может происходить смена главного критерия на основании субъективных предпочтений ЛПР.

Основной трудностью метода главного критерия является назначение пороговых значений $f_i^p(X)$, от выбора которых зависит окончательное решение задачи. Определение пороговых значений осуществляется в режиме диалога с ЛПР.

При решении многокритериальных задач также широко используются *методы последовательной оптимизации*, в соответствии с которыми критерии предварительно упорядочиваются по важности и осуществляется последовательное решение s задач параметрической оптимизации, начиная с самого главного критерия:

$$\begin{aligned} f_1^* &= \min_{X \in D} f_1(X) \\ f_i^* &= \min_{X \in D_i} f_i(X), \quad i = \overline{2,s,} \end{aligned} \tag{7.1}$$

где

$$D_i = D \cap D_{i-1};$$

$$D_{i-1} = \{X \mid f_j(X) = f_j^*, \ j = \overline{1,i-1}\}$$

В качестве оптимально-компромиссного решения принимается оптимальное решение, полученное на s-м шаге итерационного процесса последовательной параметрической оптимизации. При использовании рассмотренного подхода минимизация k-го частного критерия оптимальности начинается только тогда, когда получены минимальные значения всех предыдущих k-1 критериев оптимальности. При этом области

допустимых решений D_i, $i=\overline{1,s-1}$ образуют сужающиеся подмножества:

$$D_{s-1} \subseteq ... \subseteq D_2 \subseteq D_1 \subseteq D.$$

Недостатком данного метода является то, что он не позволяет справедливо учитывать интересы менее важных частных критериев, так как не допускает их уменьшения, если это вызывает хотя бы незначительное увеличение более важных критериев.

Этот недостаток в некоторой степени устраняется в *методе последовательных уступок*. При использовании этого метода на каждом k-м шаге последовательной оптимизации (7.1) вводится уступка Δf_{k-1}, характеризующая допустимое отклонение (k-1)-го частного критерия оптимальности от его минимального значения f_{k-1}^*:

$$\begin{aligned} f_1^* &= \min_{X \in D} f_1(X) \\ f_i^* &= \min_{X \in D_i} f_i(X), \quad i=\overline{2,s,} \end{aligned} \tag{7.2}$$

где

$$D_i = D \cap D_{i-1};$$

$$D_{i-1} = \{X \mid f_j(X) \le f_j^* + \Delta f_j,\ j=\overline{1,i-1}\}$$

Введение уступки по (k-1)-му наиболее важному частному критерию позволяет улучшить значение k-го менее важного частного критерия.

Развитием метода последовательных уступок является процедура, в которой уступки Δf_k, $k=\overline{1,s}$ упорядочиваются по важности с помощью весовых коэффициентов $\lambda_i > 0, i=\overline{1,s}, \sum_{i=1}^{s} \lambda_i = 1$. Реализация данного соглашения о компромиссе, называемая *методом минимизации уступок*, может быть сведена к задаче параметрической оптимизации:

$$\min\{\sum_{k=1}^{s}\lambda_k\Delta f_k\} \qquad (7.3)$$

при условии, что

$$(f_k(X)-f_k^*)/f_k^* \le \Delta f_k, \quad k=\overline{1,s}$$
$$X \in D$$

Если все критерии являются равноценными ($\lambda_i = 1/s, \quad i=\overline{1,s}$), а все уступки одинаковы ($\Delta f_k = \Delta f, \ k=\overline{1,s}$), задача параметрической оптимизации (7.3) примет следующий вид:

$$\min \Delta f_k \qquad (7.4)$$

при условии, что

$$(f_k(X)-f_k^*)/f_k^* \le \Delta f_k, \quad k=\overline{1,s}$$
$$X \in D$$

Решение экстремальной задачи (7.4) позволяет получить оптимально-компромиссное решение, которое минимизирует наибольшее отклонение каждого k-го частного критерия оптимальности от его минимально возможного значения:

$$\min_{X\in D}\{\max_{1\le k\le s}((f_k(X)-f_k^*)/f_k^*)\} \qquad (7.5)$$

Данный подход к определению оптимально-компромиссного решения получил в литературе название *метода гарантированного результата*. Решение задачи (7.7) обеспечивает наибольшую равномерность отклонения значений частных критериев от их минимально допустимых значений за счет подтягивания “наихудшего” частного критерия до уровня остальных. Вследствие того, что операции проводятся в области компромисса, подтягивание “отстающего” критерия неизбежно приводит к ухудшению значений части остальных критериев. Но при проведении ряда шагов можно добиться определенной степени уравнивания противоречивых (конфликтных) частных критериев.

