СИСТЕМНИЙ АНАЛІЗ

И.В СЕРГИЕНКО, В.П. ШИЛО

УДК 519.854

ТЕХНОЛОГИЯ ЯДРА ДЛЯ РЕШЕНИЯ ЗАДАЧ ДИСКРЕТНОЙ ОПТИМИЗАЦИИ

Аннотация. Предложена новая технология для решения задач дискретной оптимизации, формирующая «ядро» решения, что позволяет эффективно случайно возмущать это решение в схемах итерирования. Проведено сравнительное исследование двух версий нового алгоритма решения квадратичной задачи о назначениях (с технологией выделения ядра и без неё) с современными алгоритмами, которое показало перспективность использования этой технологии как по быстродействию, так и по возможности получения лучших решений. Технологию ядра легко инкорпорировать в уже существующие алгоритмы.

Ключевые слова: технология ядра, квадратичная задача о назначениях QAP, задача о максимальном взвешенном разрезе графа WMaxCut, задача о коммивояжере TSP, квадратичная задача булевого программирования UBQP, задача составления расписаний JSP, вычислительный эксперимент, сравнительное исследование алгоритмов.

ВВЕДЕНИЕ

В настоящее время интенсивное развитие получило дискретное программирование, ориентированное на решение различных типов задач, которые возникают при принятии решений в разных сферах деятельности человека. Актуальность исследования методов решения таких задач обусловлена их важностью и сложностью решения. Разработаны методы решения как общих, так и специальных задач дискретной оптимизации. Исследование теоретических аспектов этих методов достаточно продвинулось, однако считать, что эти методы приемлемы для решения широкого спектра практических задач пока невозможно. Кроме того, результаты теории сложности для оптимизационных задач показывают, что рассчитывать на создание точных методов, гарантирующих решение задач дискретной оптимизации, приемлемых для многих практических целей, по-видимому, невозможно. Хотя и имеются примеры решения дискретных задач с несколькими сотнями переменных, в общем случае нельзя пока гарантировать получение оптимального решения в задачах с сотнями переменных. Отсюда следует вывод о необходимости разработки и использования широкого спектра приближенных и эвристических методов для решения общих и специальных задач дискретной оптимизации. Отметим, что построение приближенных решений для многих задач столь же сложно, как и построение оптимального решения.

Оценка эффективности алгоритмов решения задач дискретной оптимизации является чрезвычайно важной проблемой. Для ее решения необходимы наличие представительного множества тестовых задач и детальный сравнительный анализ алгоритмов. Алгоритмы можно исследовать как теоретическим, так и экспериментальным путем. Поскольку первый подход можно использовать для ограниченно-

го круга задач, целесообразно рассматривать вопрос об оценке эффективности алгоритмов решения задач дискретной оптимизации в основном с помощью машинного эксперимента. Такой подход считается рациональным и потому, что существующие теоретические разработки относятся в основном к оценке трудоемкости для самых сложных задач исследуемого класса, в то время как трудоемкость реальных задач в среднем может быть существенно ниже.

В Институте кибернетики им. В.М. Глушкова НАН Украины интенсивно ведутся работы по созданию новых и модификации известных методов дискретной оптимизации, сравнительному анализу их с лучшими известными методами решения различных классов сложных дискретных оптимизационных задач, в частности задач большой размерности.

Настоящая статья посвящена технологии выделения ядра для решения задач дискретной оптимизации и ее практическому применению для квадратичной задачи о назначениях. Для этой задачи получены результаты сравнительного исследования двух версий повторяемого итерированного алгоритма табу (с технологией ядра и без нее) с лучшими современными алгоритмами, подтверждающие эффективность технологии ядра.

Отметим, что теоретические исследования эффективности методов решения задач дискретной оптимизации по-прежнему актуальны. Большой интерес представляет разработка технологий решения задач дискретной оптимизации и исследования их эффективности с помощью вычислительного эксперимента особенно при решении задач большой размерности.

При решении задач дискретной оптимизаци сталкиваются со следующей проблемой: получено «хорошее» решение задачи, как получить еще лучшее решение? Предлагается множество рекомендаций, как поступить (собственно, все эвристики современной дискретной оптимизации представляют свод этих советов). Но ответ прост — универсального рецепта нет, поскольку все зависит от решаемой задачи и применяемого алгоритма. В работе [1] приводится следующий пример: С. Лин предложил идею, названную сведением. Он обнаружил, что в конкретных задачах некоторые черты присущи всем хорошим решениям и предложил их фиксировать, что может ускорить получение еще лучших решений. Однако, как обычно бывает с эвристиками, можно высказать также и прямо противоположное соображение. Если такие общие черты обнаружены, то они, скорее, запрещенные. Если алгоритм длительное время не находит глобального оптимума, то запрещение этих черт может ускорить его получение.

По мнению авторов данной работы, исходить все-таки следует из принципа ближайшей оптимальности (proximable optimality principle, POP) [13], согласно которому хорошие решения имеют сходные структуры. Это основное предположение касается большинства эвристик, и неизвестно, что произошло бы с нашей цивилизацией, если бы этот принцип не выполнялся для подавляющего числа оптимизационных задач.

