ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Сентябрь—октябрь 2012. Том 19, № 5. С. 21–34

УДК 519.8

ОЦЕНКИ ВРЕМЕНИ РАБОТЫ АЛГОРИТМОВ ЛОКАЛЬНОГО СПУСКА ДЛЯ ЗАДАЧИ ПОСТРОЕНИЯ РАСПИСАНИЙ НА ПАРАЛЛЕЛЬНЫХ МАШИНАХ *)

Ю. Ю. Великанова

Аннотация. Изучаются свойства алгоритмов локального спуска с окрестностями квадратичной мощности для NP-трудной задачи теории расписаний $P||C_{\max}$. Получены новые верхние и нижние оценки на время работы алгоритмов локального спуска с заданным направлением выбора соседнего решения.

Ключевые слова: алгоритм локального спуска, окрестность, время работы алгоритма, верхняя и нижняя границы.

Введение

Рассматривается известная NP-трудная задача теории расписаний, в которой заданы множества работ $\mathcal{I} = \{1, \dots, n\}$ и параллельных идентичных машин $M = \{M_1 \dots, M_m\}$. Для работы i определена её длительность p_i, $i = 1, \dots, n$. Требуется распределить работы по машинам так, чтобы минимизировать время завершения всех работ. Согласно принятой в [6] системе обозначений задач теории расписаний данная задача обозначается через $P||C_{\max}$.

В настоящее время для решения NP-трудных задач широко применяются методы локального поиска. Многие из них используют алгоритмы нахождения локальных оптимумов для полиномиально проверяемых окрестностей [8]. Среди таких алгоритмов чаще всего используются алгоритмы локального спуска. Приближённые алгоритмы (алгоритмы имитации отжига [10], поиск с запретами [4, 5] и др.) хорошо работают на практике, но пока недостаточно изучены теоретически.

С теоретической точки зрения интересны два важных аспекта: отклонение полученного локального оптимума от глобального и сложность его нахождения, т. е. как быстро можно найти локальный оптимум [11]. Для задачи $P \mid\mid C_{\max}$ вопрос о сложности нахождения локального минимума

*)Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проект 06–01–00075).

исследовался в [1, 2]. Показано, что для окрестности линейной мощности алгоритму локального спуска требуется $O(n^2)$ шагов. В [7] построен пример, доказывающий точность этой оценки. В данной статье исследуется аналогичный вопрос для более широких окрестностей с квадратичными мощностями. Доказано, что алгоритму локального спуска с заданным направлением выбора соседнего решения требуется полиномиальное число шагов.

1. Основные определения и обозначения

Допустимым решением задачи называется разбиение $I = (I_1, \dots, I_m)$ множества работ $\mathcal{I}$ на m непересекающихся подмножеств, где I_l — множество работ, выполняющихся на машине l, $l = 1, \dots, m$. Обозначим загруженность l-й машины через $S_l = \sum\limits_{i \in I_l} p_i$. Целевой функцией является длина расписания $c_{\max}(I) = \max\{S_1, \dots, S_m\}$. Подмножество работ, на котором достигается максимум, обозначим через $I_{\max}$. Величина $\triangle_l = c_{\max} - S_l$ задаёт простой машины M_l. Машину M_l назовём *критической*, если $S_l = c_{\max}(I)$.

Определение 1. *Окрестностью* Swap *допустимого решения* называется множество решений, которое получается из данного перестановкой двух работ, расположенных на разных машинах.

Определение 2. *Окрестностью* Flip *допустимого решения* называется множество решений, которое получается из данного перемещением одной работы на другую машину.

Определение 3. *Окрестность* Swap∪Flip — объединение окрестностей Swap и Flip. При перестановке работ или перемещении работы будем говорить, что *применён оператор* Swap или Flip соответственно.

Определение 4. Решение называется Swap-*оптимальным*, если оно является локальным оптимумом для окрестности Swap.

Определение 5. Решение называется Swap ∪ Flip-*оптимальным*, если оно является локальным оптимумом для окрестности Swap ∪ Flip.

Алгоритм локального спуска

Шаг 1. Выбирается допустимое решение I.

Шаг 2. Повторяются шаги 2.1 и 2.2, пока I не станет локальным минимумом.

Шаг 2.1. Находится решение I' в окрестности решения I такое, что $c_{\max}(I') < c_{\max}(I)$.

Шаг 2.2. Полагается $I := I'$.

Шаг 3. Выдаётся локальный минимум.

Выполнение шагов 2.1 и 2.2 будем называть *итерацией*.

2. Нахождение локального минимума

Сначала рассмотрим задачу на двух машинах, а потом полученный результат обобщим для произвольного числа машин.