Для несоизмеримых по важности частных критериев можно принять соглашение о том, что они являются равноценными. В этом случае используется *принцип равенства частных критериев*:

$$\min_{X \in D} f_1(X) \tag{7.5}$$

при условии, что

$$f_1(X) = f_2(X) = ... = f_s(X)$$

При использовании данного принципа необходимо учитывать, что в некоторых случаях задача параметрической оптимизации (7.8) может оказаться неразрешимой, либо ее оптимальное решение не будет эффективным для исходной многокритериальной задачи.

Из рассмотренных в данной главе подходов к определению оптимально-компромиссного решения в задачах векторной оптимизации видно, что с помощью различных преобразований многокритериальные задачи сводятся к задачам скалярной оптимизации. Для решения полученных однокритериальных задач могут использоваться методы, изложенные в главах 2 – 5 настоящего пособия. Эти методы составляют основу инвариантного математического обеспечения для решения оптимизационных задач.

8. НЕКОТОРЫЕ ПРИКЛАДНЫЕ ЗАДАЧИ ОПТИМИЗАЦИИ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

Рассмотрим примеры прикладных оптимизационных задач, использующихся при поиске оптимальных проектных и управленческих решений в автоматизированных системах.

Задачи производственного планирования

Задачи производственного планирования широко используются в автоматизированных системах управления производством. Рассмотрим некоторые обобщённые оптимизационные модели для решения задач производственного планирования.

1. Задача определения оптимального производственного плана. Для изготовления n видов продукции $A_1,\ldots,A_n$ необходимы ресурсы $S_1,\ldots,S_m$ (сырье, рабочая сила, оборудование и т. д.). Заданы значения a_{ij}, которые характеризуют количество ресурса S_i, необходимого для выпуска единицы продукции A_j. Запас каждого ресурса S_i ограничен и равен b_i . От реализации единицы продукции A_j может быть получена прибыль c_j. Необходимо так составить производственный план (т. е. определить, сколько продукции каждого вида необходимо выпустить), чтобы общая прибыль от производства всей продукции была максимальной.

Обозначим через x_j количество продукции A_j, которое необходимо выпустить по плану. Тогда математическая модель данной задачи имеет следующий вид:

$$c_1x_1+ c_2x_2+\ldots+ c_nx_n \to \max;$$

$$\sum_{j=1}^{n} a_{ij}x_j \le b_i,\ i=\overline{1,m};$$

$$x_j \ge 0,\ j=\overline{1,n}.$$

Здесь целевая функция – это общая прибыль от производства всей продукции. Каждое i-е ограничение характеризует общее количество ресурса S_i, которое не должно превышать величины b_i.

Пример задачи производственного планирования и ее решение симплекс-методом приведены в разделе 3.2.

Кроме ограничения по ресурсам, в условия задачи (а, следовательно, и в ее математическую модель) могут вводиться дополнительные ограничения на планируемый выпуск продукции (ограничения по ассортименту, условия комплектности и т. д.).

2. Задача о комплектах.

Пусть планируется производство комплектных изделий. Каждое изделие содержит m видов деталей, причем деталей первого вида – k_1 единиц, деталей второго вида – k_2 единиц, деталей j-го вида – k_j единиц. Изделия могут изготавливаться n различными исполнителями. Пусть в единицу времени i-й исполнитель может изготовить a_{ij} элементов j-го вида. Сдаче подлежат только комплектные изделия. Каждый i-й исполнитель работает не более b_i часов. Определить такой план загрузки исполнителей, чтобы общее число выпускаемых комплектных изделий было максимальным.