Известным способом использования хороших решений является итерирование. Согласно его схеме фиксируется некоторое количество компонентов хорошего решения, а остальные компоненты случайно возмущаются. Остаются открытыми основные вопросы: какие компоненты следует фиксировать и как возмущать незафиксированные компоненты.

Более полные ответы на эти вопросы предлагает метод глобального равновесного поиска [2]. Случайное возмущение компонентов задается вероятностью возмущения, определяемой количеством и качеством хороших решений, в которых имеется этот компонент. Однако мечта любого оптимизатора — своеобразный философский камень — нахождение ядра хорошего решения, с помощью которого можно быстро получить очень хорошие решения. Попытаемся математически описать понятие ядра.

ТЕХНОЛОГИЯ ВЫДЕЛЕНИЯ ЯДРА РЕШЕНИЯ

Пусть задача оптимизации формулируется следующим образом: найти

$$x^* = \arg\min_{x \in D^n} f(x), \tag{1}$$

где D^n — дискретное конечное множество элементов $x = (x_1, \dots, x_n)$.

Предположим, что $\bar{x} = (\bar{x}_1, \dots, \bar{x}_n)$ — решение задачи (1). Зададим число k, $0 \le k \le n$, и пусть $L_k = \{l_1, \dots, l_k\}$ — некоторое множество индексов переменных задачи (1). Рассмотрим задачу: найти

$$\psi(L_k, \bar{x}) = \min_{x \in D^n} f(x) \tag{2}$$

при ограничениях

$$x_l = \bar{x}_l, \ l \in L_k. \tag{3}$$

Пусть $L_k^*(\bar{x}) = \arg\min_{L_k} \psi(L_k, \bar{x})$ и $k^*(\bar{x})$ —максимальное k, $k = 0, \dots, n$, для которого выполняется условие $\psi(L_k^*, \bar{x}) = f(x^*)$. Множество $L_{k^*}^*(\bar{x})$ является тем ядром решения $\bar{x}$, которое стремятся получить. Но поскольку для получения только одного k необходимо C_n^k раз решать задачу (2), (3), то прямой путь к нахождению ядра становится бесперспективным. Мы попытаемся, усилив РО-принцип, предложить способ получения квазиядра, которое будет полезно при приближенном решении задачи (1).

Рассмотрим следующую оптимизационную задачу: найти

$$L_k^\wedge(\bar{x}) = \arg\min_{L_k} f(\bar{x}, L_k). \tag{4}$$

Задача состоит в нахождении множества $L_k^\wedge$, на котором целевая функция исходной задачи (1) будет минимальной. Рассмотрим задачу, подобную (2), (3), для различных значений k, $k = 0, \dots, n$: найти

$$\psi(L_k^\wedge, \bar{x}) = \min_{x \in D^n} f(x)$$

при ограничениях

$$x_l = \bar{x}_l, \ l \in L_k^\wedge.$$

Решая ее приближенно, находим $k^\wedge = \arg\min_k \psi(L_k^\wedge, \bar{x})$. Если $\psi(L_{k^\wedge}^\wedge, \bar{x}) < f(\bar{x})$, то улучшим решение $\bar{x}$.

Определение 1. Множество $L_k^\wedge(\bar{x})$ называется k min-ядром решения $\bar{x}$ задачи (1).

Найдем расстояние между множествами $L_k^1(\bar{x})$ и $L_k^2(\bar{x})$: $\rho(L_k^1(\bar{x}), L_k^2(\bar{x})) = k - |L_k^1(\bar{x}) \cap L_k^2(\bar{x})|$, где $|S|$ — мощность множества S.

Усиленный принцип оптимальности утверждает, что

$$\rho(L_k^*(\bar{x}), L_k^\wedge(\bar{x})) < \frac{1}{C_n^k} \sum_{L_k} \rho(L_k^*(\bar{x}), L_k(\bar{x})). \tag{5}$$

Предположим, что по окончании процесса интенсификации оптимизационным алгоритмом получено решение $\bar{x} = (\bar{x}_1, \dots, \bar{x}_n)$. Если в процессе диверсификации изменяются d компонентов решения $\bar{x}$, то предлагается найти $(n-d)$ min-ядро этого

решения и использовать для изменения те компоненты, которые не вошли в ядро. Ввиду усиленного принципа оптимальности, если в решении $\bar{x}$ какие-то компоненты совпадают с компонентами глобального оптимума, шансы на то, что они попадут в k min-ядро $L_k^{\wedge}(\bar{x})$ без изменения, возрастают. Поэтому имеем естественный шаг, который при разумных предположениях даст спектр перспективных возмущенных решений.

Рассмотрим примеры определения k min-ядра для различных задач.

