Вероятностные методы локального поиска для задач дискретной оптимизации *

Ю. А. Кочетов

Аннотация

Приводится обзор современных методов локального поиска для решения NP-трудных задач дискретной оптимизации. Обсуждаются общие схемы алгоритмов поиска с запретами, имитации отжига и генетических алгоритмов. Показано, что эти разные по своей структуре алгоритмы используют общую математическую конструкцию конечных цепей Маркова. Это свойство гарантирует сходимость по вероятности наилучшего найденного решения к оптимальному решению задачи. Приводятся примеры удачного применения данных алгоритмов при поиске дискретных структур, существование которых казалось проблематичным.

Введение

Идеи локального поиска при решении задач дискретной оптимизации являются, по-видимому, наиболее естественными и наглядными. Первые шаги их реализации относятся к концу 50-х, началу 60-х годов двадцатого столетия [22, 30, 80] и связаны в основном с задачей коммивояжера. Позднее эти идеи использовались и для задач размещения, построения сетей, расписаний и др. [72, 77, 56, 10]. Однако довольно быстро выяснилось, что простой локальный спуск не позволяет находить глобальный оптимум задачи, и отсутствие концептуального прогресса ослабило интерес к данному направлению. В последние 10-15 лет наблюдается возрождение этого подхода. Оно связано как с новыми алгоритмическими схемами, построенными на аналогиях с живой

*Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 99-01-00510).

и неживой природой, так и новыми теоретическими результатами в области локального поиска. Изменился общий взгляд на построение локальных алгоритмов. Требование монотонного улучшения по целевой функции больше не является доминирующим. Наиболее мощные алгоритмы допускают произвольное ухудшение и многие из них могут рассматриваться как способ порождения конечных неразложимых цепей Маркова на подходящем множестве состояний.

В данной работе приводится краткий обзор результатов в этой бурно развивающейся области дискретной оптимизации. В § 1 вводятся основные определения и дается краткое введение в теорию сложности задач локального поиска. В § 2 на примере задачи коммивояжера демонстрируются разные способы построения окрестностей и их свойства. Три наиболее продуктивные схемы локального поиска приводятся в § 3 и, наконец, в § 4 на примерах из теории графов и кодирования иллюстрируются возможности локального поиска по обнаружению дискретных структур, существование которых представляется проблематичным.

1 Основные определения

Пусть пара $(\mathcal{I}, f)$ задает индивидуальную задачу дискретной оптимизации с конечным множеством допустимых решений $\mathcal{I}$ и целевой функцией $f : \mathcal{I} \to R$. Для определенности будем считать, что требуется найти минимум фунции f на множестве $\mathcal{I}$ и само решение $i^* \in \mathcal{I}$, на котором этот минимум достигается. Решение i^* будем называть глобальным минимумом и множество глобальных минимумов будем обозначать через $\mathcal{I}^*$. Для каждого $i \in \mathcal{I}$ определим функцию окрестности $N : \mathcal{I} \to 2^{\mathcal{I}}$, которая для каждого допустимого решения задает множество *соседних* решений, в некотором смысле близких к данному. Множество $N(i)$ будем называть окрестностью решения i в множестве $\mathcal{I}$. Функция окрестности может быть достаточно сложной и отношение соседства не всегда симметрично, то есть i может быть соседом j, но j может не быть соседом i (см., например, [63, 85]).

Определение 1. Решение $i \in \mathcal{I}$ называется локальным минимумом по отношению к функции N, если

$$f(i) \leq f(j) \text{ для всех } j \in N(i).$$

Множество локальных минимумов обозначим через $\widehat{\mathcal{I}}$. Это множество, очевидно, зависит от выбора функции N.

Определение 2. Функцию окрестности $N : \mathcal{I} \to 2^{\mathcal{I}}$ будем называть точной, если $\widehat{\mathcal{I}} \subseteq \mathcal{I}^*$.

Стандартный алгоритм локального спуска начинает свою работу с некоторого начального решения i_0, выбранного случайно или с помощью какого-либо вспомогательного алгоритма. На каждом шаге локального спуска происходит переход от текущего решения к соседнему решению с меньшем значением целевой функции до тех пор, пока не будет достигнут локальный оптимум.

Алгоритмы локального поиска широко применяются для решения NP-трудных задач дискретной оптимизации. Вместе с тем, многие полиномиально разрешимые задачи могут рассматриваться как задачи, легко решаемые локальным спуском. При подходящем выборе полиномиальной окрестности соответствующая теорема может быть сформулированна в следующем виде: допустимое решение не является глобальным оптимумом, если и только если оно может быть улучшено некоторым локальным образом. Ниже приводится несколько примеров таких задач, подробное описание которых можно найти в [9]. Данные примеры указывают на важность локального поиска при построении оптимизационных алгоритмов и достаточно общий характер этого подхода.

1. <u>Линейное программирование</u>. Геометрически алгоритм симплекс метода можно интерпретировать как движение по вершинам многогранника допустимой области. Вершина не является оптимальной, если и только если существует смежная с ней вершина с меньшим значением целевой функции. Алгебраически, предполагая невырожденность задачи, базисное допустимое решение не является оптимальным, если и только если оно может быть улучшено локальным изменением базиса, т. е. заменой одной базисной переменной на небазисную. Получающаяся таким образом окрестность является точной и имеет полиномиальную мощность.

2. <u>Минимальное остовное дерево</u>. Остовное дерево не является оптимальным, если и только если локальной перестройкой, добавляя одно ребро и удаляя из образовавшегося цикла другое ребро, можно получить новое остовное дерево с меньшим суммарным весом. Операция локальной перестройки задает отношение соседства на множестве остовных деревьев. Окрестность любого дерева имеет полиномиальную мощность, а функция окрестности яв-

ляется точной.

3. Максимальное паросочетание . Паросочетание не является максимальным, если и только если существует увеличивающий путь. Два паросочетания называют соседними, если их симметрическая разность образует путь. Определенная таким образом окрестность является точной и имеет полиномиальную мощность. Аналогичные утверждения справедливы для взвешенных паросочетаний, совершенных паросочетаний минимального веса, задач о максимальном потоке и потоке минимальной стоимости.

На каждом шаге локального спуска функция окрестности N задает множество возможных направлений движения. Очень часто это множество состоит из нескольких элементов и имеется определенная свобода в выборе следующего решения. Правило выбора может оказать существенное влияние на трудоемкость алгоритма и результат его работы. Например, в задаче о максимальном потоке алгоритм Форда-Фалкерсона (который тоже можно рассматривать как вариант локального спуска) имеет полиномиальную временную сложность при выборе кратчайшего пути для увеличения потока и экспоненциальную временную сложность без гарантии получить глобальный оптимум при произвольном выборе пути [11]. Таким образом, при разработке алгоритмов локального поиска важно не только правильно определить окрестность, но и верно задать правило выбора направления спуска. Интуитивно кажется, что в окрестности надо брать элемент с наименьшим значением целевой функции. Однако, как мы увидим ниже, разумным оказывается не только такой выбор, но и движение в "абсурдном"направлении, когда несколько шагов с ухудшением могут привести (и часто приводят) к лучшему локальному оптимуму.

При выборе окрестности хочется иметь множество $N(i)$ как можно меньшей мощности, чтобы сократить трудоемкость одного шага. С другой стороны, более широкая окрестность, вообще говоря, приводит к лучшему локальному оптимуму. Поэтому при создании алгоритмов каждый раз приходится искать оптимальный баланс между этими противоречивыми факторами. Ясных принципов разрешения этого противоречия на сегодняшний день не известно, и для каждой задачи этот вопрос решается индивидуально.

Определение 3. Графом окрестностей $G_N = (\mathcal{I}, E)$ будем называть взвешенный ориентированный граф, вершинами которого являются допустимые решения задачи, а дугами — упорядоченные пары (i, j), если $j \in N(i)$. Веса в этом графе приписаны вершинам и равны соответствующим значениям целевой функции.

Граф окрестностей G_N (*neighborhood graph*) иногда называют еще ландшафтом целевой функции или просто ландшафтом (*landscape, fitness landscape*, см., например, [68, 82, 83]). При определении функции окрестности N важно следить за тем, чтобы получающийся граф G_N был *строго связен*, т. е. для каждой пары вершин i и j существовал путь из i в j. Это свойство является важным при анализе асимптотического поведения алгоритмов, например, вероятностных метаэвристик, о которых пойдет речь в разделе 3. Если же это свойство не выполняется, то стараются получить хотя бы свойство *вполне связности*, когда из любой вершины существует путь в вершину $i^* \in \mathcal{I}^*$ с минимальным значением целевой функции [24, 63]. Если же и этого свойства нет, то мы теряем увереность в достижении глобального оптимума локальными методами и должны либо ограничиться локальными оптимумами, либо переопределить функцию окрестности.

Анализ вычислительной сложности локального поиска в последние годы интенсивно ведется в двух направлениях: эмпирическом и теоретическом. Как ни странно, но эти направления дают существенно разные оценки возможностям локального поиска.

Эмпирические результаты. Для многих NP-трудных задач локальный поиск позволяет находить приближенные решения, близкие по целевой функции к глобальному оптимуму. Трудоемкость алгоритмов часто оказывается полиномиальной, причем степень полинома достаточно мала. Так для задачи о разбиении множества вершин графа на две равные части разработаны алгоритмы локального поиска со средней трудоемкостью $O(n \log n)$, n — число вершин, которые дают всего несколько процентов погрешности [53].

Для задачи коммивояжера алгоритмы локального поиска являются наилучшими с практической точки зрения. Один из таких алгоритмов с окрестностью Лина-Кернигхана в среднем имеет погрешность около 2% и максимальная размерность решаемых задач достигает 1 000 000 городов [55, 33]. На случайно сгенерированных задачах такой колоссальной размерности итерационная процедура Джонсона позволяет находить решения с отклонением около $0,5\%$ за несколько минут на современных компьютерах. Аналогичные результаты для реальных задач размерностью до 100 000 получены в [86].