2.1. Задача $P2 \mid\mid C_{\max}$. Пусть (I_1, I_2) — допустимое решение. Обозначим подмножество работ, которое определяет длину расписания, через $I_{\max}$, а через $I_{\min}$ — другое подмножество. Если машины имеют одинаковые загруженности, то решение оптимально. Без ограничения общности будем считать, что загруженности машин разные.

Через $S_{\max}$ и $S_{\min}$ будем обозначать загруженности машин. Тогда простой машины $I_{\min}$ равен $\triangle = S_{\max} - S_{\min}$. Сформулируем критерий локальной оптимальности решения (I_1, I_2) для окрестностей Swap и Swap$\cup$Flip.

Утверждение 1. *Решение (I_1, I_2) является* Swap-*оптимальным тогда и только тогда, когда $p_i - p_j \geqslant \triangle$ для любой пары таких работ i, j, что $p_i > p_j$ при $i \in I_{\max}$ и $j \in I_{\min}$.*

Доказательство. Необходимость. Пусть (I_1, I_2) — Swap-оптимальное решение. Тогда применение оператора Swap к любой паре работ не приводит к уменьшению длины расписания. Пусть $i \in I_{\max}$ и $j \in I_{\min}$. Применим к этой паре работ оператор Swap. Тогда либо $S_{\min} + p_i - p_j \geqslant S_{\max}$, либо $S_{\max} - p_i + p_j \geqslant S_{\max}$. Отсюда либо $p_i - p_j \geqslant S_{\max} - S_{\min} = \triangle$, либо $p_j \geqslant p_i$.

Достаточность. Пусть $p_i - p_j \geqslant \triangle$ для любых работ i, j таких, что $p_i > p_j$, где $i \in I_{\max}$, $j \in I_{\min}$. Покажем, что решение (I_1, I_2) является локальным минимумом. Для этого достаточно проверить пары работ, у которых работа с большей длительностью принадлежит подмножеству $I_{\max}$. Пусть $i \in I_{\max}$, $j \in I_{\min}$ и $p_i > p_j$. По условию теоремы $p_i - p_j \geqslant \triangle$. Применим к этой паре оператор Swap. Получим новое разбиение работ, в котором одно из подмножеств имеет длину $S_{\min} + p_i - p_j \geqslant S_{\min} + \triangle = S_{\max}$, т. е. полученное расписание не лучше по длине. Утверждение 1 доказано.

Утверждение 2. *Решение (I_1, I_2) является* Swap$\cup$Flip-*оптимальным тогда и только тогда, когда $p_i - p_j \geqslant \triangle$ для любых работ i, j таких, что $p_i > p_j$, где $i \in I_{\max}$, $j \in I_{\min}$, и $p_{i'} \geqslant \triangle$ для любого $i' \in I_{\max}$.*

Доказательство проводится аналогично.

Рассмотрим все пары работ (i, j), для которых разность длительностей $p_i - p_j$ положительна. Отсортируем эти пары в порядке невозрастания разности $p_i - p_j$. Получим список пар работ $\Im = ((i_1, j_1), \ldots, (i_h, j_h))$, в котором $p_{i_{(k-1)}} - p_{j_{(k-1)}} \geqslant p_{i_k} - p_{j_k}$ для любого k и $h \leqslant n(n-1)/2$.

На шаге 2.1 алгоритма локального спуска в окрестности решения I может оказаться несколько решений с меньшим значением целевой функции. Выбор решения I' может происходить по разным правилам перехода. Будем пользоваться следующим: берём первую пару из списка $\Im$, для которой выполняется неравенство

$$p_i - p_j < \triangle, \quad i \in I_{\max},\ j \notin I_{\max}, \tag{$*$}$$

и применяем к ней оператор Swap.

Аналогичное правило перехода будем использовать для алгоритма локального спуска с окрестностью Swap∪Flip. Каждую работу i будем рассматривать как пару $(i, 0)$, где 0 — фиктивная работа с длительностью $p_0 = 0$. В этом случае появляется ещё n пар, где одна из работ фиктивная.

Из утверждений 1 и 2 следует, что только пары работ, удовлетворяющие условию $(*)$, могут привести к уменьшению значения целевой функции. Далее будем использовать алгоритм локального спуска только с этим правилом выбора работ.

Применим оператор Swap к выбранной паре работ i, j. Возможны два случая.

Случай 1: $p_i - p_j \leqslant \triangle/2$. Тогда для нового решения $I' = (I'_1, I'_2)$ имеем

$$S'_{\min} = S_{\min} + p_i - p_j, \quad S'_{\max} = S_{\max} - p_i + p_j,$$
$$\triangle' = S'_{\max} - S'_{\min} = \triangle - 2(p_i - p_j).$$

В силу того, что $p_i - p_j \leqslant \triangle/2$ и $p_i - p_j > 0$, получаем $\triangle' < \triangle$. Машина $M_{\max}$ остаётся критической.