Предположим, что целевая функция задачи (1) представима в виде

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij}(x_i, x_j). \tag{6}$$

Заметим, что важные в практических приложениях задачи QAP [3] (квадратичная задача о назначениях), WMaxCut [4] (задача о максимальном взвешенном разрезе графа) и многие другие имеют целевую функцию вида (4). Математическая постановка задачи выбора k min-ядра с целевой функцией вида (4) формулируется следующим образом: найти

$$\arg\min_{z \in B^n} f(z, \bar{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij}(\bar{x}_i, \bar{x}_j)\, z_i z_j \tag{7}$$

при условиях

$$\sum_{j=1}^{n} z_j = k, \tag{8}$$

$$z = (z_1, \ldots, z_n), \;\; z_j = 0 \vee 1, \;\; j = 1, \ldots, n, \tag{9}$$

где B^n — множество n-мерных векторов, компоненты которых принимают значения 0 или 1.

Имеем квадратичную задачу булевого программирования, для которой разработано много хороших приближенных алгоритмов [5]. Заметим, что если задачи QAP уже трудно решать при $n \geq 100$, то квадратичные задачи вида (7)–(9) такой же размерности решаются быстро и с хорошей точностью.

Для многомерной булевой задачи о ранце [1] задача нахождения k min-ядер сводится к выбору k ненулевых компонентов $\bar{x}_j$ вектора $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_n)$, которым соответствуют наибольшие (наименьшие) значения коэффициентов целевой функции.

В задаче составления расписания JSP [6] заданы n работ, которые необходимо выполнить за минимальное время на множестве из m машин. Каждая работа включает последовательности операций. Обозначим o_{ij} операцию работы i, $i = 1, \ldots, n$, которая должна быть выполненной j-й по счету, $j = 1, \ldots, m$. Каждой операции o_{ij} поставлены в соответствие машина m_{ij}, на которой ее можно выполнить, и время выполнения t_{ij}. Пусть s_{ij} — время начала выполнения операции o_{ij}. Каждая машина в процессе работы может выполнять только одну операцию, при этом выполнение операции не может прерываться. Цель задачи — нахождение такого расписания, т.е. распределения операций с временными интервалами по машинам, чтобы время выполнения всех работ было минимальным.

Графически задача составления расписания может быть сведена к задаче поиска кратчайшего пути на графе, в котором каждой операции o_{ij} соответствует вершина v_{ij}. Каждая пара вершин v_{ij} и v_{ik}, соответствующих операциям одной работы, соединена дугой, направление которой определяет, какая из операций должна выполняться первой. Обозначим E1 множество таких дуг. При этом каждая пара вершин, соответствующих операциям, выполняемым на одной машине,

соединена дугой без направления. Обозначим E2 множество ненаправленных дуг. Задача составления расписания состоит в том, чтобы задать направление каждой дуге из E2 и в результате получить ориентированный граф, в котором длина максимального пути была бы минимальной. Очевидно, что в полученном графе после задания направлений всем дугам из E2 не должно быть циклов, т.е. конечный граф должен задавать топологическую сортировку на множестве операций. Если задан ориентированный граф (после задания направлений дугам из E2), который представляет какое-то решение задачи составления расписания, то ядро, состоящее из k работ, можно получить, выбрав $n-k$ работ и удалив все вершины, соответствующие операциям этих работ. Длина максимального пути после такого удаления задаст качество выбранного ядра.

И, наконец, для задачи о коммивояжере TSP [1] задача о нахождении k min-ядра формулируется следующим образом: найти

$$\arg\min_{z\in B^n} f(\pi)=\sum_{i=1}^{n} d_{\overline{\pi}_i\overline{\pi}_{i+1}} z_i z_{i+1} \tag{10}$$

при условиях

$$\sum_{j=1}^{n} z_j = k, \tag{11}$$

$$z=(z_1,\dots,z_n),\ \ z_j=0\vee 1,\ \ j=1,\dots,n, \tag{12}$$

где перестановка $\overline{\pi}=(\overline{\pi}_1,\dots,\overline{\pi}_n,\overline{\pi}_{n+1})$, $\overline{\pi}_1\equiv\overline{\pi}_{n+1}$, — полученное решение задачи о коммивояжере.

Пусть в процессе решения задачи найдено текущее ядро $\widetilde{z}=(\widetilde{z}_1,\dots,\widetilde{z}_n)$ и для некоторых r и s выполняется неравенство $\widetilde{z}_r\neq\widetilde{z}_s$. Предположим для определенности, что $\widetilde{z}_r=1$ и $\widetilde{z}_s=0$. Тогда приращение целевой функции (10) при изменении $\widetilde{z}_r=1-\widetilde{z}_r$, $\widetilde{z}_s=1-\widetilde{z}_s$ имеет вид

$$\Delta_{rs} f(\widetilde{\pi})=d_{r_l r_r}-d_{r_l r}-d_{r r_r}+d_{s_l s}+d_{s s_r}-d_{s_l s_r},$$

где $\widetilde{z}_{r_l}=\widetilde{z}_{rr}=\widetilde{z}_{s_l}=\widetilde{z}_{s_{rl}}=1$, r_l, r_r и s_l, s_r — ближайшие индексы слева и справа соответственно к индексам r и s в перестановке $\overline{\pi}$. Как видим, эти приращения рассчитываются для любых $\widetilde{z}_r\neq\widetilde{z}_s$ без особых вычислительных усилий; следовательно, можно разработать приближенный алгоритм решения задачи (10)–(12).