Для задач теории расписаний, размещения, покрытия, раскраски и многих других NP-трудных задач алгоритмы локального поиска показывают превосходные результаты. Более того, их гибкость при изменении математической модели, простота реализации и наглядность превращают локальный поиск в мощное средство для решения практических задач.

Теоретические результаты. Исследование локального поиска с точки зрения гарантированных оценок качества показывают границы его возможностей. Построены трудные для локального поиска примеры, из которых следует, что [84]

1) минимальная точная окрестность может иметь экспоненциальную мощность;

2) число шагов для достижения локального оптимума может оказаться экспоненциальным;

3) значение локального оптимума может сколь угодно сильно отличаться от глобального оптимума.

Эти результаты подталкивают к более пристальному рассмотрению задачи нахождения произвольного локального оптимума, ее трудоемкости в худшем и среднем случае. Абстрактная (массовая) задача локального поиска L задается множеством ее индивидуальных примеров, каждый из которых однозначно определяет целевую функцию, функцию окрестности и множество допустимых решений.

Определение 4 [89]. Задача L принадлежит к классу PLS (polynomial-time local search), если за полиномиальное время от длины записи исходных данных можно выполнить следующие три операции:

1. Для произвольного примера x проверить принадлежность x к L, и если $x \in L$, то найти допустимое решение s_0 задачи x.

2. Для любого решения s проверить его допустимость и вычислить для него значение целевой функции.

3. Узнать, является ли данное допустимое решение s локальным оптимумом для x, и если нет, то найти в окрестности решение с лучшим значением целевой функции.

Другими словами, в классе PLS содержатся все задачи локального поиска, для которых проверка локальной оптимальности может быть осуществлена за полиномиальное время. По аналогии с классами P и NP для задач локального поиска можно ввести в рассмотрение классы P_s и NP_s. Неформально, класс P_s (NP_s) состоит из задач нахождения локального оптимума, разрешимых за полиномиальное время на детерминированной (недетерминированной) машине Тьюринга. Заметим, что классы P_s и NP_s вводятся без перехода к задачам распознавания, так как, имея алгоритм для нахождения локального оптимума неясно, как отвечать на вопрос: *Существует ли локальный оптимум со значением целевой функции, не превосходящим A?* Одним из представителей класса P_s является, например, задача линейного программирования, так как метод эллипсоидов имеет полиномиальную трудоемкость. Другим примером может служить задача о максимальной клике, где число локальных улучшений, очевидно, не привышает числа вершин графа. Для классов P_s и NP_s установлены, в частности, следующие свойства [89]:

Теорема 1. *$P_s = NP_s$, если и только если $P = NP$.*

Теорема 2. $P_s \subseteq PLS \subseteq NP_s$.

Теорема 3. *Если задача L из класса PLS является NP-трудной, то NP=co-NP.*

Для задач поиска локального оптимума естественным образом определяется понятие PLS-сведения.

Определение 5. Пусть L_1 и L_2 — две задачи поиска локального оптимума. PLS-сведение задачи L_1 к задаче L_2 состоит в построении двух полиномиально вычислимых функций h и g таких, что

1. h переводит индивидуальную задачу $x \in L_1$ в индивидуальную задачу $h(x) \in L_2$;

2. g переводит пару (решение задачи $h(x), x$) в решение задачи x;

3. для всех $x \in L_1$, если s — локальный оптимум для $h(x) \in L_2$, то $g(s, x)$ — локальный оптимум для задачи x.

Если такие функции h и g удается построить, то говорят, что задача L_1 PLS-сводится к задаче L_2. Нетрудно заметить, что так определенное понятие PLS-сводимости обладает свойством транзитивности и, если $L_2 \in P_s$, а L_1 PLS-сводится к L_2, то $L_1 \in P_s$. Говорят, что задача $L \in PLS$ является PLS-полной, если любая задача в PLS может быть PLS-сведена к ней. Проще говоря, PLS-полные задачи являются наиболее трудными в этом классе и если хотя бы одна из них может быть решена за полиномиальное время, то все остальные задачи могут быть решены за полиномиальное время.

2 Окрестности

Выбор окрестности, как уже отмечалось выше, играет важную роль при построении алгоритмов локального поиска. От него существенно зависит трудоемкость одного шага алгоритма, общее число шагов и, в конечном счете, качество получаемого локального оптимума. На сегодняшний день нет и, возможно никогда не будет, единого правила выбора окрестности. Для каждой задачи функцию N приходится определять заново, учитывая специфику данной задачи. Более того, по-видимому для каждой задачи можно предложить несколько функций окрестности с разными по мощности множествами $N(i)$ и, как следствие, разными множествами локальных оптимумов. Ниже будут приведены три примера выбора окрестностей для задачи коммивояжера, которые иллюстрируют возможные пути построения окрестностей и их свойства.

2.1 Окрестности на перестановках

Напомним, что задача коммивояжера состоит в нахождении минимального по длине гамильтонова цикла в полном взвешенном ориентированном (или неориентированном) графе с n вершинами. Для удобства ниже будет рассматриваться только неориентированные графы с симметричной целочисленной неотрицательной матрицей расстояний $w_{ij} = w_{ji}, 1 \le i,j \le n$. Каждое допустимое решение задачи коммивояжера или тур в графах будем представлять в виде перестановки $\pi = \{i_1, \ldots, i_n\}$. Для $\pi \in \mathcal{I}$ определим значение функции окрестности $N(\pi)$ как множество всех перестановок, отличающихся от π только в двух позициях (*city-swap*). Множество $N(\pi)$ содержит ровно $n(n-1)/2$ элементов и вычислительная сложность одного шага локального поиска с учетом вычисления целевой функции не превосходит $O(n^2)$ операций.

Обозначим через $W(\pi)$ длину гамильтонова цикла и определим разностный оператор ∇^2 для $W(\pi)$ следующим образом [39]:

$$\nabla^2 W(\pi) = \frac{1}{|N(\pi)|} \sum_{\pi' \in N(\pi)} W(\pi') - W(\pi),\ \pi \in \mathcal{I}.$$

Оператор $\nabla^2 W(\pi)$ задает среднее отклонение целевой функции $W(\pi)$ в окрестности данной перестановки π. Для локального оптимума $\pi \in \widehat{\mathcal{I}}$ справедливо неравенство $\nabla^2 W(\pi) \ge 0$. Пусть W_{av} — средняя длина тура на множестве всех допустимых решений $\mathcal{I}$. Тогда справедливы следующие утверждения [39]

Теорема 4. *Функция* $\widetilde{W} = W - W_{av}$ *удовлетворяет уравнению*

$$\nabla^2 \widetilde{W} = -\frac{4}{n}\widetilde{W}.$$

Следствие 1. *Любой локальный минимум* $\pi \in \widehat{\mathcal{I}}$ *имеет длину* $W(\pi) \leq W_{av}$.

Следствие 2. *Алгоритм локального поиска, начиная с произвольной перестановки, достигнет локального оптимума за* $O(nW)$ *шагов, если длина максимального тура превосходит среднее значение* W_{av} *не более чем в* 2^W *раз.*

Аналогичные утверждения справедливы и для следующих задач:

(i) о разбиении $2n$-вершиного графа на две части по n вершин с минимальным суммарным весом ребер, соединяющих эти части;

(ii) о раскраске вершин графа в n цветов так, чтобы смежные вершины имели разные цвета;

(iii) о разбиении n-элементного множества на два подмножества так, чтобы суммарный вес одного подмножества совпал с суммарным весом другого подмножества;

(iv) о 3-выполнимости в следующей постановке: для n булевых переменных задан набор троек; каждая переменная может входить в набор с отрицанием или без него; набор считается выполненным, если он содержит разные значения, т. е. хотя бы одно *истинное* и хотя бы одно *ложное*; требуется узнать, существует ли назначение переменных, при котором все наборы будут выполнены.

2.2 Окрестности Лина-Кернигхана

Рассмотрим гамильтонов цикл как последовательность ребер. Окрестностью 2-*opt* будем называть множество всех гамильтоновых циклов, получающихся из заданного заменой двух несмежных ребер. Такая окрестность содержит $n(n-3)/2$ элементов, что несколько меньше, чем в окрестности *city-swap*. На рисунке 1 показано различие между окрестностями 2-*opt* и *city-swap*. Для окрестности 2-*opt* удаление ребер (a, b) и (e, f) приводит к появлению двух новых ребер (a, e) и (b, f) и обратному обходу дуг от d до c. Если в окрестности *city-swap* в перестановке поменять местами b и e, то появятся четыре новых ребра $(a, e), (e, c), (d, b)$ и (b, f), но обход дуг от c до d останется прежним.

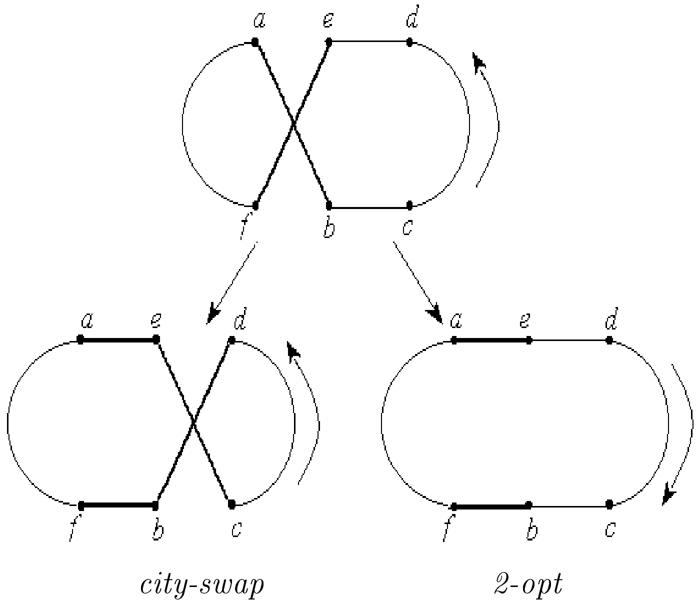

Рис. 1. Различия между окрестностями

В случае, когда точки расположены на плоскости, а веса w_{ij} являются евклидовыми расстояниями, пересечение ребер (a, b) и (e, f) свидетельствует о возможности улучшения данного тура в силу неравенства труугольника. Однако это условие не является необходимым и улучшение возможно даже без пересечения ребер.