Случай 2: $p_i - p_j > \triangle/2$. Тогда

$$S'_{\min} = S_{\min} + p_i - p_j, \quad S'_{\max} = S_{\max} - p_i + p_j,$$

$$\triangle' = S'_{\min} - S'_{\max} = 2(p_i - p_j) - \triangle = (p_i - p_j) - (\triangle - (p_i - p_j)) < p_i - p_j < \triangle.$$

Значит, критической станет другая машина и $\triangle' < \triangle$. В этом случае получаем, что $p_i - p_j > \triangle'$. Так как $\triangle'$ при дальнейшей работе алгоритма локального спуска уменьшается, согласно правилу перехода пара (i, j) на следующих шагах алгоритма не будет выбрана. Далее эти замечания будут использоваться для доказательства теорем.

Теорема 1. *Для задачи $P2 \mid\mid C_{\max}$ алгоритм локального спуска с окрестностью* Swap *и заданным правилом перехода достигает локального минимума с любого стартового решения за $O(n^2)$ итераций.*

Доказательство. Обозначим простой машины до итерации τ через $\triangle(\widetilde{\tau})$, а после — $\triangle(\tau)$.

Как замечено ранее, если к паре работ применяется оператор Swap и возникает случай 2, то в дальнейшей работе алгоритма она не участвует. При этом можно отметить, что все пары, у которых разность длительностей больше, также в работе алгоритма участвовать не будут.

Докажем, что если к некоторой паре работ (i_k, j_k) применяется оператор Swap и имеет место случай 1, то в дальнейшем оператор Swap может применяться только к парам, у которых разность длительностей меньше, чем у пары (i_k, j_k).

От противного. Пусть на итерации τ_1 оператор Swap применялся к паре (i_k, j_k), $i_k \in M_{l_1}$, $j_k \in M_{l_2}$, и имел место случай 1. Машина M_{l_1} осталась критической. После этой итерации $\triangle(\tau_1) = \triangle(\widetilde{\tau}_1) - 2(p_{i_k} - p_{j_k})$. Пусть пара работ $(i_{k'}, j_{k'})$ такая, что $p_{i_{k'}} - p_{j_{k'}} \geqslant p_{i_k} - p_{j_k}$, — первая пара с не меньшей разностью длительностей, к которой применяется оператор Swap после итерации τ_1. Пусть это происходит на итерации τ_2, $\tau_1 < \tau_2$. Заметим, что между итерациями τ_1, τ_2 не было итераций, на которых возникал случай 2, иначе было бы $p_{i_{k'}} - p_{j_{k'}} \geqslant \triangle(\widetilde{\tau}_2)$.

Пусть пара работ (i_z, j_z) такая, что $p_{i_k} - p_{j_k} > p_{i_z} - p_{j_z}$, — последняя пара, к которой применялся оператор Swap на итерации $\tau_2 - 1$. Поскольку к паре $(i_{k'}, j_{k'})$ не применялся оператор Swap на предыдущей итерации, обе работы лежали на одной машине и либо $i_{k'} = j_z$, либо $j_{k'} = i_z$.

Пусть $i_{k'} = j_z$. Тогда $p_{i_z} > p_{j_z} = p_{i_{k'}} > p_{j_{k'}}$. Так как по предположению пара $(i_{k'}, j_{k'})$ будет меняться на следующей итерации, $\triangle(\widetilde{\tau}_2) = \triangle(\widetilde{\tau_2 - 1}) - 2(p_{i_z} - p_{j_z}) > p_{i_{k'}} - p_{j_{k'}}$. Согласно правилу выбора получаем $p_{i_z} - p_{j_{k'}} \geqslant \triangle(\widetilde{\tau_2 - 1})$. Имеем

$$
\begin{aligned}
p_{i_{k'}} - p_{j_{k'}} < \triangle(\widetilde{\tau}_2) = \triangle(\widetilde{\tau_2 - 1}) - 2(p_{i_z} - p_{j_z}) \\
\leqslant p_{i_z} - p_{j_{k'}} - 2(p_{i_z} - p_{j_z}) = -p_{i_z} - p_{j_{k'}} + 2p_{i_{k'}},
\end{aligned}
$$

откуда $0 > p_{i_z} - p_{i_{k'}}$; противоречие.