Обозначим через x_{ij} время изготовления деталей j-го вида i-м исполнителем. Тогда общее число деталей j-го вида, изготавливаемых всеми исполнителями, определится в виде

$$D_j = a_{1j}x_{1j} + \ldots + a_{nj}x_{nj}, \ \forall j = \overline{1,m}.$$

Общее количество изделий, для выпуска которых хватит деталей j-го типа при условии, что k_j единиц расходуется на одно изделие равно

$$I_j = \frac{D_j}{k_j} = \frac{a_{1j}x_{1j} + \ldots + a_{nj}x_{nj}}{k_j}.$$

Тогда общее число комплектов определится следующим образом:

$$\min_{j=1,m} I_j$$

В результате математическую модель задачи можно сформулировать в виде:

$$\max_{x_{ij}}\left(\min_{j=1,m} I_j\right) = \max_{x_{ij}}\left(\min_{j=1,m} \frac{a_{1j}x_{1j} + \ldots + a_{nj}x_{nj}}{k_j}\right);$$

$$x_{i1} + \ldots + x_{im} \le b_i,\ i = \overline{1,n};$$

$$x_{ij} \ge 0, i = \overline{1,n},\ j = \overline{1,m}.$$

Для формализации данной постановки в виде задачи линейного программирования введем новую переменную:

$$y = \min_{j=1,m} \frac{\sum_{i=1}^{n} a_{ij}x_{ij}}{k_j},$$

что равносильно записи

$$y \le \frac{\sum_{i=1}^{n} a_{ij}x_{ij}}{k_j} \qquad \forall j = \overline{1,m}.$$

Тогда окончательный вид математической модели задачи:

$$y \to \max$$

$$\begin{cases} \sum_{i=1}^{n} a_{ij}x_{ij} - yk_j \ge 0,\ j = \overline{1,m} \\ \sum_{j=1}^{m} x_{ij} \le b_i,\ i = \overline{1,n} \\ x_{ij} \ge 0,\ i = \overline{1,n},\ j = \overline{1,m} \end{cases}$$

3. Задача о раскрое. Для изготовления заготовок $D_1,\ldots,D_m$ имеется n способов раскроя материала $A_1,\ldots,A_n$. При этом a_{ij} – количество заготовок i-го типа, получаемых из единицы материала при способе A_j. Имеется D единиц материала. При раскрое единицы материала по способу A_j имеются отходы площадью c_j. Необходимо произвести b_i заготовок i-го типа и выполнить план с минимизацией отходов.

Пусть x_j – количество единиц материала, раскраиваемого по j-му способу.

Математическая модель задачи имеет вид:

$$\sum_{j=1}^{n} c_j x_j \to \min;$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i; \qquad i = \overline{1,m};$$

$$\sum_{j=1}^{n} x_j = D;$$

$$x_j \geq 0, \ j = \overline{1,n}, \ x_j - \text{целые}.$$

Задачи транспортного типа

Одним из наиболее значимых классов задач линейной оптимизации являются задачи транспортного типа.

Пусть имеется m пунктов отправления $A_1,A_2,\ldots,A_m$, в которых сосредоточен однородный груз в количествах $a_1,a_2,\ldots,a_m$ единиц, и n пунктов назначения $B_1,B_2,\ldots,B_n$, потребности которых в данном грузе равны $b_1,b_2,\ldots,b_n$. Стоимость перевозки единицы груза из пункта A_i в пункт B_j (тариф) составляет c_{ij} единиц. Требуется составить план перевозок, позволяющий вывезти все грузы из пунктов отправления, удовлетворить все потребности в грузах в пунктах назначения и имеющий минимальную стоимость.

Обозначим через x_{ij} количество единиц груза, перевозимое из i-го пункта отправления (A_i) в j-й пункт назначения (B_j).

Математическая модель задачи имеет следующий вид:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to \min$$

$$\sum_{j=1}^{n} x_{ij} = a_i, \quad i = \overline{1,m} \tag{8.1}$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \quad j = \overline{1,n} \tag{8.2}$$

$$x_{ij} \geq 0$$

Ограничения (8.1) означают, что весь груз должен быть полностью вывезен из пунктов отправления. Ограничения (8.2) означают, что все потребности в грузах должны быть удовлетворены.