Окрестность 2-*opt* может быть естественным образом расширена до 3-*opt* и в общем случае до k-*opt* окрестности, где разрешается заменять не более k ребер данного тура на k новых ребер [65].

Теорема 5 [61]. *Поиск локального оптимума в задаче коммивояжера с окрестностью k-opt является PLS-полной для некоторого k.*

Минимальное значение константы k, при котором утверждение остается верным, пока неизвестно. Из работы [61] следует, что утверждение справедливо при $k = 8$ и возможно это значение может быть уменьшено до $k = 6$. На практике широко применяются окрестности с $k = 2$, но теоретических результатов о сложности таких алгоритмов пока не получено.

На основе окрестности 2-opt Лином и Кернигханом предложена очень эффективная эвристика [66]. Она позволяет заменять произвольное число ребер и переходит от одного тура к другому, используя принципы жадных алгоритмов. Основная идея эвристики заключается в следующем. Удалим из гамильтонового цикла произвольное ребро, скажем (a, b). В полученном пути один конец (вершину a) будем считать фиксированной, а другой конец будем менять, перестраивая гамильтонов путь. Добавим ребро из вершины b, например (b, c), и разорвем образовавшийся единственный цикл так, чтобы снова получить гамильтонов путь. Для этого придется удалить ребро, инцидентное вершине c. Обозначим его (c, d). Новый гамильтонов путь имеет концевые вершины a и d (см. рис. 2). Эту процедуру будем называть ротацией. Для получения нового гамильтонова цикла достаточно добавить ребро (a, d). Согласно алгоритму Лина-Кернигхана, переход от одного тура к другому состоит из удаления некоторого ребра, выполнении серии последовательных ротаций и, наконец, замыкания концевых вершин полученного гамильтонова пути. Существуют различные варианты этой основной схемы, которые отличаются правилами выбора ротаций и ограничениями на множества удаляемых и добавляемых ребер [54]. В алгоритме Лина-Кернигхана ротации выбираются так, чтобы минимизировать разность $w_{bc} - w_{cd}$. При этом множества удаляемых и добавляемых ребер в серии ротаций не должны пересекаться. Последнее ограничение гарантирует, что число ротаций не превысит n^2 и трудоемкость одного шага (перехода от одного тура к другому) останется полиномиальной. Общее число шагов алгоритма, по-видимому, не может быть ограничено полиномом и известен вариант окрестности Лина-Кернигхана, для которого задача коммивояжера становится PLS-полной [89].

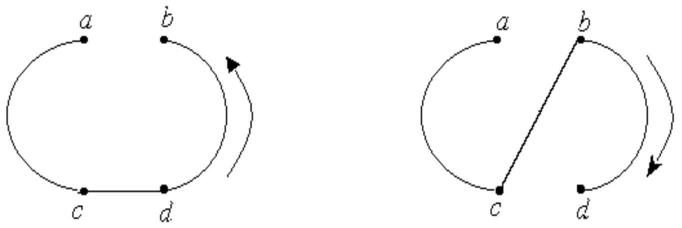

Рис. 2. Процедура ротации

2.3 Экспоненциальные окрестности

Одним из новых перспективных направлений в области локального поиска является исследование свойств окрестностей экспоненциальной мощности. Если лучшее решение в такой окрестности можно найти за полиномиальное время, то алгоритм локального поиска с такой окрестностью получает определенное приемущество перед остальными. Для задачи коммивояжера первые примеры таких окрестностей были предложены в работах [25, 19]. Мы не будем останавливаться на всех результатах в данной области. Изложим, в качестве иллюстрации, идею одного из таких подходов. Этот пример наглядно показывает тесную связь между экспоненциальными окрестностями и полиномиально разрешимыми случаями задачи коммивояжера. Наличие такой связи, по-видимому, будет стимулировать поиск новых полиномиально разрешимых случаев для трудных комбинаторных задач, что в свою очередь может привести к дальнейшему прогрессу в области локального поиска.

Основная идея алгоритмов из [78] может быть представлена следующим образом. Предположим, что множество из n городов разбито на две части $x_1, \dots, x_m$ и $y_1, \dots, y_k$ так, что $k + m = n$ и $m \le k$. Для удобства и простоты изложения добавим в первую группу $k - m$ фиктивных городов $x_{m+1}, \dots, x_k$ и рассмотрим окрестность $N(y_1, \dots, y_k, x_1, \dots, x_m)$, состоящую из всех туров вида

$$y_1 x_{\tau_1} y_2 x_{\tau_2}, \dots, y_k x_{\tau_k} y_1,$$

где $(\tau_1, \dots, \tau_k)$ — произвольная перестановка элементов $\{1, \dots, k\}$. Содержательно, каждый такой тур получается вставкой городов x_i между парами городов y_j, y_{j+1}. Множество $N(y_1, \dots, y_k, x_1, \dots, x_m)$ имеет экспоненциальную мощность $[\frac{n+1}{2}]!$ при $2m \le n \le 2m + 1$. Оптимальный тур в этом множестве может быть найден с помощью решения задачи о назначениях следующим образом. При $j = 1, \dots, k$ положим

$$d_{ij} = \begin{cases} w(y_j x_i) + w(x_i y_{j+1}), & i = 1, \dots, m, \\ w(y_j y_{j+1}), & i = m+1, \dots, k, \end{cases}$$

введем переменные

$$z_{ij} = \begin{cases} 1, & \text{если } x_i \text{ располагается между } y_j, y_{j+1}, \\ 0 & \text{в противном случае} \end{cases}$$

и рассмотрим задачу

$$\min \sum_{i=1}^{k} \sum_{j=1}^{k} d_{ij} z_{ij}$$

$$\sum_{i=1}^{k} z_{ij} = 1, \; j = 1, \dots, k,$$

$$\sum_{j=1}^{k} z_{ij} = 1, \; i = 1, \dots, k,$$

$$z_{ij} \in \{0, 1\}, \; i, j = 1, \dots, k.$$

Оптимальное решение задачи определяет наилучшее размещение городов $x_i, i = 1, \dots, m$, между городами $y_j, j = 1, \dots, k$,, то есть оптимальный тур в множестве $N(y_1, \dots, y_k, x_1, \dots, x_m)$. Решение этой задачи может быть найдено с временной сложностью $O(k^3)$, например, алгоритмом Форда-Фалкерсона для максимизации потока в сети [11] или построением совершенного паросочетания минимального веса для двудольного графа с долями $X = \{x_1, \dots, x_k\}$ и $Y = \{y_1, \dots, y_k\}$.

Заметим, что максимальная мощность окрестности $N(y_1, \dots, y_k, x_1, \dots, x_m)$ достигается при $m < n/2$. Точнее, пусть $m = n/2 - p, p \geq 0$, целое, $n \geq 5$ и $\sigma(n) \equiv n \pmod 2$. Для действительного числа r обозначим через $[r]_0$ (соответственно $[r]_1$) максимальное целое (полуцелое, то есть число вида $q/2$ для нечетных q), не превосходящее r. Тогда максимальная мощность $N_{max}(n)$ окрестности превосходит $[\frac{n+1}{2}]!$ и равна [40]

$$N_{max}(n) = (n/2 + p_0)!/2p_0!,$$

где $p_0 = [\sqrt{\frac{1}{8}n + \frac{9}{8}} + \frac{3}{8}]_{\sigma(n)}$. Применение этой формулы позволяет получить асимптотическое выражение для мощности окрестности.

Теорема 6 [40]. $N_{max}(n) = \Theta(\frac{e^{\sqrt{n/2}}[\frac{n+1}{2}]!}{n^{1/4+[1/2]_{\sigma(n)}}})$.

Перейдем теперь к наиболее интригующему свойству данной окрестности, связанному к расстоянием между турами в графе окрестностей. Оказывается, что с помощью множества $N(y_1, \ldots, y_k, x_1, \ldots, x_m)$ можно так определить новую окрестность, что расстояние между любыми двумя турами не будет превосходить 4. Сам факт, что максимальное расстояние между турами не зависит от размерности задачи, представляется удивительным. Более того, это константа очень мала; всего 4 шага достаточно сделать, чтобы добраться из заданного решения до любого другого и, в частности, до оптимального. К сожалению, мы не знаем в какую именно сторону нужно сделать эти четыре шага, иначе задача стала бы полиномиально разрешимой. Кроме того, мы не знаем, как много локальных оптимумов типа 2-*opt* или *city-swap* содержится в такой экспоненциальной окрестности. Тем не менее, сам факт ограниченности диаметра свидетельствует о больших потенциальных возможностях такой окрестности.

Пусть T — произвольный гамильтонов цикл в полном взвешенном неориентированном n-вершином графе. Обозначим через $\mathcal{F}_{nm}$ семейство всех подмножеств мощности m, состоящих из несмежных вершин тура T. Каждый элемент семейства $(x_1, \ldots, x_m) \in \mathcal{F}_{nm}$ однозначно определяет множество туров $N(y_1, \ldots, y_{n-m}, x_1, \ldots, x_m)$. Объединение таких множеств обозначим через $N_m(T)$. Можно показать, что при фиксированном m мощность семейства $\mathcal{F}_{nm}$ равна [ ]

$$|\mathcal{F}_{nm}| = \begin{pmatrix} n-m \\ m \end{pmatrix} + \begin{pmatrix} n-m-1 \\ m-1 \end{pmatrix},$$

то есть выражается полиномом от n и, следовательно, оптимальный тур в множестве $N_m(T)$ можно найти за полиномиальное время. Будем говорить, что тур R является соседним к туру T, если в туре T найдутся m несмежных вершин $(x_1, \ldots, x_m)$ таких, что $R \in N(y_1, \ldots, y_{n-m}, x_1, \ldots, x_m)$.