Пусть теперь $j_{k'} = i_z$. Тогда $p_{i_{k'}} > p_{j_{k'}} = p_{i_z} > p_{j_z}$. Верны неравенства $p_{i_z} - p_{j_z} < p_{i_{k'}} - p_{j_{k'}} = p_{i_{k'}} - p_{i_z} < p_{i_{k'}} - p_{j_z}$. Согласно правилу выбора получаем $p_{i_{k'}} - p_{j_z} \geqslant \triangle(\widetilde{\tau_2 - 1})$. Тогда

$$p_{i_{k'}} - p_{j_z} - 2(p_{i_z} - p_{j_z}) \geqslant \triangle(\widetilde{\tau_2 - 1}) - 2(p_{i_z} - p_{j_z}) \\ = \triangle(\widetilde{\tau}_2) > p_{i_{k'}} - p_{j_{k'}} = p_{i_{k'}} - p_{i_z},$$

откуда $0 > p_{i_z} - p_{j_z}$; противоречие. Тем самым доказано, что после случая 1 оператор Swap применяется только к парам с меньшей разностью длительностей.

Учитывая, что после случая 2 оператор Swap также может применяться только к парам с меньшей разностью длительностей, заключаем, что к каждой паре работ оператор Swap применяется не более одного раза. Значит, алгоритм локального спуска с заданным правилом выбора может делать не более $O(n^2)$ шагов. Теорема 1 доказана.

Теорема 2. *Для задачи $P2 \mid\mid C_{\max}$ алгоритм локального спуска с окрестностью* Swap$\cup$Flip *и заданным правилом перехода достигает локального минимума с любого стартового решения за $O(n^2)$ итераций.*

Доказательство повторяет доказательство теоремы 1. Как отмечалось, оператор Flip можно рассматривать как оператор Swap, где одна из меняющихся работ имеет длительность 0.

Заметим, что случаи 1 и 2 возникают и тогда, когда одна из работ фиктивна. Аналогично показывается, что если возник случай 2, то в дальнейшей работе эта пара не участвует, независимо была ли одна из работ фиктивной. Доказательство того, что после случая 1 могут меняться пары с меньшей разностью длительностей, аналогично.

Всего возможно $O(n^2 + n)$ итераций, так как пар стало на n больше. Следовательно, число итераций равно $O(n^2)$. Теорема 2 доказана.

2.2. Задача $P \mid\mid C_{\max}$. Пусть m — число машин, $I = (I_1, \ldots, I_m)$ — допустимое решение задачи. Следующие утверждения аналогичны утверждениям 1 и 2.

Утверждение 3. *Решение $(I_1, \ldots, I_m)$ является* Swap-*оптимальным тогда и только тогда, когда выполняется одно из следующих условий:*

(i) $p_i - p_j \geqslant \triangle_k$ *для любой пары работ (i, j), удовлетворяющей неравенству $p_i > p_j$, где $i \in I_{\max}$, $j \in I_k$, и $S_k < S_{\max}$;*

(ii) *существует не менее двух критических машин.*

Утверждение 4. *Решение $(I_1, \ldots, I_m)$* Swap$\cup$Flip-*оптимально тогда и только тогда, когда выполняется одно из следующих условий:*

(i) $p_i - p_j \geqslant \triangle_k$ *для любой пары работ* (i, j), *удовлетворяющей неравенству* $p_i > p_j$, $i \in I_{\max}$, $j \in I_k$, *при* $S_k < S_{\max}$ *и* $p_{i'} \geqslant \triangle_l$ *при* $S_l < S_{\max}$ *для любой работы* $i' \in I_{\max}$;

(ii) *существует не менее двух критических машин.*

Как и для задачи на двух машинах, рассмотрим упорядоченный список пар работ $\Im = ((i_1, j_1), \ldots, (i_h, j_h))$, $h \leqslant n(n-1)/2$, в котором $p_{i_{(k-1)}} - p_{j_{(k-1)}} \geqslant p_{i_k} - p_{j_k}$ и $p_{i_k} - p_{j_k} > 0$ для любого k. Правило перехода для алгоритма локального спуска следующее: берём первую пару из списка $\Im$, для которой выполняется неравенство $p_i - p_j < \triangle_l$, где $i \in I_{\max}$, $j \in I_l$, и применяем к этой паре оператор Swap.

Через τ_r будем обозначать номер итерации. Решение непосредственно до итерации τ_r обозначим через $I(\widetilde{\tau}_r) = (I_1(\widetilde{\tau}_r), \ldots, I_m(\widetilde{\tau}_r))$, решение после итерации τ_r — через $I(\tau_r) = (I_1(\tau_r), \ldots, I_m(\tau_r))$. Простой машины M_l перед итерацией τ_r обозначим через $\triangle_l(\widetilde{\tau}_r) = S_{\max}(\widetilde{\tau}_r) - S_l(\widetilde{\tau}_r)$.

Замечание 1. Если в процессе работы алгоритма с итерации τ_1 до итерации τ_2 машина M_l не была критической, то величина $\triangle_l$ уменьшится.