На примере задачи о коммивояжере хорошо иллюстрировать понятия k min(max)-ядер. Пусть задача о коммивояжере поставлена для городов США. Тогда в k min-ядро войдут, скорее всего, города одного из густонаселенных штатов, а в k max-ядро — города из различных штатов. Этот пример свидетельствует и об устойчивости ядер. Вряд ли после изменения решения на Гавайских островах следует корректировать решение в штате Айова или изменять последовательность посещений столиц штатов.

Для проверки эффективности технологии ядра была выбрана задача QAP — одна из сложных и практически важных задач дискретного программирования. Кроме того, для экспериментов использовалась задача tai100a, наиболее трудно решаемая тестовая задача. Для решения был выбран новый повторяемый итерированный алгоритм табу (его описание готовится к публикации) и проведены сравнительные вычислительные эксперименты по двум версиям алгоритма: с технологией ядра RITSLK и без нее RITSL.

Напомним, что математическая постановка QAP имеет вид: найти

$$\min_{\pi\in\Pi_n} f(\pi)=\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{\pi_i\pi_j}, \tag{13}$$

где $A=[a_{ij}]$ и $B=[b_{rs}]$ — квадратные матрицы порядка n; Π_n — множество всех перестановок $\{1,\dots,n\}$; π_i задает номер локации объекта i.

Обозначим $\overline{\pi} = (\overline{\pi}_1, \ldots, \overline{\pi}_n)$ полученное решение задачи (13). Тогда целевая функция задачи определения k min-ядра (6) такова: $f(z, \overline{\pi}) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\overline{\pi}_i \overline{\pi}_j} z_i z_j$. Для решения задачи (7)–(9) использовался случайный повторяемый локальный поиск.

РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ

Первый эксперимент состоял в проверке выполнимости усиленного принципа оптимальности для исследуемой квадратичной задачи о назначениях tai100a. Заметим, что требование о необходимости теоретического обоснования выполнимости этого принципа может быть непосредственно получено при обосновании эффективности локального поиска. Для эксперимента было взято лучшее из найденных решений π^*. В этом решении случайным образом отбирались k объектов, $k = n - d \cdot n/20$, $d = 1, \ldots, 19$, которые фиксировались. Остальные объекты случайным образом обменивались локациями. В результате получали решение $\overline{\pi}$ и множество L_k^*, состоящее из индексов фиксированных объектов. В этом решении выделялось k min-ядро $L_k^{\wedge}$ и случайно выбиралось множество L_k. При каждой попытке выбора подсчитывались расстояния $\rho(L_k^*(\overline{\pi}), L_k^{\wedge}(\overline{\pi}))$ и $\rho(L_k^*(\overline{\pi}), L_k(\overline{\pi}))$, которые затем по 1000 попыткам усреднялись. На рис. 1 отображен график усредненных расстояний. Результаты проведенного эксперимента подтверждают применимость усиленного принципа оптимальности.

В алгоритмах RITSL и RITSLK использовались два способа изменения решения. Обозначим число изменяемых компонентов n_d. Первым способом RD (случайное возмущение) случайно выбирались $n - n_d$ объектов из возмущаемого решения, при этом локации их не изменялись. Остальные n_d объектов случайно обменивались между собой локациями.

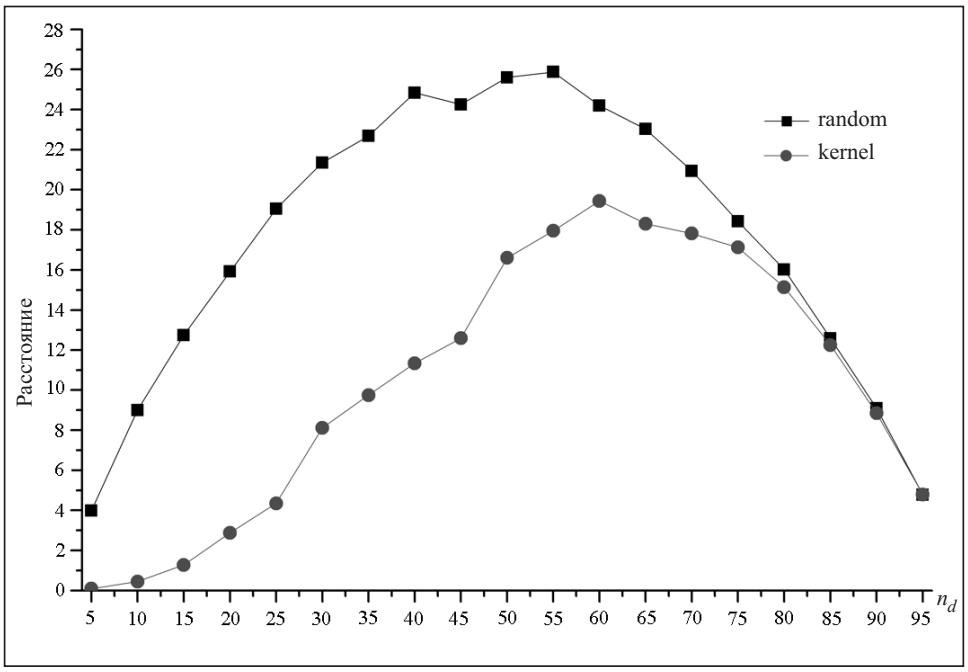

Рис. 1. Графики усредненных расстояний $\rho(L_k^*(\overline{\pi}), L_k^{\wedge}(\overline{\pi}))$ и $\rho(L_k^*(\overline{\pi}), L_k(\overline{\pi}))$ по числу попыток