Теорема 7 [41]. *При $m = \lfloor (n-1)/2 \rfloor$ для любых туров T_1 и T_5 существуют такие туры T_2, T_3 и T_4, что T_i является соседним к $T_{i-1}, i = 2, 3, 4, 5$.*

В общем случае при произвольных m длина пути в графе окрестностей G_N между двумя произвольными турами ограничена сверху величиной $4\lceil \lfloor (n-1)/2 \rfloor / m \rceil$.

3 Вероятностные эвристики

Идеи локального поиска получили свое дальнейшее развитие в так называемых *метаэвристиках* [73], то есть в общих схемах построения алгоритмов, которые могут быть применены практически к любой задаче дискретной оптимизации. Все метаэвристики являются итерационными процедурами и для многих из них установлена асимптотическая сходимость наилучшего найденного решения к глобальному оптимуму. К числу метаэвристик относятся алгоритмы имитации отжига, поиск с запретами, генетические алгоритмы, о которых пойдет речь в этом разделе, а также нейронные сети, муравьиные колонии, вероятностные жадные процедуры и другие. Обширную информацию по данной теме можно найти по адресам:
http://www.ing.unlp.edu.ar/cetad/mos/TSPBIB_home.html
http://www.informatik.hu-berlin.de/∼weinert/localsearch.html
http://iridia.ulb.ac.be/dorigo/ACO/ACO.html

3.1 Алгоритм имитации отжига

Экзотическое название данного алгоритма связано с методами имитационного моделирования в статистической физике [20], основанными на технике Монте-Карло. Исследование кристаллической решетки и поведения атомов при медленном остывании тела привело к появлению на свет вероятностных алгоритмов, которые оказались чрезвычайно эффективными в комбинаторной оптимизации. Впервые это было замечено в 1983 году [29, 58]. Сегодня этот алгоритм является популярным как среди практиков благодаря своей простоте, гибкости и эффективности, так и среди теоретиков, поскольку для данного алгоритма удается аналитически исследовать его свойства и доказать асимптотическую сходимость.

Алгоритм имитации отжига отностится к классу пороговых алгоритмов локального поиска. На каждом шаге этого алгоритма в окрестности текущего решения i_k выбирается некоторое решение j, и если разность по целевой функции между новым и текущим решением не превосходит заданного порога t_k, то новое решение j заменяет текущее. В противном случае выбирается новое соседнее решение. Общая схема пороговых алгоритмов может быть представлена следующим образом.

Пороговый алгоритм

1. Выбрать начальное решение $i_0 \in \mathcal{I}$ и положить $f^* = f(i_0), k = 0$.
2. Пока не выполнен критерий остановки делать следующее:
   2.1. Случайно выбрать $j \in N(i_k)$.
   2.2. Если $f(j) - f(i_k) < t_k$ то $i_{k+1} := j$.
   2.3. Если $f^* > f(i_k)$, то $f^* := f(i_k)$.
   2.4. Положить $k := k + 1$.

В зависимости от способа задания последовательности $\{t_k\}$ различают три типа алгоритмов

1. Последовательное улучшение: $t_k = 0,\ k = 0, 1, 2, \dots$ — вариант классического локального спуска с монотонным улучшением по целевой функции.

2. Пороговое улучшение: $t_k = c_k,\ k = 0, 1, 2, \dots,\ c_k \geq 0,\ c_k \geq c_{k+1}$ и $\lim_{k\to\infty} c_k \to 0$ — вариант локального поиска, когда допускается ухудшение по целевой функции до некоторого заданного порога, и этот порог последовательно снижается до нуля.

3. Имитация отжига: $t_k \geq 0,\ k = 0, 1, 2, \dots,$ — случайная величина с математическим ожиданием $\mathbb{E}(t_k) = c_k \geq 0$ — вариант локального поиска, когда допускается произвольное ухудшение по целевой функции, но вероятность такого перехода обратно пропорциональна величине ухудшения, точнее для любого $j \in N(i)$

$$P_{ij} = \begin{cases} 1, & \text{если } f(j) \leq f(i), \\ \exp(\frac{f(i)-f(j)}{c_k}), & \text{если } f(j) > f(i). \end{cases}$$

Последовательность $\{c_k\}$ играет важную роль при анализе сходимости и выбирается так, чтобы $c_k \to 0$ при $k \to \infty$. Иногда параметр c_k называют температурой [15], имея ввиду указанные выше истоки.

Для анализа данного и всех последующих алгоритмов нам потребуются некоторые определения из теории конечных цепей Маркова [8, 15].

Определение 6. Пусть $\mathfrak{O}$ обозначает множество возможных исходов некоторого случайного процесса. Цепь Маркова есть последовательность испытаний, когда вероятность исхода в каждом испытании зависит только от результата предшествующего испытания. Пусть $x(k)$ — случайная переменная, обозначающая результат k-го испытания. Тогда для каждой пары $i, j \in \mathfrak{O}$ вероятность перехода от i к j при k-ом испытании задается выражением

$$P_{ij}(k) = \mathbb{P}\{x(k) = j \mid x(k-1) = i\}.$$

Матрица $\{P_{ij}\}$ называется *переходной матрицей*. Цепь Маркова называется *конечной*, если множество исходов $\mathfrak{O}$ конечно, и *однородной*, если переходные вероятности не зависят от номера шага k.

Для алгоритма имитации отжига испытание состоит в выборе очередного решения на k-ом шаге. Вероятность перехода от i к j зависит от i, так как переход возможен только в соседнее решение $j \in N(i)$. Она не зависит от того, каким путем добрались до i. Таким образом, последовательность решений $\{i_k\}$ образует цепь Маркова. Эта цепь конечна, так как множество допустимых решений $\mathcal{I}$ конечно. Для каждой пары $i, j \in \mathcal{I}$ вероятность перехода от i к j задается равенством

$$P_{ij}(k) = \begin{cases} D_{ij}A_{ij}(c_k), & \text{если } i \neq j; \\ \\ 1 - \sum\limits_{l \neq i} D_{il}A_{il}(c_k), & \text{если } i = j, \end{cases}$$

где D_{ij} есть вероятность выбора решения j из окрестности $N(i)$ и $A_{ij}(c_k)$ — вероятность принятия решения j в качестве очередного текущего решения. Если решения из окрестности $N(i)$ выбираются равновероятно, то

$$D_{ij} = \begin{cases} 1/|N(i)|, & \text{если } j \in N(i); \\ 0 & \text{в противном случае }, \end{cases}$$

а вероятность принятия j определяется как

$$A_{ij}(c_k) = \exp\frac{(f(j) - f(ij))^+}{c_k},$$

где $a^+ = \max\{0, a\}$. Если $c_k = c$, то цепь Маркова является однородной. В общем случае алгоритм имитации отжига порождает конечную неоднородную цепь Маркова.

При исследовании асимптотического поведения алгоритмов важную роль играет так называемое стационарное распределение и вектор q_i размерности $|\mathfrak{O}|$, компоненты которого определяются как

$$q_i = \lim_{k\to\infty} \mathbb{P}\{x(k) = i \mid x(0) = j\}, j \in \mathfrak{O},$$

если такой предел существует и не зависит от j. Величина q_i равна вероятности оказаться в состоянии i после достаточно большого числа шагов.

Теорема 8 [15]. *Пусть $c_k = c$ для всех k и для любой пары $i, j \in \mathcal{I}$ найдется число $p \geq 1$ и $l_0, l_1, \ldots, l_p \in \mathcal{I}$ такие, что $l_0 = i, l_p = j$ и $D_{l_k l_{k+1}} > 0, k = 0, 1, \ldots, p-1$. Тогда для цепи Маркова, порожденной алгоритмом имитации отжига, существует единственное стационарное распределение, компоненты которого задаются формулой*

$$q_i(c) = \frac{\exp(-f(i)/c)}{\sum\limits_{j \in \mathcal{I}} \exp(-f(j)/c)}, \;\; i \in \mathcal{I},$$

а

$$\lim_{c \downarrow 0} q_i(c) = \begin{cases} 1/|\mathcal{I}^*|, & \text{если } i \in \mathcal{I}^*, \\ 0 & \text{в противном случае .} \end{cases}$$

Последнее равенство по сути означает, что с ростом числа итераций вероятность оказаться в точке глобального оптимума $i^* \in \mathcal{I}^*$ стремится к 1, если значение порога c_k стремится к нулю, т. е.

$$\lim_{c \downarrow 0} \lim_{k \to \infty} \mathbb{P}_c\{x(k) \in \mathcal{I}^*\} = 1.$$

На практике, конечно, нет возможности выполнить бесконечное число итераций. Поэтому теорема 8 является только источником вдохновения при разработке реальных алгоритмов. Существует много эвристических способов выбора конечной последовательности $\{c_k\}$ с целью повышения вероятности обнаружить глобальный оптимум. В некоторых работах рассматриваются небольшие примеры, для которых удается полностью исследовать характеристики цепей Маркова [14, 44]. В большинстве работ следуют рекомендациям первопроходцев [58] и используют схему геометрической прогрессии.

1. Начальное значение: $c_0 = \triangle f_{\max}$ — максимальная разность между двумя соседними решениями.

2. Понижение порога: $c_{k+1} = \alpha c_k, \; k = 0, 1, \ldots, K-1$, где α — положительная константа, достаточно близкая к 1, например, $\alpha \in [0,8; 0,99]$.