Лемма 1. *Пусть* $\triangle_l(\widetilde{\tau}_1) < \triangle_l(\widetilde{\tau}_2)$ *при* $\tau_1 < \tau_2$ *для некоторой машины* M_l, *т. е. простой машины увеличился. Тогда между итерациями* τ_1 *и* τ_2 *имеется итерация, на которой машина* M_l *критическая.*

Доказательство. По определению

$$\triangle_l(\widetilde{\tau}_1) = S_{\max}(\widetilde{\tau}_1) - S_l(\widetilde{\tau}_1), \quad \triangle_l(\widetilde{\tau}_2) = S_{\max}(\widetilde{\tau}_2) - S_l(\widetilde{\tau}_2).$$

Так как в процессе работы алгоритма значение целевой функции уменьшается, $S_{\max}(\widetilde{\tau}_1) > S_{\max}(\widetilde{\tau}_2)$. Поэтому

$$S_{\max}(\widetilde{\tau}_2) - S_l(\widetilde{\tau}_1) < S_{\max}(\widetilde{\tau}_1) - S_l(\widetilde{\tau}_1) < S_{\max}(\widetilde{\tau}_2) - S_l(\widetilde{\tau}_2).$$

Следовательно, $S_l(\widetilde{\tau}_1) > S_l(\widetilde{\tau}_2)$. Поскольку загруженность машины M_l уменьшилась, между итерациями τ_1 и τ_2 существует итерация, на которой машина M_l является критической. Лемма 1 доказана.

Лемма 2. *Пусть машина* M_l *в начале итерации* τ_1 *критическая и на этой итерации оператор* Swap *применяется к паре работ* (i, j), $i \in I_l(\widetilde{\tau}_1)$, $j \in I_z(\widetilde{\tau}_1)$. *Тогда* $p_i - p_j \geqslant \triangle_l(\tau_1)$.

Доказательство. Перед итерацией τ_1 имеем $i \in I_l(\widetilde{\tau}_1)$, $j \in I_z(\widetilde{\tau}_1)$, $z \neq l$, $p_i > p_j$, $p_i - p_j < \triangle_z(\widetilde{\tau}_1)$. После τ_1 получаем $S_l(\tau_1) = S_l(\widetilde{\tau}_1) + p_j - p_i$. Следовательно, $p_i - p_j = S_l(\widetilde{\tau}_1) - S_l(\tau_1) = S_{\max}(\widetilde{\tau}_1) - S_l(\tau_1)$ и

$$\triangle_l(\tau_1) = S_{\max}(\tau_1) - S_l(\tau_1).$$

Так как $S_{\max}(\tau_1) \leqslant S_{\max}(\widetilde{\tau}_1)$, имеем

$$\triangle_l(\tau_1) \leqslant S_{\max}(\widetilde{\tau}_1) - S_l(\tau_1) = p_i - p_j$$

и $p_i - p_j \geqslant \triangle_l(\tau_1)$. Лемма 2 доказана.

Лемма 3. *Пусть для некоторой пары работ (i, j) выполняются неравенства $p_i - p_j \geqslant \triangle_l(\tau_1)$ и $p_i - p_j < \triangle_l(\tau_2)$, где $\tau_1 < \tau_2$. Тогда существует итерация τ_3, $\tau_1 < \tau_3 < \tau_2$, на которой машина M_l является критической и оператор* Swap *применяется к паре работ (i_k, j_k) такой, что $p_{i_k} - p_{j_k} > p_i - p_j$, $i_k \in I_l(\widetilde{\tau}_3)$, $j_k \in I_z(\widetilde{\tau}_3)$.*

Доказательство. По условию леммы $\triangle_l(\tau_1) \leqslant p_i - p_j < \triangle_l(\tau_2)$. Следовательно, простой машины M_l увеличился. По лемме 1 между итерациями τ_1 и τ_2 машина M_l критическая. Пусть итерация τ_3 — последняя итерация до τ_2, на которой машина M_l является критической, т. е. простой на этой машине после итерации τ_3 и до итерации τ_2 мог только уменьшиться. Пусть на итерации τ_3 оператор Swap применяется к паре работ (i_k, j_k). По лемме 2 имеем $p_{i_k} - p_{j_k} \geqslant \triangle_l(\tau_3) \geqslant \triangle_l(\tau_2) > p_i - p_j$. Лемма 3 доказана.