Для разрешимости транспортной задачи необходимо, чтобы её модель была закрытой, т.е. общая потребность в грузе в пунктах назначения была равна общему запасу груза в пунктах отправления: $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$. В противном случае модель транспортной задачи должна быть преобразована к закрытому виду путём введения фиктивного пункта отправления (если $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$.) или фиктивного пункта назначения (если $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$.).

Рассмотренная оптимизационная модель может использоваться не только для решения задач перевозки грузов, но и в других предметных областях, например, для решения задач передачи информации в информационно-вычислительных системах.

Задача о ранце

Имеются предметы n видов, причем для каждого предмета j-го вида (j=1,n) известны его объем $a_j > 0$ и стоимость $c_j > 0$. Необходимо определить такой набор предметов, суммарный объем которых не превышал бы заданного числа b, а суммарная ценность была бы максимальной. Эта задача интерпретируется как задача загрузки ранца объема b и называется *одномерной задачей о ранце.*

Введем целочисленные переменные x_j $(j=\overline{1,n})$, значения которых характеризует количество предметов j-го вида, помещенных в ранец. Тогда математическая модель данной задачи имеет вид:

$$\sum_{j=1}^{n} c_j x_j \to \max;$$

$$\sum_{j=1}^{n} a_j x_j \le b;$$

$$x_j \ge 0,\ j=\overline{1,n},\ x_j \text{ - целые }.$$

Если ограничениями могут быть не только объем ранца, но и другие его характеристики b_i, то получим *многомерную задачу о ранце*:

$$\sum_{j=1}^{n} c_j x_j \to \max;$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, i=\overline{1,m};$$

$$x_j \ge 0,\ \ j=\overline{1,n},\ \ x_j \text{ - целые}$$

В случае, если количество предметов j-го вида ограничено и равно d_j, к задаче добавляется ограничение $x_j \le d_j$.

Если d_j=1, то получим *задачу о ранце с булевыми переменными*. Тогда $x_j \in \{0,1\}$, причем $x_j = 1$, если j-й предмет помещен в ранец, $x_j = 0$ в противном случае.

К задаче о ранце сводится широкий класс задач дискретной оптимизации с ограниченными ресурсами. Рассмотрим примеры прикладных задач.

Пример 8.1. Для организации вычислительной лаборатории могут быть использованы четыре типа ЭВМ. Производительность ЭВМ первого типа равна t_1,, второго - t_2, третьего - t_3 , четвертого - t_4 . Стоимость ЭВМ первого типа a_1 тыс.руб, второго - a_2 тыс. руб, третьего - a_3 тыс. руб, четвертого - a_4 тыс. руб. Потребляемая мощность ЭВМ первого типа b_1 Вт, второго - b_2 Вт, третьего - b_3 Вт, четвертого - b_4 Вт. ЭВМ первого типа занимает площадь c_1 кв.м., второго c_2 кв.м., третьего c_3 кв.м., четвертого c_4 кв.м. Необходимо определить, сколько ЭВМ и какого типа надо выбрать, чтобы обеспечить максимальную производительность при ограниченных стоимостных ресурсах d_1 тыс. руб, энергетических ресурсах d_2 Вт и ограниченной площади d_1 кв.м.

Математическая модель задачи имеет вид:

$$\sum_{j=1}^{4} t_j x_j \to \max;$$

$$\sum_{j=1}^{4} a_j x_j \le d_1$$

$$\sum_{j=1}^{4} b_j x_j \le d_2;$$

$$\sum_{j=1}^{4} c_j x_j \le d_3;$$

$$x_j \ge 0,\ j = \overline{1,4},\ x_j \text{ - целые } .$$

Пример 8.2. Для оценки работоспособности систем перед эксплуатацией производится проверка их функционирования. При проверке контролируются отдельные параметры системы, каждый из которых характеризуется вероятностью отказа проверяемых элементов q_j и временем контроля t_j. Так как допустимое время контроля всей системы T ограничено, то для проверки необходимо выбрать параметры, контролирующие наиболее ненадежные элементы и требующие для контроля наименьшего времени.