3. Конечное значение: $c_K > 0$ определяется либо по числу сделанных изменений, либо как максимальное c_k, при котором алгоритм не меняет текущее

решение в течение заданного числа шагов. При каждом значении c_k алгоритм выполняет порядка $N(i)$ шагов, не меняя значение порога.

3.2 Поиск с запретами

Основоположником алгоритма поиска с запретами (Tabu search) является Ф. Гловер, который предложил принципиально новую схему локального поиска [34, 35]. Она позволяет алгоритму не останавливаться в точке локального оптимума, как это предписано в стандартном алгоритме локального спуска, а путешествовать от одного оптимума к другому в надежде найти среди них глобальный оптимум. Основным механизмом, позволяющим алгоритму выбираться из локального оптимума, является список запретов $Tabu(i_k)$. Он строится по предыстории поиска, то есть по нескольким последним точкам $i_k, i_{k-1}, \ldots, i_{k-l+1}$, и запрещает часть окрестности текущего решения $N(i_k)$. Точнее на каждом шаге алгоритма очередная точка i_{k+1} является оптимальным решением подзадачи

$$f(i_{k+1}) = \min\{f(j) \mid j \in N(i_k) \setminus Tabu_l(i_k)\}.$$

Множество $Tabu_l(i_k) \subseteq N(i_k)$ определяется по предшествующим решениям. Список запретов учитывает специфику задачи и, как правило, запрещает использование тех "фрагментов" решения (ребер графа, координат вектора, цвет вершины), которые менялись на последних l шагах алгоритма. Константа $l \geq 0$ определяет его память. При "короткой памяти"($l = 0$) получаем стандартный локальный спуск.

Существует много вариантов реализации основной идеи поиска с запретами [37]. Приведем один из них, для которого удается установить асимптотические свойства. Рассмотрим рандомизированную окрестность $N_p(i) \subseteq N(i)$, где каждый элемент окрестности $j \in N(i)$ включается в множество $N_p(i)$ с вероятностью p независимо от других элементов. С ненулевой вероятностью множество $N_p(i)$ может совпадать с $N(i)$, может оказаться пустым или содержать ровно один элемент. Общая схема алгоритма поиска с запретами может быть представлена следующим образом.

Алгоритм поиска с запретами

1. Выбрать начальное решение $i_0 \in \mathcal{I}$ и положить
   $f^* := f(i_0), Tabu_l(i_0) := \emptyset, k := 0.$
2. Пока не выполнен критерий остановки делать следующее.

2.1. Сформировать окрестность $N_p(i_k)$.
2.2. Если $N_p(i_k) = \emptyset$, то $i_{k+1} := i_k$, иначе найти i_{k+1} такой, что $f(i_{k+1}) = \min\{f(j) \mid j \in N_p(i_k) \setminus Tabu_l(i_k)\}$.
2.3. Если $f^* > f(i_{k+1})$, то $f^* := f(i_{k+1})$.
2.4. Положить $k := k + 1$ и обновить список запретов $Tabu_l(i_k)$.

Параметры p и l являются управляющими для данного алгоритма и выбор их значений зависит от размерности задачи и мощности окрестности. Примеры адаптации этой схемы к разным задачам комбинаторной оптимизации можно найти, например, в [36].

Пусть $l = 0$. Тогда $Tabu(l) = \emptyset$ и случайная последовательность $\{i_k\}$ является конечной однородной цепью Маркова. Можно показать, что в этом случае переходные вероятности $P_{ij}, i, j \in \mathcal{I}$, определяются равенством

$$P_{ij} = \begin{cases} 0, & \text{если } j \notin N(i); \\ p(1-p)^{s-1}, & \text{если } j \in N(i) \text{ и } f(j) = \min\limits_{\tau \in N(i)}{}^{s}(f(\tau)), \end{cases}$$

где $\min^s$ означает s-й минимальный элемент множества.

Теорема 9. *Если граф окрестностей G_N строго связен, то для цепи Маркова существует единственное стационарное распределение $q_i > 0$, $i = 1, 2, \dots |\mathcal{I}|$, и независимо от выбора начальной точки $i_0 \in \mathcal{I}$ вероятность не найти глобальный оптимум стремится к нулю со скоростью геометрической прогрессии, то есть найдутся $b \geq 1$ и $0 < c < 1$ такие, что*

$$\mathbb{P}\{\min_{\tau \leq k} f(i_\tau) > f(i^*)\} \leq bc^k.$$

Заметим, что для стационарного распределения не получено точной формулы, как это удалось сделать для алгоритма имитации отжига. Утверждается только существование и единственность такого распределения. При малых размерностях его можно получить численно из соотношений

$$q_i = \sum_{j \in \mathfrak{O}} q_j P_{ij}, \; i \in \mathfrak{O},$$

$$\sum_{i \in \mathfrak{O}} q_i = 1.$$

Аналитического выражения для решения такой системы, по-видимому, не существует. Тем не менее, исследование этого распределения и, в частности, его компонент, соответствующих глобальному оптимуму $i^* \in \mathcal{I}^*$, представляет несомненный интерес.

Рассмотрим общий случай $l \geq 0$. В этом случае последовательность $\{i_k\}$ уже не является цепью Маркова, так как выбор очередной точки зависит от предыстории. Однако, если рассмотреть множество кортежей $\langle i_k, i_{k-1}, \dots, i_{k-l+1} \rangle$ длины l, то случайная последовательность таких кортежей, порождаемая алгоритмом, уже будет конечной однородной цепью Маркова. Матрица переходных вероятностей имеет теперь значительно большую размерность, но ее элементы будут определяться по сути теми же формулами, что и раньше.

Список запретов $Tabu_l(i_k)$ оказывает существенное влияние на поведение алгоритма. В частности, если на некотором шаге $Tabu_l(i_k) \supseteq N(i_k)$, то процесс поиска прекращается, $i_k = i_{k+1} = \dots$. Таким образом, даже свойство строгой связности графа G_N не гарантирует желаемого асимптотического поведения. Тем не менее при определенных условиях на список запретов удается обеспечить сходимость по вероятности наилучшего найденного решения к глобальному оптимуму, и правильное использование запретов заметно сокращает время решения задачи [59].

3.3 Генетические алгоритмы

Идея генетических алгоритмов заимствована у живой природы и состоит в организации эволюционного процесса, конечной целью которого является получение оптимального решения в сложной комбинаторной задаче. Разработчик генетических алгоритмов выступает в данном случае как "создатель", который должен правильно установить законы эволюции, чтобы достичь желаемой цели как можно быстрее. Впервые эти нестандартные идеи были применены к решению оптимизационных задач в середине 70-х годов [10, 23]. Примерно через десять лет появились первые теоретические обоснования этого подхода [50, 79, 81]. На сегодняшний день генетические алгоритмы доказали свою конкурентноспособность при решении многих NP-трудных задач [7, 38] и особенно в практических приложениях, где математические модели имеют сложную структуру и применение стандартных методов типа ветвей и границ, динамического или линейного программирования крайне затруднено.

Общую схему генетических алгоритмов проще всего понять, рассматривая задачи безусловной оптимизации

$$\max\{f(i) \mid i \in B^n\}, \quad B^n = \{0,1\}^n.$$

Примерами служат задачи размещения, стандартизации, выполнимости и другие [2, 5, 69]. Стандартный генетический алгоритм начинает свою работу с формирования начальной *популяции* $I_0 = \{i_1, i_2, \dots, i_s\}$ — конечного набора допустимых решений задачи. Эти решения могут быть выбраны случайным образом или получены с помощью вероятностных жадных алгоритмов [1, 3, 4]. Как мы увидим ниже, выбор начальной популяции не имеет значения для сходимости процесса в асимптотике, однако формирование "хорошей" начальной популяции (например из множества локальных оптимумов) может заметно сократить время достижения глобального оптимума.

На каждом шаге эволюции с помощью вероятностного оператора *селекции* выбираются два решения, *родители* i_1, i_2. Оператор *скрещивания* по решениям i_1, i_2 строит новое решение i', которое затем подвергается небольшим случайным модификациям, которые принято называть *мутациями*. Затем решение добавляется в популяцию, а решение с наименьшим значением целевой функции удаляется из популяции. Общая схема такого алгоритма может быть записана следующим образом.

<u>Генетический алгоритм</u>

1. Выбрать начальную популяцию I_0 и положить
   $f^* = \max\{f(i) \mid i \in I_0\}, k := 0$.
2. Пока не выполнен критерий остановки делать следующее.
   2.1. Выбрать родителей i_1, i_2 из популяции I_k.
   2.2. Построить i' по i_1, i_2.
   2.3. Модифицировать i'.
   2.4. Если $f^* < f(i')$, то $f^* := f(i')$.
   2.5. Обновить популяцию и положить $k := k + 1$.

Остановимся подробнее на основных операторах этого алгоритма: селекции, скрещивании и мутации. Среди операторов селекции наиболее распространенными являются два вероятностных оператора *пропорциональной* и *турнирной* селекции. При пропорциональной селекции вероятность на k-м шаге выбрать решение i в качестве одного из родителей задается формулой

$$\mathbb{P}\{i - \text{выбрано}\} = \frac{f(i)}{\sum\limits_{j \in I_k} f(j)},\ i \in I_k,$$

в предположении, что $f(i) > 0$ для всех $i \in \mathcal{I}$. При турнирной селекции формируется случайное подмножество из элементов популяции и среди них выбирается один элемент с наибольшим значением целевой функции. Турнирная селекция имеет определенные приемущества перед пропорциональной, так как не теряет своей избирательности, когда в ходе эволюции все элементы популяции становятся примерно равными по значению целевой функции. Операторы селекции строятся таким образом, чтобы с ненулевой вероятностью любой элемент популяции мог бы быть выбран в качестве одного из родителей. Более того, допускается ситуация, когда оба родителя представлены одним и тем же элементом популяции.