Лемма 4. *Во время работы алгоритма локального спуска оператор* Swap *может применяться к первой паре работ $(i_1, j_1) \in \Im$ не более одного раза.*

Доказательство. Пусть на итерации τ_1 к паре работ (i_1, j_1) применяется оператор Swap, при этом работы i_1, j_1 выполняются на машинах M_{l_1} и M_{l_2} соответственно. По лемме 2 имеем $p_{i_1} - p_{j_1} \geqslant \triangle_{l_1}(\tau_1)$. По лемме 3 к паре (i_1, j_1) не может применяться оператор Swap, когда $j_1 \in M_{l_1}$. Тогда для того чтобы оператор Swap применялся к паре (i_1, j_1) ещё раз, необходимо, чтобы работа j_1 переместилась с машины M_{l_1}. Так как работа j_1 — самая маленькая из всех работ, она может поменяться с какой-то работой i_k такой, что $p_{i_k} > p_{j_1}$. Пусть это происходит на итерации τ_2 и $i_k \in M_{l_3}$. Имеем $\triangle_{l_3} \leqslant p_{i_k} - p_{j_1} \leqslant p_{i_1} - p_{j_1}$. Это означает, что к паре (i_1, j_1) не может применяться оператор Swap. Таким образом, пара работ (i_1, j_1) может участвовать не более чем в одной итерации алгоритма. Лемма 4 доказана.

Лемма 5. *Пусть оператор* Swap *применяется к паре работ (i_k, j_k) на итерациях τ_1 и τ_2, причём оба раза работа j_k выполняется на машине M_l. Тогда между этими итерациями машина M_l критическая.*

Доказательство. После итерации τ_1 и до итерации τ_2 работа j_k должна вернуться на машину M_l. Пусть это происходит на итерации τ_3.

Вернуться j_k может, поменявшись либо с работой $i_z \in I_l(\widetilde{\tau}_3)$ такой, что $p_{i_z} > p_{j_k}$, либо с работой $j_z \in I_l(\widetilde{\tau}_3)$ такой, что $p_{j_k} > p_{j_z}$. В первом случае машина M_l является критической на итерации τ_3.

Рассмотрим случай, когда работа j_k на итерации τ_3 меняется с работой $j_z \in I_l(\widetilde{\tau}_3)$ такой, что $p_{j_k} > p_{j_z}$. Пусть между итерациями τ_1 и τ_2 машина M_l не критическая. Тогда $p_{i_k} - p_{j_z} > p_{i_k} - p_{j_k}$. Согласно правилу выбора работ на итерации τ_1 имеем либо $j_z \in I_l,\ p_{i_k} - p_{j_z} \geqslant \triangle_l(\widetilde{\tau}_1)$, либо $j_z \notin I_l$.

Пусть $j_z \in I_l,\ p_{i_k} - p_{j_z} \geqslant \triangle_l(\widetilde{\tau}_1)$ и машина M_l не является критической между итерациями τ_1 и τ_2. Тогда

$$S_l(\tau_1) \leqslant S_l(\tau_3) \leqslant S_l(\tau_2), \quad \tau_1 < \tau_3 < \tau_2. \qquad (**)$$

Кроме того, из определения итераций τ_1 и τ_3 получаем

$$S_l(\tau_1) = S_l(\widetilde{\tau}_1) + p_{i_k} - p_{j_k}, \quad S_l(\tau_3) = S_l(\widetilde{\tau}_3) + p_{j_k} - p_{j_z}.$$

Используя эти соотношения, имеем

$$\begin{aligned} S_l(\widetilde{\tau}_2) \geqslant S_l(\tau_3) = S_l(\widetilde{\tau}_3) + p_{j_k} - p_{j_z} \geqslant S_l(\tau_1) + p_{j_k} - p_{j_z} \\ = S_l(\widetilde{\tau}_1) + p_{i_k} - p_{j_k} + p_{j_k} - p_{j_z} = S_l(\widetilde{\tau}_1) + p_{i_k} - p_{j_z}. \end{aligned}$$

На итерации τ_2 меняется пара (i_k, j_k), поэтому $p_{i_k} - p_{j_k} < \triangle_l(\widetilde{\tau}_2)$. Отсюда $p_{i_k} - p_{j_k} < S_{\max}(\widetilde{\tau}_2) - S_l(\widetilde{\tau}_2) \leqslant S_{\max}(\widetilde{\tau}_1) - S_l(\widetilde{\tau}_2)$. Используя $(**)$, получаем

$$p_{i_k} - p_{j_k} < S_{\max}(\widetilde{\tau}_1) - S_l(\widetilde{\tau}_1) - p_{i_k} + p_{j_z} = \triangle_l(\widetilde{\tau}_1) - p_{i_k} + p_{j_z}.$$

По предположению $p_{i_k} - p_{j_z} \geqslant \triangle_l(\widetilde{\tau}_1)$, поэтому $p_{i_k} - p_{j_k} < 0$; противоречие.