Пусть n – общее количество параметров. Введем альтернативные бинарные переменные:

$$x_j = \begin{cases} 1, & \text{если выбирается } j\text{-й параметр} \\ 0 & \text{в противном случае} \end{cases}$$

Тогда математическую модель задачи можно сформулировать как одномерную задачу о ранце:

$$\sum_{j=1}^{n} q_j x_j \to \max;$$

$$\sum_{j=1}^{n} t_j x_j \le T;$$

$$x_j \in \{0,1\}.$$

Задача о назначениях

Линейная задача о назначениях. Имеется n исполнителей и n видов работ, которые они могут выполнять. Пусть c_{ij} – производительность i-го исполнителя при выполнении j работы. Каждый исполнитель может выполнять один вид работ, а каждый вид работ может быть выполнен одним исполнителем. Требуется таким образом распределить исполнителей по видам работ, чтобы общая производительность была максимальной.

Введем альтернативные переменные x_{ij} :

$$x_{ij} = \begin{cases} 1, & \text{если } i\text{-й исполнитель назначается на } j\text{-ю работу} \\ 0 & \text{в противном случае} \end{cases}$$

Тогда математическая модель задачи имеет вид:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \to \max;$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = \overline{1,n};$$

$$\sum_{i=1}^{n} x_{ij} = 1, j = \overline{1,n};$$

$$x_{ij} \in \{0,1\}.$$

Иногда линейная задача о назначениях формулируется как задача минимизации, если в качестве c_{ij} выбирается время, затраченное i-м исполнителем на выполнение j-й работы.

Необходимо заметить, что условие целочисленности переменных в задаче о назначениях можно не накладывать, т. к. эта задача является частным случаем транспортной задачи, и при целочисленности правых частей ограничений целочисленность решения обеспечивается автоматически.

К задаче о назначениях сводится широкий круг задач дискретной оптимизации (распределение исполнителей по видам работ, закрепление за станками операций, распределение задач между различными ЭВМ, назначение претендентов на вакантные должности при формировании штатного расписания и т. д).

Пример 8.3. При передаче сообщений по каналу с шумом необходимо каждой букве передаваемого алфавита поставить в соответствие букву принимаемого таким образом,

чтобы сумма вероятностей правильности принимаемых букв была бы максимальна.

Пусть p_{ij} – вероятность соответствия принимаемой j-й буквы передаваемой i-й букве. Введем альтернативные переменные:

$$x_{ij} = \begin{cases} 1, & \text{если j - я принимаемая буква соответствует i - й передаваемой} \\ 0 & \text{в противном случае} \end{cases}$$

Тогда математическая модель будет иметь вид:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}x_{ij} \to \max;$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = \overline{1,n};$$

$$\sum_{i=1}^{n} x_{ij} = 1, j = \overline{1,n};$$

$$x_{ij} \in \{0,1\}.$$

Квадратичная задача о назначениях. Имеется m пунктов, в которых необходимо разместить n различных объектов. В каждом пункте может быть размещен только один объект. Даны расстояния между i-м и j-м пунктами d_{ij}, а также c_{sk} – показатели, характеризующие взаимосвязи между s-м и k-м объектами (стоимость перевозок, число коммукникаций и т.д.). Необходимо определить такое назначение объектов в выделенные пункты, чтобы минимизировать соответствующие затраты.

Математическая модель рассматриваемой задачи имеет следующий вид:

$$\sum_{s=1}^{n-1}\sum_{k=s+1}^{n}\sum_{i=1}^{m}\sum_{j=1}^{m} x_{si}x_{kj}c_{sk}d_{ij} \to \min;$$

$$\sum_{k=1}^{m} x_{sk} = 1, s = \overline{1,n};$$

$$\sum_{s=1}^{n} x_{sk} = 1, k = \overline{1,m};$$

$$x_{sk} \in \{0,1\}.$$

В данном случае целевая функция зависит не от всевозможных назначений, а от всевозможных пар назначений (s-й элемент размещается в i-й пункт, а k-й – в j-й пункт). Целевая функция при этом не является линейной.