Как только два решения выбраны, к ним применяется вероятностный оператор скрещивания (*crossover*). Существует много различных версий этого оператора [38], среди которых простейшим, по видимому, является однородный оператор. По решениям i_1, i_2 он строит решение i' присваивая каждой координате этого вектора с вероятностью 0,5 соответствующее значение одного из родителей. Если вектора i_1, i_2 совпадали скажем по первой координате, то вектор i' "унаследует" это значение. Геометрически, оператор скрещивания случайным образом выбирает в гиперкубе вершину i', которая принадлежит минимальной грани, содержащей вершины i_1, i_2. Можно сказать, что оператор скрещивания старается выбрать новое решение i' где-то между i_1, i_2 полагаясь на удачу. Более аккуратная процедура могла бы выглядеть таким образом. Новым решением i' является оптимальное решение исходной задачи на соответствующей грани гиперкуба. Конечно, если расстояние Хемминга между i_1, i_2 равно n, то задача оптимального скрещивания совпадает с исходной. Тем не менее даже приближенное решение этой задачи вместо случайного выбора заметно улучшает работу генетического алгоритма [16, 17, 18, 32]. По аналогии с однородным оператором скрещивания легко предложить и другие операторы, использующие не только два, но и произвольное число решений из популяции. Например, в [71] использовалось восемь родителей. Другие примеры можно найти в [31]. С адаптацией этой идеи к задаче коммивояжера можно познакомиться в [54].

Оператор мутации, применяемый к решению i' в п. 2.3. генетического алгоритма, с заданной вероятностью $p_m \in (0, 1)$ меняет значение каждой координаты на противоположное. Например, вероятность того, что $i' = (0,0,0,0,0)$ в ходе мутации перейдет в $j' = (1,1,1,0,0)$, равна $p_m \times p_m \times p_m \times (1-p_m) \times (1-p_m) > 0$. Таким образом, с ненулевой вероятностью решение i' может перейти в любое другое решение. Отметим, что модификация решения i' может состоять не только в случайной мутации, но и в частичной перестройке решения алгоритмами локального поиска. Применение локального спуска позволяет генетическому алгоритму сосредоточиться только на локальных оптимумах. Множество локальных оптимумов может оказаться экспоненциально большим и на первый взгляд кажется, что такой вариант алгоритма не будет иметь больших преимуществ. Однако экспериментальные исследования распределения локальных оптимумов свидетельствуют о высокой концентрации их в непосредственной близости от глобального оптимума [21, 57]. Это наблюдение известно как гипотеза о существовании "большой долины"для задач на минимум или "центрального горного массива"для задач на максимум.

Гипотеза 1. В среднем локальные оптимумы расположены гораздо ближе к глобальному оптимуму чем к случайно выбранной точке. Их распределение в области допустимых решений на является равномерным. Они концентрируются в районе глобального оптимума, занимая область небольшого диаметра.

Эта гипотеза отчасти объясняет работоспособность генетических алгоритмов. Если в популяции собираются локальные оптимумы, которые согласно гипотезе сконцентрированы в одном месте, и очередное решение i' выбирается где-то между двумя произвольными локальными оптимумами, то такой процесс имеет много шансов найти глобальный оптимум. Аналогичные рассуждения объясняют работоспособность и других локальных алгоритмов. В связи с этим проверка и теоретическое обоснование данной гипотезы представляет несомненный интерес.

Перейдем теперь к анализу генетического алгоритма с позиций цепей Маркова. На каждом шаге эволюции алгоритм имеет дело с популяцией — набором из s случайных решений. Множество состояний такого стохастического процесса $\mathcal{I}^s$ — все возможные популяции. Переход одной популяции к другой не зависит от того, каким образом была получена данная популяция, а зависит только от состава самой популяции. Другими словами, генетический алгоритм порождает цепь Маркова на конечном множестве состояний. Так как оператор мутации позволяет за один шаг достичь любое решение задачи и, в частности, оптимальное решение, то с ростом числа шагов вероятность найти глобальный оптимум стремится к единице. Если при изменении популяции наилучшее найденное решение всегда остается в популяции, то в пределе она будет состоять из одних оптимальных решений задачи. Таким образом, имеет место сходимость по вероятности наилучшего найденного решения к глобальному оптимуму.

4 Поиск дискретных структур

В этом разделе будут приведены примеры удачного применения методов локального поиска в теории графов и кодировании. Оба примера используют один и тот же прием сведе́ния задачи поиска объекта с заданными свойствами к задаче дискретной оптимизации. На содержательном уровне этот прием состоит в задании желаемых свойств в виде ограничений на равенства или неравенства, а целевая функция задачи дискретной оптимизации состоит в поиске решения с минимальным значением суммарной невязки. Если для такой задачи удалось найти решение с нулевой невязкой, то нужный объект найден. Таким образом, для вероятностных алгоритмов локального поиска имеется простой критерий остановки, и если нужный объект существует, то рано или поздно он будет обнаружен. Если же объект не существует, то алгоритм может работать бесконечно долго. Вероятностные алгоритмы, которые дают точное решение, но время работы которых является случайной величиной, называются алгоритмами типа Лас-Вегас [70]. В отличие от них алгоритмы типа Монте-Карло могут никогда не дать точное решение, но их погрешность стремится к нулю с ростом числа шагов алгоритма. Ниже будут приведены примеры алгоритмов типа Лас-Вегас.

4.1 Палиндромные деревья

Пусть G — связный граф и D — его диаметр. Обозначим через $d(G,k)$ число пар вершин в G, находящихся на расстоянии k друг от друга. По определению $d(G,0)$ есть число вершин в G. Полином вида

$$H(G) = H(G,\lambda) = \sum_{k=0}^{D} d(G,k)\lambda^k$$

называют полиномом Хосойа [26, 52, 64].

Определение 7 [27]. Граф G с диаметром D называется H-палиндромным, если равенство $d(G,k)=d(G,D{-}k)$ выполняется при всех $k=0,1,\dots,\lfloor D/2\rfloor$.

Определение 8 [27]. Граф G с диаметром D называется H^*-палиндромным, если равенство $d(G,k)=d(G,D{-}k{+}1)$ выполняется при всех $k=1,\dots,\lceil D/2\rceil$.

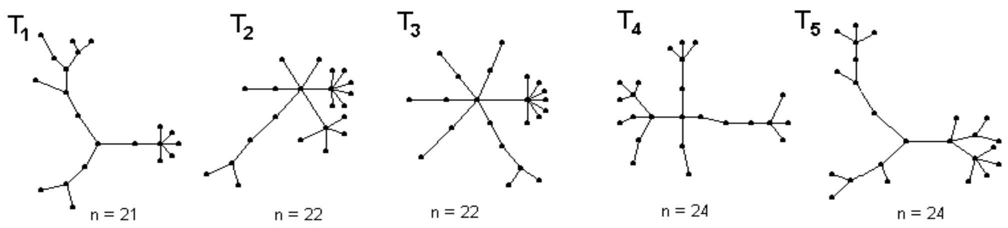

Рис. 3. H-палиндромные деревья

В 1993 году И. Гутман сделал предположение, что H-палиндромные деревья не существуют [42]. Позднее было найдено три примера H^*-палиндромных деревьев с $n\le 12$ вершинами [43] и семейство H-палиндромных графов, не являющихся деревьями [6]. Эти находки стимулировали дальнейший поиск H-палиндромных деревьев, которые и были обнаружены при $n\ge 21$. Для меньшей размерности таких объектов не существует. На рис. 3 изображены деревья, найденные методами локального поиска [45, 46]. Диаметры деревьев равны 6 или 8, а полиномы Хосойа имеют вид:

$$\begin{aligned}
&H(T_1)=21+20\lambda+34\lambda^2+25\lambda^3+31\lambda^4+25\lambda^5+34\lambda^6+20\lambda^7+21\lambda^8\\
&H(T_2)=H(T_3)=22+21\lambda+52\lambda^2+63\lambda^3+52\lambda^4+21\lambda^5+22\lambda^6\\
&H(T_4)=24+23\lambda+37\lambda^2+41\lambda^3+50\lambda^4+41\lambda^5+37\lambda^6+23\lambda^7+24\lambda^8\\
&H(T_5)=24+23\lambda+39\lambda^2+41\lambda^3+46\lambda^4+41\lambda^5+39\lambda^6+23\lambda^7+24\lambda^8.
\end{aligned}$$

В данном случае целевая функция $f(G)$ оптимизационной задачи определяется выражением

$$f(G)=\sum_{k=0}^{\lfloor D/2\rfloor}|d(G,k)-d(G,D-k)|$$

и может интерпретироваться как расстояние до H-палиндромного графа, если он существует. В качестве ограничений задачи выступает требование, чтобы граф был деревом. В таблице 1 приведены результаты поиска H- и H^*-палиндромных деревьев с $n \leq 26$ вершинами. Так как методами локального поиска нельзя доказать несуществование объектов, то полным перебором было проверено, что других H- и H^*-палиндромных деревьев с $n \leq 26$ вершинами нет.

Таблица 1. Число деревьев, H- и H^*-палиндромных деревьев

n	Деревья	H-палиндромные деревья	H^*-палиндромные деревья
4	2	0	1
5	3	0	0
6	6	0	0
7	11	0	1
8	23	0	0
9	47	0	1
10	106	0	0
11	235	0	0
12	551	0	0
13	1301	0	3
14	3159	0	0
15	7741	0	0
16	19320	0	5
17	48629	0	6
18	123867	0	0
19	317955	0	0
20	823065	0	0
21	2144505	1	39
22	5623756	2	11
23	14828074	0	2
24	39299897	2	0
25	104636890	0	410
26	279793450	0	69

Все найденные H-палиндромные деревья имеют четный диаметр. Поэтому было бы интересно найти такое дерево с нечетным диаметром. Интенсивные поиски таких деревьев для $n = 32$ и $n = 36$ не дали положительных результатов, но было замечено, что при любом n, $5 \le n \le 30$ и нечетном диаметре

$$\min f(G) = \lceil n/2 \rceil.$$

Это наблюдение послужило поводом для нового предположения [27].