Пусть $j_z \notin I_l(\widetilde{\tau}_1)$. Следовательно, эта работа появилась позже, поменявшись с другой работой j_w. Так как машина M_l не критическая, $p_{j_w} < p_{j_z} < p_{j_k} < p_{i_k}$. Применив приведённые рассуждения к работе j_w, получим, что $j_w \notin I_l(\widetilde{\tau}_1)$ на итерации τ_1. Поскольку множество работ конечно, получим противоречие с предположением, что машина M_l не является критической. Лемма 5 доказана.

Для каждой пары $(i_k, j_k) \in \Im$ посчитаем максимальное число итераций, в которых пара (i_k, j_k) может участвовать. Для этого введём понятия первичных и повторных итераций. Для пары (i_k, j_k) и машины M_l *первичной итерацией* называется первая итерация, на которой оператор Swap применяется к паре (i_k, j_k), где работа j_k выполняется на машине M_l. Для любой машины M_l и пары работ (i_k, j_k) может быть не более одной первичной итерации. Все итерации, не являющиеся первичными для какой-нибудь машины, называются *повторными*. Далее для подсчёта количества итераций, в которых участвует пара (i_k, j_k), не имеет значения, на каких машинах выполняются работы, а важно, является ли итерация первичной или нет. Так как машин m, для любой пары $(i_k, j_k) \in \Im$ может быть не более m первичных итераций.

Теорема 3. *Для задачи $P \,||\, C_{\max}$ алгоритм локального спуска с окрестностью* Swap *и заданным правилом перехода достигает локального минимума с любого стартового решения за $O(n^4 m)$ итераций, где m — число машин.*

Доказательство. Обозначим через $N(i_k, j_k)$ максимальное число применений оператора Swap к паре (i_k, j_k). Покажем, что

$$N(i_k, j_k) \leqslant (k-1)m + 1, \quad 2 \leqslant k \leqslant h.$$

Пусть оператор Swap применён дважды к паре (i_k, j_k), когда $j_k \in M_l$. По лемме 5 между повторными итерациями машина M_l должна быть критической, т. е. оператор Swap применяется к некоторой паре работ (i_z, j_z), $p_{i_z} > p_{j_z}$, $i_z \in M_l$, на итерации τ. Следовательно, по лемме 2 имеем, что $p_{i_z} - p_{j_z} \geqslant \triangle_l(\tau)$. Если $p_{i_z} - p_{j_z} \leqslant p_{i_k} - p_{j_k}$, то $p_{i_k} - p_{j_k} \geqslant \triangle_l(\tau)$. Значит, по лемме 3 пара (i_k, j_k) сможет поменяться только после того, как оператор Swap будет применён к паре (i_t, j_t), $t \leqslant k$. Если $p_{i_z} - p_{j_z} > p_{i_k} - p_{j_k}$, то $z < k$. Таким образом, между повторными применениями оператора Swap к паре (i_k, j_k), $j_k \in M_l$, оператор Swap должен быть применён к паре (i_t, j_t), $t < k$.

Посчитаем число повторных итераций пары (i_k, j_k). Пусть на итерации τ_1 пара (i_k, j_k) делает повторную итерацию. Перед каждой такой итерацией должна быть итерация, на которой оператор Swap применялся к паре с большей разностью длительностей. Пусть это пара (i_t, j_t), $t < k$, и итерация τ_2. Если для пары (i_t, j_t) это тоже повторная итерация, то до итерации τ_2 была итерация τ_3, на которой оператор Swap применялся к паре работ (i_z, j_z), $z < t$. Если она опять совершает повторную итерацию, то существует ещё пара, и т. д., пока не дойдём до пары с первичной итерацией. Такая цепочка не может быть бесконечной, поскольку каждый раз номер пары должен быть меньше номера предыдущей пары. Тем самым каждой повторной итерации (i_k, j_k) соответствует первичная итерация пары (i_s, j_s) такой, что $s < k$. Всего различных пар может быть $k - 1$, и у каждой пары, за исключением (i_1, j_1), не может быть более m первичных итераций. Следовательно, возможных цепочек может быть не более $m(k-2) + 1$. Значит, повторных итераций пары

(i_k, j_k) не более $m(k-2)+1$. С учётом первичных итераций получаем, что $N(i_k, j_k) \leqslant (k-2)m + 1 + m = (k-1)m + 1$.

Число шагов алгоритма локального спуска можно оценить величиной

$$\sum_{k=1}^{h} N(i_k, j_k) = \sum_{k=2}^{h}((k-1)m+1) + 1 \leqslant \frac{n^2(n-1)^2 m}{8} + n^2 \approx O(n^4 m).$$

Теорема 3 доказана.

Замечание 2. Полученные результаты справедливы и в более общем случае, когда алгоритм локального спуска продолжает работу даже при наличии нескольких критических машин. В этом случае согласно правилу выбора работ алгоритм выбирает пару (i_k, j_k) из списка $\Im$, просматривая i_k по всем критическим машинам. Остановка алгоритма происходит тогда, когда не существует пар (i_k, j_k) со свойством $p_{i_k} > p_{j_k}$, где $i_k \in I_{\max}$, $j_k \in I_l$, $p_{i_k} - p_{j_k} < \triangle_l$.