Квадратичная задача о назначениях имеет широкий круг практических приложений. Она может быть использована для решения задач размещения (оборудования, предприятий, деталей, и т. д.). Особенно часто эта модель используется в задачах конструкторско-топологического проектирования.

Пример 8.4. Необходимо таким образом разместить n компонентов печатной платы на m позициях монтажного пространства, чтобы суммарная длина электрических соединений между компонентами была бы минимальной.

Предположим, что n=m. Пусть l_{ks} – расстояние между k-й и s-й позициями, m_{ij} – число связей между i-м и j-м компонентами. Введем бинарные переменные:

$$x_{ik} = \begin{cases} 1, \text{если i - й компонент назначается на k - ю позицию} \\ 0 \quad \text{в} \quad \text{противном} \quad \text{случае} \end{cases}$$

Математическая модель может быть записана в виде:

$$\sum_{i=1}^{n-1}\sum_{j=1+i}^{n}\sum_{k=1}^{m}\sum_{S=1}^{m} x_{ik}x_{jS}l_{kS}m_{ij} \to \min;$$

$$\sum_{k=1}^{m} x_{ik} = 1;\ i = \overline{1,n};$$

$$\sum_{i=1}^{n} x_{ik} = 1;\ k = \overline{1,m};$$

$$x_{ik} \in \{0,1\}.$$

Задача коммивояжера

Имеются n пунктов с заданными расстояниями d_{ij} между i-м и j-м пунктами. Необходимо составить оптимальный маршрут из условия минимизации суммарного расстояния для коммивояжера, выходящего из пункта 1, который должен побывать во всех пунктах ровно по одному разу и вернуться в исходный пункт.

Введем альтернативные бинарные переменные:

$$x_{ij} = \begin{cases} 1, \text{если переезд из i-го города в j-й входит в маршрут} \\ 0 \quad \text{в} \quad \text{противном случае} \end{cases}$$

Условия минимизации общего расстояния, а также прибытия в каждый пункт и выезда из него ровно по одному разу могут быть выражены следующим образом:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \to \min;$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = \overline{1,n};$$

$$\sum_{i=1}^{n} x_{ij} = 1, j = \overline{1,n};$$

Однако необходимо обеспечить непрерывность маршрута, чтобы набор звеньев, входящих в маршрут, образовывал единую цепочку (например, при n=8 цепочка (1,2) - (2,6) - (6,4) - (4,8) - (8,5) - (5,3) - (3,7) - (7,1)), а не состоял бы из несвязанных цепочек (например, (1,2) - (2,6) - (6,1) и (3,8) - (8,7) - (7,5) - (5,4) - (4,3)). Для устранения замкнутых циклов (подобходов), включающих количество вершин меньшее n, в модель вводятся n дополнительные переменных $u_i \geq 0$ ($i = \overline{1,n}$) и n^2 дополнительных ограничений:

$$U_i - U_j + (n-1)x_{ij} \leq n-2, i = \overline{2,n}, j = \overline{2,n}.$$

Действительно, пусть маршрут включает несколько цепочек. Тогда существует цепочка, начинающаяся и заканчивающаяся в начальном пункте, но включающая n_1 звеньев, где $n_1<n$. Просуммировав эти неравенства вдоль такой цепочки (т. е. при $x_{ik}=1$), получим бессмысленное неравенство $n_1(n-1)\leq n_1(n-2)$ (все u_i и u_j при суммировании взаимно уничтожаются). Покажем теперь, что для маршрута, исключающего подобходы, это неравенство выполняется, т. е., можно найти значения u_i, удовлетворяющие дополнительным ограничениям. Положим $u_i=p$, если город i посещается на p-м шаге, $p=\overline{2,n}$. Отсюда следует, что $u_i - u_j \leq n-2 \ \forall i, j = \overline{2,n}$. Таким образом, ограничения выполняются для всех $x_{ij} = 0$. При $x_{ij} = 1$ они превращаются в равенства:

$$U_i - U_j + (n-1)x_{ij} = p - (p+1) + n - 1 = n - 2.$$

Задача коммивояжера имеет широкий круг приложений. К ней сводятся задачи оптимальной маршрутизации (выбор маршрутов движения транспорта, минимизация расстояния, проходимого исполнительным механизмом станка с ЧПУ и др.), задачи проектирования электрических и вычислительных сетей, задачи определения последовательности обработки деталей на станках с условием минимизации времени переналадок и т. д.