Гипотеза 2. Не существуют H-палиндромные деревья с нечетным диаметром.

4.2 Построение кодов

Рассмотрим задачу покрытия n-мерного гиперкуба $B^n = \{0,1\}^n$ шарами радиуса R. Центры шаров образуют бинарный покрывающий код (covering code), и задача состоит в том, чтобы для данных n и R найти минимальное по мощности множество шаров, обладающих указанным свойством. Обозначим через $K_2(n,R)$ мощность такого множества и покажем как методы локального поиска могут быть применены к вычислению верхних оценок для $K_2(n,R)$.

Зафиксируем мощность кода M и постараемся построить покрытие гиперкуба M шарами. Если это удастся сделать, то уменьшим M на единицу и повторим процедуру. При заданном M допустимыми решениями оптимизационной задачи будем считать любые наборы из M вершин гиперкуба. Целевая функция для набора C определяется как число непокрытых вершин в B^n, то есть

$$f(C) = |\{x \in B^n \mid d_H(x,C) > R\}|,$$

где $d_H(x,C)$ — расстояние Хемминга от вершины x до множества C. Два кода C_1 и C_2 будем называть соседними, если они отличаются только двумя кодовыми словами и расстояние Хемминга между этими словами не превышает R. Для кода C множество соседних кодов образует его окрестность и соответствующий граф окрестностей, очевидно, является сильно связным. Для $n \le 16$ и $r \le 4$ полученные результаты приведены в таблице 2 [47, 48, 49, 74, 75, 76, 87, 88].

Таблица 2. Границы для $K_2(n, R)$

n	$R=1$	$R=2$	$R=3$	$R=4$
1	1			
2	2	1		
3	2	2	1	
4	4	2	2	1
5	7	2	2	2
6	12	4	2	2
7	16	7	2	2
8	32	$11-12$	4	2
9	$55-62^*$	$15-16$	7	2
10	$105-120^*$	$23-30^*$	9-12	4
11	$178-192$	$36-44$	$12-16$	7
12	$342-380^*$	$61-78^*$	$18-28^*$	$8-12$
13	$598-736^*$	$97-128$	$28-42^*$	$11-16$
14	$1172-1408^*$	$157-256$	$44-64$	$15-28$
15	2^{11}	$309-384^*$	$70-112$	$22-40^*$
16	2^{12}	$512-768$	$114-192^*$	$33-64$

Верхние оценки для $K_2(n, R)$, полученные локальным поиском, помечены звездочкой. Нижние оценки найдены аналитическими методами и приводятся, чтобы обозначить границы для $K_2(n, r)$. Данный подход легко может быть перенесен с бинарных кодов на произвольные q-арные коды. Для $q \geq 3$ методами локального поиска было установлено, что $K_3(6, 1) \leq 73$ и $K_3(7, 1) \leq 186$.

С ростом n размерность задачи о покрытии гиперкуба быстро возрастает. Один из способов решения этой проблемы состоит в том, чтобы как-то ограничить пространство поиска, но при этом не слишком сильно сузить класс рассматриваемых кодов. Например, можно потребовать, чтобы код был линейным. Наилучшие линейные покрывающие коды можно найти в [67]. Следующая теорема описывает класс кодов, который "богаче" линейных, но все еще обладает свойствами, полезными с точки зрения локального поиска.

Пусть I_r — единичная матрица порядка r и $A = (I_r, D)$ — матрица размера $r \times n$ в алфавите $F = \{0, 1, \ldots, q-1\}$. Множество $S \subseteq F^r$ называется R-покрытием F^r с помощью матрицы A, если любое слово $x \in F^r$ может быть представлено в виде

$$x = Ay + s,$$

где $s \in S$ и слово $y \in F^n$ имеет вес не более R.

Теорема 10 [51]. *Если S является R-покрытием F^r с помощью матрицы A, то код*
$$C = \{x \in F^n \mid Ax \in S\}$$
имеет $|S|q^{n-r}$ кодовых слов и радиус покрытия не более R.

Сформулированная теорема позволяет направить локальный поиск при данных q, n, R, r на построение матрицы A и R-покрытия S. [62, 76]. Целевая функция оптимизационной задачи выражает число слов в F^r, которые не могут быть представлены в виде $Ay+S$ ни для какого $y \in F^r$ веса не более R. Окрестность для данных A и S может определяться различными способами, например,

- одновременным изменением как слов из S, так и столбцов матрицы A [62];

- изменением только слов из S и рассмотрением большого числа различных матриц [75, 76];

- изменением только слов из S и рассмотрением матриц A специального вида [60, 76].

Линейные коды могут быть тоже получены данным способом, полагая $S = \{0, \ldots, 0\}$ и меняя только столбцы матрицы A.

Предложенный подход применим также и для кодов, исправляющих ошибки. Полученные результаты в этом направлении можно найти в [51].

Список литературы

[1] Александров Д. А. Алгоритм муравьиной колонии для задачи о минимальном покрытии // XI междунар. Байкальская школа-семинар "Методы оптимизации и их приложения": Труды, Т. 3, Иркутск, 1998. С. 17–20.

[2] Береснев В. Л., Гимади Э. Х., Дементьев В. Т. Экстремальные задачи стандартизации. Новосибирск: Наука, 1978.

[3] Гончаров Е. Н., Кочетов Ю. А. Поведение вероятностных жадных алгоритмов для многостадийной задачи размещения // Дискретный анализ и исследование операций. 1999. Сер. 2. Т. 6, № 1. С. 12–32.

[4] Горбачевская Л. Е., Кочетов Ю. А. Вероятностная эвристика для двухуровневой задачи размещения // XI междунар. Байкальская школа-семинар "Методы оптимизации и их приложения": Труды, Т. 1, Иркутск, 1998. С. 249–252.

[5] Гэри В., Джонсон Д. Вычислительные машины и трудноразрешаемые задачи. М.: Мир, 1982.

[6] Добрынин А. А. Построение графов с палиндромным полиномом Винера // Вычислительные системы. Сб. науч. тр. Новосибирск: Ин-т математики СО РАН, 1994. Вып. 151. С. 37–54.

[7] Еремеев А. В. Разработка и анализ генетических и гибридных алгоритмов для решения задач дискретной оптимизации // Дисс. канд.физ.-мат.наук. Омск, 2000.

[8] Кемени Дж., Снелл Дж. Конечные цепи Маркова. М.: Наука, 1970.

[9] Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. М.: Мир, 1985.

[10] Растригин Л. А. Случайный поиск — специфика, этапы истории и предрассудки // Вопросы кибернетики. 1978. Вып. 33, С. 3–16.

[11] Форд Л., Фалкерсон Д. Потоки в сетях. М.: Мир, 1966.

[12] Черенин В. П. Решение некоторых комбинаторных задач оптимального планирования методом последовательных расчетов. Материалы конф.

по опыту и перспективам примен. матем. методов и ЭВМ в планировании. Новосибирск, 1962.

[13] Черенин В. П., Хачатуров В. Р. Решение методом последовательных расчетов одного класса задач о размещении производства. Сб. Эконом.-матем. методы, Вып. 2, М.: Наука, 1965, С. 279–290.

[14] Aarts E. H. L., Korst J. H. M. Simulated annealing and Boltzmann machines. Chichester: Wiley, 1989.

[15] Aarts E. H. L., Korst J. H. M., Laarhoven van P. J. M. Simulated annealing // Local search in combinatorial optimization. Chichester: Wiley, 1997. P. 91–120.

[16] Aggarwal C. C., Orlin J. B., Tai R. P. Optimized crossover for maximum independent set // Oper. Res. 1997. V. 45, P. 225–234.

[17] Balas E., Niehaus W. Finding large cliques in arbitrary graphs by bipartite matching // Cliques, coloring, and satisfiability. DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 1996, V. 26, P. 29–49.

[18] Balas E., Niehaus W. Optimized crossover-based genetic algorithms for the maximum cardinality and maximum weight clique problems // J. Heuristics. 1998. V. 4, N 4, P. 107–122.

[19] Balas E., Simonetti N. Linear time dynamic programming algorithms for some new classes of restricted TSP's // Proc. IPCO V (Lecture Notes in Computer Science, V. 1084). Berlin: Springer, 1996. P. 316–329.

[20] Binder K. Monte Carlo methods in statistical physics. Berlin: Springer, 1978.

[21] Boese K. D., Kahng A. B., Muddu S. A new adaptivemulti–start technique for combinatorial global optimizations // Oper. Res. Lett. 1994. V. 16, N 2, P. 101–114.

[22] Bock F. An algorithm for solving "travelling-salesman"and related network optimization problems: abstract. Bulletin Fourteenth National Meeting of the Operations Research Society of America. 1958. P. 897.

[23] Bremermann H. J., Roghson J., Salaff S. Global properties of evolution processes // Natural automata and useful simulations. London: Macmillan. 1966. P. 3–42.

[24] Brucher P., Hurink J., Werner F. Improving local search heuristics for some scheduling problems. Part II // Discrete Appl. Math. 1997. V. 72, P. 47–69.

[25] Burkard R. E., Deineko V.G., Woeginger G. J. The traveling salesman problem and the PQ-tree // Proc. IPCO V (Lecture Notes in Computer Science, V. 1084). Berlin: Springer, 1996. P. 490–504.

[26] Caporossi G., Dobrynin A. A., Gutman I., Hansen P. Trees with palindromic Hosoya polynomials // Les Cahiers du GERAD, G-99-10, 1999.

[27] Caporossi G., Dobrynin A. A., Gutman I., Hansen P. Trees with palindromic Hosoya polynomials // Graph Theory Notes N.Y. V. 37, 1999. P. 10–16.

[28] Caporossi G., Hansen P. Variable neighborhood search for extremal graphs: 1 The AutoGraphiX system // Discrete Math. 2000. V. 212, N 1–2. P. 29–44.