Второй способ KD (с выделением ядра) отличается лишь тем, что выбиралось $n - n_d$ min-ядро возмущаемого решения и объекты его оставались на прежних локациях, а n_d объектов, не входящих в ядро, случайно обменивались между собой локациями.

Вначале были проведены эксперименты по выделению k min-ядра. Для этого были выбраны предварительно полученные решения, которые упорядочивались по возрастанию целевой функции. Далее каждое решение возмущалось описанными выше способами.

В рамках эксперимента число компонентов n_d составляло 30, а возмущенное решение (его расстояние до возмущаемого решения не превосходило n_d) улучшалось алгоритмом табу. Далее рассматривались значения функции цели полученного решения и его расстояние от возмущаемого решения и они усреднялись по 10 попыткам возмущения. Считалось успехом, если полученное значение функции цели было строго меньше значения функции цели возмущаемого решения. На рис. 2 показаны лучшие значения целевой функции (рекорды) и средние значения целевой функции улучшенных решений. На рис. 3 приведены средние расстояния полученных решений до возмущаемого решения и число успехов для двух способов возмущения.

Как видно из графиков рис. 2, 3, качество полученных решений при втором способе KD возмущения выше, а среднее расстояние до возмущаемого решения меньше, чем при первом способе. Особо отметим, что второй способ является более предпочтительным, чем первый по количеству и качеству успехов (сравнить результаты на графике рис. 2, начиная с 31 решения).

Рассматриваемые алгоритмы решения этой задачи реализованы на языке C++, среда Microsoft Visual Studio 2017, все вычислительные эксперименты проводились с использованием PC с Intel® Core QUAD CPU Q9550 2.83GHz и 8.0GB оперативной памяти. Каждая задача решалась пять раз с помощью портфеля из восьми одинаковых алгоритмов (т.е. каждая задача решалась 40 раз при различных значениях датчика псевдослучайных чисел), выделенный лимит времени для одной попытки решения задачи составлял 2 ч. Так как загрузка процессоров при использовании портфеля алгоритмов была максимальной, то скорость обработки отдельного алгоритма уменьшалась приблизительно в 2,2 раза.

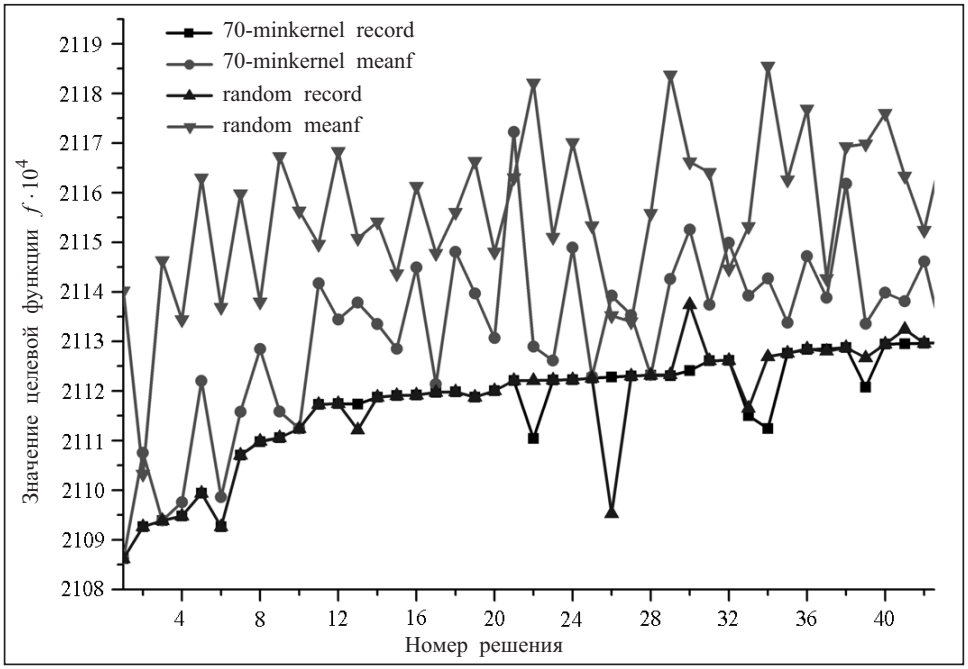

Рис. 2. Рекорды и средние значения целевой функции полученных решений для двух способов возмущения