Пример 8.5. Пусть необходимо проложить коммуникации между n различными ЭВМ таким образом, чтобы каждая ЭВМ была связана с двумя соседними, вся сеть была бы подключена к центральной ЭВМ, а суммарная длина коммуникаций была бы минимальна. Заданы расстояния d_{ij} между i-й и j-й ЭВМ.

Данная задача формализуется в виде задачи коммивояжера:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \to \min;$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = \overline{1,n};$$

$$\sum_{i=1}^{n} x_{ij} = 1, j = \overline{1,n};$$

$$U_i - U_j + (n-1)x_{ij} \le n-2, i = \overline{2,n}, j = \overline{2,n};$$

$$x_{ij} \in \{0,1\}, u_i \ge 0.$$

При этом $x_{ij}=1$, если i-я и j-я ЭВМ соединяются и $x_{ij}=0$ в противном случае.

Задача о покрытии

Пусть имеется n предметов, каждый из которых обладает некоторым числом признаков из заданного множества m признаков, а в совокупности эти n предметов обладают всеми m признаками. Необходимо выбрать минимальное число предметов, которые в совокупности обладали бы m признаками. Условия задачи задаются матрицей A с элементами

$$a_{ij} = \begin{cases} 1, \text{ если } j\text{-й предмет обладает } i\text{-м признаком} \\ 0 \quad \text{в противном случае} \end{cases}$$

Введем бинарные переменные:

$$x_j = \begin{cases} 1, \text{ если } j\text{-й предмет выбран} \\ 0 \quad \text{в противном случае} \end{cases}$$

Тогда математическая модель задачи имеет следующий вид:

$$\sum_{j=1}^{n} x_j \to \min;$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1, i = \overline{1, m};$$

$$x_j \in \{0,1\}, \quad j = \overline{1, n}$$

Каждое i-е ограничение в данном случае показывает, что должен быть выбран хотя бы один предмет, обладающий i-м признаком.

Если каждому j-му предмету приписывается вес c_j, может быть сформулирована *взвешенная задача о покрытии:*

$$\sum_{j=1}^{n} c_j x_j \to \min$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1, \ i = \overline{1, m},$$

$$x_j \in \{0,1\}, \quad j = \overline{1, n}$$

Если требуется найти минимальное число таких предметов, что i-м признаком обладает не менее λ_i предметов из указанного набора, получаем *задачу о кратном покрытии:*

$$\sum_{j=1}^{n} x_j \to \min$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq \lambda_i, \quad i = \overline{1,m},$$

$$x_j \in \{0,1\}, \quad j = \overline{1,n}$$

Пример 8.6. Пусть некоторое количество информации хранится в n массивах (файлах) длины $c_j, j = \overline{1,n}$, причем на каждую единицу информации отводится по крайней мере один массив. Задана матрица T с элементами

$$t_{ij} = \begin{cases} 1, \text{если в } j\text{-м массиве находится } i\text{-я информация} \\ 0 \quad \text{в противном случае} \end{cases}$$

Получена заявка на m типов единиц информации. Необходимо определить подмножество массивов минимальной длины, необходимых для удовлетворения заявки.

Введем бинарные переменные:

$$x_j = \begin{cases} 1, \text{если } j\text{-й массив выбирается} \\ 0 \quad \text{в противном случае} \end{cases}$$

Модель задачи формализуется в виде задачи о покрытии:

$$\sum_{j=1}^{n} c_{ij} x_j \to \min;$$

$$\sum_{j=1}^{n} t_{ij} x_j \geq 1, \quad i = \overline{1,m};$$

$$x_j \in \{1,0\}, \quad j = \overline{1,n}$$

Пример 8.7. Имеется m неисправностей и n диагностических тестов для их проверки. Задана матрица A с элементами

$$t_{ij} = \begin{cases} 1, \text{ если j-й тест диагностирует i-ю. неисправность} \\ 0 \quad \text{в} \quad \text{противном случае} \end{cases}$$

Необходимо выбрать минимальное число диагностических тестов для проверки всех неисправностей.