[29] Černý V. Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm // J. Opt. Theory Appl. 1985. V. 45. P. 41–51.

[30] Croes G. A. A method for solving traveling salesman problems // Oper. Res. 1958. V. 6, P. 791–812.

[31] Eiben A. E., Raue P. E., Ruttkay Zs. Genetic Algorithms with multiparent recombination // Parallel Problem Solving from Nature III. Berlin: Springer Verlag, 1994. (Lecture Notes in Computer Science, V. 866) P. 78–87.

[32] Eremeev A. V. A genetic algorithm with a non-binary representation for the set covering problem // Operations Research Proceedings 1998. Berlin: Springer Verlag. 1999. P. 175–181.

[33] Fridman M. L., Johnson D. S., McGeoch L. A., Ostheimer G. Data structures for traveling salesman // J. Algorithms. 1995. V. 18, P. 432–479.

[34] Glover F. Tabu search: part I // ORSA J. Comp. 1989. V. 1. P. 190–206.

[35] Glover F. Tabu search: part II // ORSA J. Comp. 1990. V. 2. P. 4–32.

[36] Glover F.(Ed.) Tabu search methods for optimization // Feature Issue of Euro. J. Oper. Res. 1998. V. 106, N. 2–3.

[37] Glover F., Laguna. M Tabu search. Boston: Kluwer Acad. Publ. 1997.

[38] Goldberg D. E. Genetic algorithms in search, optimization, and machine learning. Reading, MA: Addison-Wesley. 1989.

[39] Grover L. K. Local search and the local structure of NP-complete problems // Oper. Res. Lett. 1992. V. 12, N. 4. P. 235–244.

[40] Gutin G. Exponential neighborhood local search for the traveling salesman problem // Comput. Oper. Res. 1999. V. 26, P. 313–320.

[41] Gutin G., Yeo A. Small diameter neighborhood graphs for the traveling salesman problem: four moves from tour to tour // Comput. Oper. Res. 1999. V. 26, P. 321–327.

[42] Gutman I. Some properties of the Wiener polynomial // Graph Theory Notes N.Y. 1993. V. 25, P. 13–18.

[43] Gutman I., Estrada E., Ivanciuc O. Some properties of the Wiener polynomial of trees // Graph Theory Notes of New York, V.36, New York Akademy of Sciences, 1999. P. 7–13.

[44] Hajek B., Sasaki G. Simulated annealing: to cool it or not // Sys. Contr. Lett. 1989. V. 12, P. 443–447.

[45] Hansen P., Mladenović N. An introduction to variable neighborhood search // Meta-heuristics: advances and trends in local search paradigms for optimization. Boston: Kluwer. Acad. Publ. 1998. P. 433–458.

[46] Hansen P., Mladenović N. Variable neighborhood search: principles and applications // Les Cahiers du GERAD, G-98-20, 1998.

[47] Hämäläinen H. O., Honkala I. S., Kaikkonen M. K., Litsyn S. N. Bounds for binary multiple covering codes // Des. Codes Cryptog. 1993. V. 3, N 3. P. 251–275.

[48] Hämäläinen H. O., Honkala I. S., Litsyn S. N., Östergård P. R. J. Bounds for binary codes that are multiple coverings of the farthest-off points // SIAM J. Discrete Math. 1995. V. 8, N 2. p. 196–207.

[49] Hämäläinen H. O., Rankinen S. Upper bounds for football pool problems and mixed covering codes // J. Combinatorial Theory, Ser. A. 1991. V. 56, N 1. P. 84–95.

[50] Holland J. H. Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press. 1975.

[51] Honkala I. S., Östergård P. R. J. Code design // Local search in combinatorial optimization. Chichester: Wiley, P. 441–456.

[52] Hosoya H. On some counting polynomials in chemistry // Discrete Appl. Math. 1988. V. 19, N 2. P. 239–257.

[53] Johnson D. S., Aragon C. R., McGeoch L. A., Schevon C. Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning // Oper. Res. 1989. V. 37, N 6. P. 865–892.

[54] Johnson D. S., McGeoch L. A. The traveling salesman problem: a case study // Local search in combinatorial optimization. Chichester: Wiley, P. 215–310.

[55] Johnson D. S. Local optimization and the traveling salesman problem // Automata, Languages, and Programming. Berlin: Springer Verlag, 1990. (Lecture Notes in Computer Science, V. 443) P. 446–461.

[56] Kernighan B. W., Lin S. An efficient heuristic procedure for partitioning graphs // Bell System Technical Journal. 1970. V. 49, P. 291–307.

[57] Kirkpatrick S., Toulouse G. Configuration space analysis of traveling salesman problems // J. de Phys. 1985. V. 46, P. 1277–1292.

[58] Kirkpatrick S., Gelatt C. D., Vecchi M. P. Optimization by simulated annealing // Science. 1983. V. 220, P. 671–680.

[59] Kochetov Yu. A., Goncharov E. N., Behavior of a probabilistic tabu search algorithm for the multi stage uncapacitated facility location problem // Proc. INFORMS-KORMS, Seul 2000 (http://www.math.nsc.ru/LBRT/k5/Kochetov/publ.html).

[60] Koschnick K. -U. New upper bound for the football pool problem for nine matches // J. Combinatorial Theory, Ser. A. 1993. V. 62, P. 162–167.

[61] Krentel M. W. Structure in locally optimal solutions // 30th Annual Symposiun on Foundation of Computer Science. 1989, P. 216–222.

[62] Laarhoven van P. J. M., Aarts E. H. L., Lint van J. H., Wille L. T. New upper bounds for the football pool problem for 6,7 and 8 matches // J. Combinatorial Theory, Ser. A. 1989. V. 52, P. 304–312.

[63] Laarhoven van P. J. M., Aarts E. H. L., Lenstra J. K. Job shop scheduling by simulated annealing // Oper. Res. 1992. V. 40, P. 113–125.

[64] Lepović M., Gutman I. A collective property of trees and chemical trees // J. Chem. Inform. Comput. Sci. 1998. V. 38, P. 823–826.

[65] Lin S. Computer solutions of the traveling salesman problem // Bell Sys. Tech. J. 1965. V. 44, P. 2245–2269.

[66] Lin S., Kernighan B. W. An efficient heuristic algorithm for the traveling-salesman problem // Oper. Res. 1973. V. 21, P. 498–516.

[67] Lobstein A., Press V. The lenght function: a resived table // Algebraic coding, (Lecture Notes in Computer Science V. 781). Berlin: Springer, 1994. P. 51-55.

[68] Macken C. A., Hagan P. S., Perelson A. S. Evolutionary walks on rugged landscapes // SIAM J. Appl. Math. 1991. V. 51, N 3. P. 799–828.

[69] Mirchandani P. B., Francis R. L. Discrete Location Theory. New York: John Wiley and Sons, 1990.

[70] Motwani R., Raghavan P. Randomized algorithms. Cambridge: Cambridge Univ. Press, 1995.

[71] Mühlenbein H. Parallel genetic algorithm, population dynamics and combinatorial optimization // Proc. Third Inter. Conf. Genetic Alg. San Mateo: Morgan Kaufman, 1989. P. 416–421.

[72] Nicholson T. A. J. A sequential method for discrete optimization problems and its application to the assignment, traveling salesman and tree scheduling problems // J. Inst. Math. Appl. 1965. V. 13, P. 362–375.

[73] Osman I. H., Laporte G. Metaheuristics: a bibliography // Ann. Oper. Res. 1996. V. 63, P. 513–628.

[74] Östergård P. R. J. A new binary code of length 10 and covering radius 1 // IEEE Trans. Inform. Theory. 1991. V. 37, N 1. P. 179–180.

[75] Östergård P. R. J. Construction methods for mixed covering codes. // Analysis, algebra, and computers in mathematical research. 1994. New York: Marcel Dekker, P. 387–408.

[76] Östergård P. R. J. New upper bounds for the football pool problem for 11 and 12 matches // J. Combinatorial Theory, Ser. A. 1994. V. 67, P. 161–168.

[77] Page E. S. On Monte Carlo methods in congestion problems: I, searching for an optimum in discrete situations // Oper. Res. 1965. V. 13, P. 291–299.

[78] Punnen A. P. The traveling salesman problem: new polynomial approximation algorithms and domination analysis // Research Report. St.John: University New Brunswick. 1996.

[79] Rechenberg I. Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der Biologischen Information, Freiburg: Fromman, 1973.

[80] Reiter S., Sherman G. Discrete optimizing // J. Soc. Indust. Appl. Math. 1965. V. 13, P. 864–889.

[81] Schwefel H. P. Numerical optimization of computer models. Chichester: Wiley, 1981.

[82] Stadler P. F. Corelation in landscapes of combinatorial optimization problems // Europhys. Lett. 1992. V. 20, P. 479–482.

[83] Stadler P. F., Schnabl W. The landscape of the traveling salesman problem // Physics Letters A. 1992. V. 161, P. 337–344.

[84] Tovey C. A. Local improvement on discrete structures // Local search in combinatorial optimization. 1997. Chichester: Wiley, P. 57–90.

[85] Verhoeven M. G. A. Tabu search for resource-constrained scheduling // European J. Oper. Res. 1998. V. 106, N 2. P. 266–276.

[86] Verhoeven M. G. A., Swinkels P. C. J., Aarts E. H. L. Parallel local search and the traveling salesman problem // Working paper, Philips research laboratories, Eindhoven, 1995.

[87] Wille L. T. Improved binary code coverings by simulated annealing // Congressus Numerantium, 1990. V. 73, P. 53–58.

[88] Wille L. T. New binary covering codes obtained by simulated annealing // IEEE Trans. Inform. Ther. 1996. V. 42, P. 300–302.

[89] Yannakakis M. Computational complexity // Local search in combinatorial optimization. 1997. Chichester: Wiley, P. 19–55.