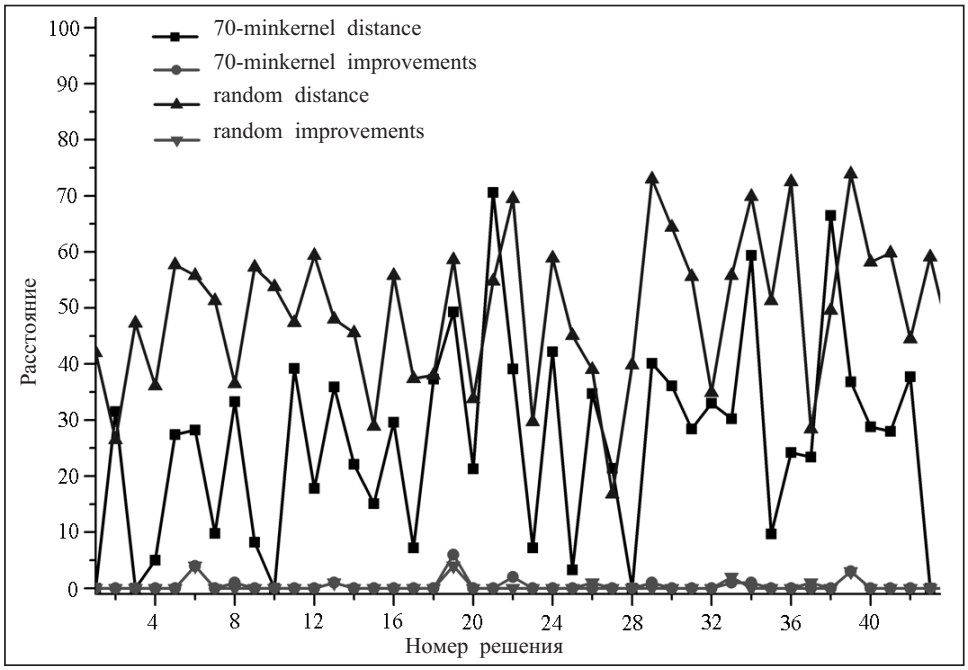

Рис. 3. Графики среднего расстояния полученных решений до возмущаемого решения и число успехов для двух способов возмущения

В работе [7] описаны подобные экспериментальные исследования, поэтому мы воспользовались данными этой статьи для сравнения с разработанными алгоритмами. Выбрано шесть современных алгоритмов, которые показывают хорошие результаты:

- Cooperative Parallel Tabu Search (CPTS) [8];
- Iterated Tabu Search (ITS) [9];
- Improved Hybrid Genetic Algorithm (IHGA) [10];
- Breakout Local Search (BLS) [11];
- Breakout Memetic Algorithm (BMA) [7];
- Population-based Iterated Local Search (PILS) [12].

В табл. 1 приведены результаты сравнительного исследования этих алгоритмов с версиями RITSLK (с определением ядра) и RITSLR (без определения ядра) алгоритма RITSL для задачи tai100a. Отметим, что BKS = 21052466 — известный из литературы рекорд. Для каждого алгоритма приведено отклонение dev (в процентах) среднего значения целевой функции, полученного с помощью данного алгоритма при 40 независимых просчетах, от BKS.

Так как расчеты проводились портфелями из восьми алгоритмов, то кроме среднего значения целевой функции, полученного алгоритмами, рассчитывалось лучшее значение (рекорд), найденное портфелем, которое и усреднялось по запускам портфеля (пять раз). На рис. 4 приведены средние значения целевой функции задачи QAP, полученные алгоритмами RITSLK (с определением ядра) и RITSL (без определения ядра) за определенный промежуток времени и средние лучшие значения. На рис. 5 графически представлены отклонения (в процентах) средних значений целевой функции задачи QAP, полученные с помощью данных

Таблица 1

Алгоритм	RITSLK	RITSL	CPTS	ITS	IHGA	BLS	BMA	PILS
dev, %	**0.290**	0.304	0.589	0.427	0.606	0.430	0.405	1.027