Введем бинарные переменные:

$$x_j = \begin{cases} 1, \text{если j - й тест выбирается} \\ 0 \quad \text{в} \quad \text{противном} \quad \text{случае} \end{cases}$$

Модель задачи формализуется в виде задачи о покрытии:

$$\sum_{j=1}^{n} x_j \to \min;$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1, i = \overline{1, m};$$

$$x_j \in \{0,1\}, \quad j = \overline{1, n}$$

Задача о разбиении

Задача о разбиении аналогична задачи о покрытии с тем отличием, что признаки у различных предметов не должны совпадать:

$$\sum_{j=1}^{n} x_j \to \min;$$

$$\sum_{j=1}^{n} a_{ij} x_j = 1, i = \overline{1, m};$$

$$x_j \in \{1,0\}, \quad j = \overline{1, n}$$

Задача о взвешенном разбиении формулируется в виде:

$$\sum_{j=1}^{n} c_j x_j \to \min;$$

$$\sum_{j=1}^{n} a_{ij} x_j = 1, i = \overline{1, m};$$

$$x_j \in \{1,0\}, \quad j = \overline{1, n}$$

ЗАКЛЮЧЕНИЕ

В учебном пособии рассмотрены основные классы методов, которые используются при поиске оптимальных проектных и управленческих решений в автоматизированных системах. Рассмотренные методы инвариантны к предметной области приложения и могут использоваться для решения широкого круга прикладных оптимизационных задач.

Алгоритмы, рассмотренные в пособии, могут быть доведены до программной реализации. Это позволяет использовать учебное пособие не только в лекционных курсах, но и в курсовом и дипломном проектировании.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Акулич И.Л Математическое программирование в примерах и задачах / И.Л. Акулич. – СПб.: Лань, 2011. – 231 с.

2. Батищев Д.И. Оптимизация в САПР: учебник / Д.И. Батищев, Я.Е Львович, В.Н. Фролов. – Воронеж: Изд-во Воронежского государственного университета, 1997. – 416 с.

3. Батищев Д.И. Решение дискретных задач с помощью эволюционно-генетических алгоритмов: учеб. пособие / Д.И. Батищев, В.Е. Костюков, Е.А. Неймарк. – Нижний Новгород: Изд-во ННГУ им. Н.И. Лобачевского, 2011. - 199 с.

4. Белецкая С.Ю. Математические методы поиска оптимальных решений: учеб. пособие / С.Ю. Белецкая. – Воронеж: ВГТУ, 2008. – 201 с.

5. Курейчик В.В. Основы теории эволюционных вычислений / В.В. Курейчик, В.М. Курейчик, С.И. Родзин.– Ростов на Дону: Изд-во ЮФУ, 2010. – 156 с.

6. Пантелеев А.В. Методы оптимизации в примерах и задачах: учеб. пособие / А.В. Пантелеев, Т.А. Летова. – М.: Высш. шк., 2002. – 544 с.

7. Реклейтис Г. Оптимизация в технике / Г. Реклейтис, А. Рейвиндран, К. Рэгсдел: в 2 кн.: пер. с англ. – М.: Мир, 1986. – 346 с. 320 с.

8. Черноруцкий И.Г. Методы оптимизации: Компьютерные технологии / И.Г. Черноруцкий. – СПб.: ИРМ, 2011. – 384 с.

ОГЛАВЛЕНИЕ

Учебное издание

Белецкая Светлана Юрьевна

МЕТОДЫ ОПТИМИЗАЦИИ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

В авторской редакции

Подписано в печать 15.12.2017.
Формат 60х84/16. Бумага для множительных аппаратов.
Усл. печ. л. 9,7. Уч.-изд. л. . Тираж экз.
Зак. № .

ФГБОУ ВО «Воронежский государственный технический университет»
394026 Воронеж, Московский просп., 14

Участок оперативной полиграфии издательства ВГТУ
394026 Воронеж, Московский просп., 14