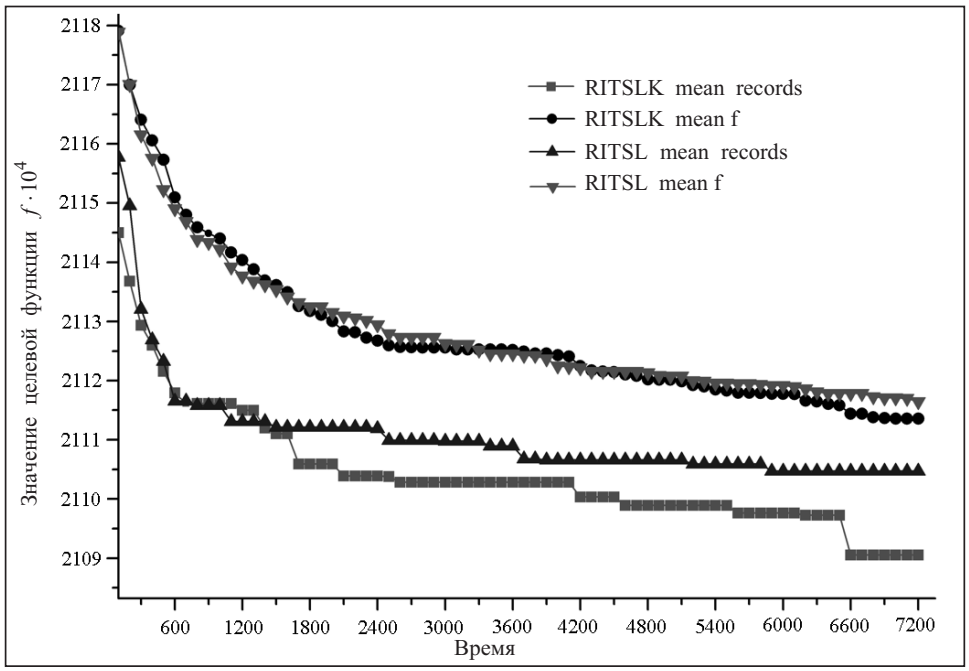

Рис. 4. Графики динамики усредненных лучших значений и значений целевой функции решений, найденных алгоритмами RITSLK и RITSL

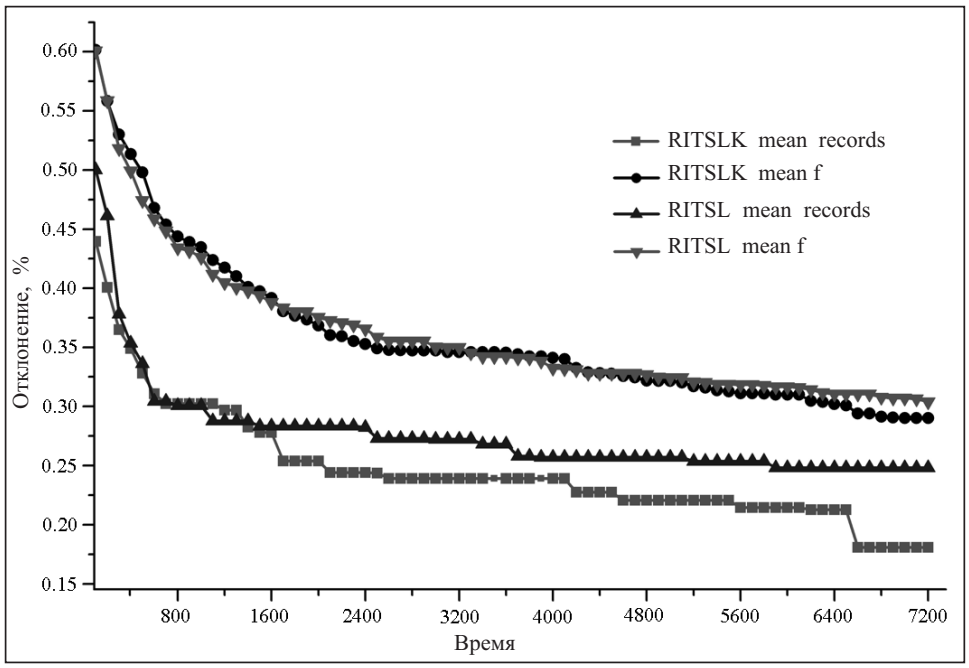

Рис. 5. Графики динамики отклонения (в процентах) усредненных лучших значений и значений целевой функции решений, найденных алгоритмами RITSLK и RITSL, от наилучшего известного рекорда

алгоритмов, от наилучших известных значений на протяжении определенного промежутка времени и отклонения средних лучших значений целевой функции.

Как видим из графика рис. 4, особенно хорошо технология ядра проявляет себя при использовании алгоритма в объединении (в данном случае — портфеле).

ЗАКЛЮЧЕНИЕ

Из анализа результатов экспериментальных расчетов можно сделать вывод о перспективности дальнейшей разработки технологии выделения ядра. Поскольку эта технология работает при решении квадратичной задачи о назначениях QAP tai100, то при решении задач WMaxCut, TSP и JSP, в которых в большинстве случаев POP принцип выполняется глобально, эта технология должна еще более успешно применяться.

XXI век сформировал новые вызовы: сложнейшие задачи огромной размерности в области экономики, медицины, биологии и т.д. невозможно решать без новых подходов. В Институте кибернетики НАН Украины основной упор [14, 15] сделан на изучение различных объединений алгоритмов, решающих задачу на многопроцессорном вычислительном комплексе, выборе их оптимального состава, способов взаимодействия. Также ведется разработка элементов искусственного интеллекта, что позволит автоматизировать решение этих задач в реальном времени. Создание новых алгоритмов, специально предназначенных для решения оптимизационных задач в составе объединений алгоритмов, является важной частью этих исследований. Различные технологии обработки решений расширяют спектр возможностей разрабатываемых алгоритмов.

Скорость обработки отдельного алгоритма в режиме портфеля уменьшалась по сравнению с последовательным режимом приблизительно в 2.2 раза, т.е. результаты, приведенные в табл. 1, следует считать полученными за 55 мин последовательного режима. При этом отметим, что последовательный алгоритм, предназначенный для решения оптимизационной задачи, и алгоритм, предназначенный для решения этой же задачи в составе объединения алгоритмов, совершенно разные и сравнивать их в последовательном режиме нельзя. Необходимо, по крайней мере, составлять портфели из последовательных алгоритмов и проводить с ними эксперименты. Не делать этого — значит, не ориентироваться на многопроцессорные комплексы, оставаясь, таким образом, в XX веке.

СПИСОК ЛИТЕРАТУРЫ

1. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. Москва: Мир, 1985. 512 с.
2. Shylo V.P. The method of global equilibrium search. *Cybernetics and Systems Analysis*. 1999. Vol. 35, N 1. P. 68–74.
3. Burkard R.E. Quadratic assignment problems. *European J. of Oper. Res*. 1984. Vol. 15. P. 283–289.
4. Шило В.П., Шило О.В., Рощин В.А. Метод глобального равновесного поиска решения задачи о максимальном взвешенном разрезе графа. *Кибернетика и системный анализ*. 2012. № 4. С. 101–105.
5. Шило В.П., Шило О.В. Решение задачи булева квадратичного программирования без ограничений методом глобального равновесного поиска. *Кибернетика и системный анализ*. 2011. № 6. С. 68–78.
6. Pardalos P.M., Shylo O.V. An algorithm for the Job shop scheduling problem based on Global Equilibrium Search Techniques. *Computational Management Science*. 2006. Vol. 3, N 4. P. 331–348.
7. Benlic U., Hao J.K. Memetic search for the quadratic assignment problem. *Expert Systems with Applications.* 2015. Vol. 42, N 1. P. 584–595.
8. James T., Rego C., Glover F. A cooperative parallel tabu search algorithm for the quadratic assignment problem. *European J. of Oper. Res.* 2009. Vol. 195, N 3. P. 810–826.
9. Misevicius A., Kilda B. Iterated tabu search: An improvement to standard tabu search. *Information Technology and Control*. 2006. Vol. 35, N 3. P. 187–197.
10. Misevicius A. An improved hybrid genetic algorithm: New results for the quadratic assignment problem. *Knowledge Based Systems*. 2004. Vol. 17, N 2–4. P. 65–73.

11. Benlic U., Hao J.K. Breakout local search for the quadratic assignment problem. *Applied Math. and Computation.* 2013. Vol. 219, N 9. P. 4800–4815.
12. Stutzle T. Iterated local search for the quadratic assignment problem. *European J. of Oper. Res*. 2006. Vol. 174, N 3. P. 1519–1539.
13. Glover F., Laguna M. Tabu search. Boston: Kluwer Academic Publishers, 1997. 383 p.
14. Shylo V.P., Glover F., Sergienko I.V. Teams of global equilibrium search algorithms for solving the weighted maximum cut problem in parallel. *Cybernetics and Systems Analysis*. 2015. Vol. 51, N 1. P. 16–24.
15. Shylo V.P., Shylo O.V. Algorithm portfolios and teams in parallel optimization. *Optimization Methods and Applications*. Ed. by S. Butenko, P.M. Pardalos, V. Shylo. New York: Springer, 2017. P. 453–465.

Надійшла до редакції 08.08.2017

І.В. Сергієнко, В.П. Шило
ТЕХНОЛОГІЯ ЯДРА ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧ ДИСКРЕТНОЇ ОПТИМІЗАЦІЇ

Анотація. Запропоновано нову технологію для розв'язання задач дискретної оптимізації, яка формує «ядро» розв'язку, що дозволяє ефективно випадково збурювати цей розв'язок у схемах ітерування. Проведено порівняльне дослідження двох версій нового алгоритму розв'язання квадратичної задачі про призначення (з технологією виділення ядра і без неї) з сучасними алгоритмами. Воно показало перспективність використання цієї технології як за швидкодією, так і за можливістю отримання кращих розв'язків. Технологію ядра легко інкорпорувати в наявні алгоритми.

Ключові слова: технологія ядра, квадратична задача про призначення QAP, задача про максимальний зважений розріз графа WMaxCut, квадратична задача булевого програмування UBQP, задача про комівояжера TSP, задача складання розкладів JSP, обчислювальний експеримент, порівняльне дослідження алгоритмів.

I.V. Sergienko, V.P. Shylo
KERNEL TECHNOLOGY TO SOLVE DISCRETE OPTIMIZATION PROBLEMS

Abstract. A kernel technology is proposed for a wide class of discrete optimization problems. Based on the notion of kernel, the technology implements stochastic perturbations for the iterative algorithmic schemes. Computational study of the proposed technology for the quadratic assignement problem demonstrated efficiency of this technology in terms of speed and solution quality. The kernel method can be easily incorporated into the available algorithms.

Keywords: kernel technology, QAP, WMaxCut, UBQP, TSP, JSP, computing experiment, a comparative study of algorithms.

Сергиенко Иван Васильевич,
академик НАН Украины, профессор, директор Института кибернетики им. В.М. Глушкова НАН Украины, Киев, e-mail: aik@public.icyb.kiev.ua.

Шило Владимир Петрович,
доктор физ.-мат. наук, профессор, ведущий научный сотрудник Института кибернетики им. В.М. Глушкова НАН Украины, Киев, e-mail: v.shylo@gmail.com.