3. Нижние оценки числа шагов алгоритма локального спуска

Покажем, что оценка, полученная в теореме 1, точна.

Рассмотрим пример задачи на двух машинах, когда алгоритму локального спуска с заданным правилом выбора работ с окрестностью Swap требуется $\Omega(n^2)$ итераций для достижения локального минимума.

Пусть $n \geqslant 16$ — некоторое положительное число, делящееся без остатка на 4, и $\mathcal{B} = n^3$.

Множество работ $\mathcal{I} = \{1, \ldots, n+1\}$ состоит из трёх подмножеств J_b, J_s, J_y. Подмножество работ $J_b = \{b_0, b_1 \ldots, b_{n/2-1}\}$ состоит из $n/2$ работ с большими длительностями, равными $p_{b_i} = \mathcal{B} - in + 2i,\ i = 0, \ldots, n/2-1$. В подмножество $J_s = \{s_1, \ldots, s_{n/2-1}\}$ входят $n/2-1$ работ с маленькими длительностями $p_{s_k} = 2k-1,\ k = 1, \ldots, n/2-1$. Подмножество работ $J_y = \{y_1, y_2\}$ состоит из двух работ. Длительности этих работ равны $p_{y_1} = \mathcal{B},\ p_{y_2} = \frac{n^2}{4} - n + 1$.

По правилу выбора работ на шаге 2.1 алгоритма локального спуска оператор Swap применяется к парам работ согласно их разности длительностей. Упорядочим все пары работ по невозрастанию разностей длительностей. Выпишем первые $(n^2/8 - n/4 + 4)$ пар из этого списка:

$$(b_0, s_1), (y_1, s_1), (b_0, s_2), (y_1, s_2), \ldots, (b_0, s_{n/2-1}), (y_1, s_{n/2-1}), (b_1, s_1), \ldots,$$
$$(b_1, s_{n/2-1}), (b_2, s_1), \ldots, (b_2, s_{n/2-1}), \ldots, (b_{n/4-1}, s_1), (b_{n/4-1}, s_2),$$
$$(b_0, y_2), (y_1, y_2).$$

В дальнейшем будут использоваться только пары работ, приведённые выше.

Начальное решение $I(0) = (I_1(0), I_2(0))$ задано следующим образом:

$$\begin{array}{llllllllll} M_1: & y_1, & b_0, & b_2, & \ldots, & b_{\frac{n}{2}-2}, & s_2, & s_4, & \ldots, & s_{\frac{n}{2}-2}, \\ M_2: & y_2, & b_1, & b_3, & \ldots, & b_{\frac{n}{2}-1}, & s_1, & s_3, & \ldots, & s_{\frac{n}{2}-1}, \end{array}$$

$S_1(0) > S_2(0),\ \triangle(0) = S_1(0) - S_2(0) = \mathcal{B}$.

Согласно правилу выбора работ на первых $n/2 - 1$ итерациях работы алгоритма оператор Swap поочерёдно применяется к парам (b_0, s_1), $(b_o, s_2), \ldots, (b_0, s_{n/2-1})$. Так как $p_{b_0} - p_{s_i} > \triangle(i-1)/2$, при перестановке этих работ возникает случай 2. Критическая машина меняется и

$$\begin{aligned} \triangle(i) = 2(p_{b_0} - p_{s_i}) - \triangle(i-1) = \triangle(i-1) - 2 \\ = 2(n^3 - (2i-1)) - n^3 + 2(i-1) = n^3 - 2i. \end{aligned}$$

После $n/2 - 1$ итераций получим решение $I(n/2-1) = (I_1(n/2-1), I_2(n/2-1))$, которое имеет вид

$$\begin{array}{lllllllllll} M_1: & y_1, & & & b_2, & \ldots, & b_{\frac{n}{2}-2}, & s_1, & s_3, & \ldots, & s_{n/2-1}, \\ M_2: & y_2, & b_0, & b_1, & b_3 & \ldots, & b_{\frac{n}{2}-1}, & s_2, & s_4, & \ldots, & s_{n/2-2}, \end{array}$$

$$S_2(n/2-1) > S_1(n/2-1),$$

$$\triangle(n/2-1) = S_2(n/2-1) - S_1(n/2-1) = \mathcal{B} - n + 2.$$

Для пары (b_1, s_1) имеем $p_{b_1} - p_{s_1} = \mathcal{B} - n + 2 - 1 = \triangle(n/2-1) - 1$. На следующих $n/2-1$ итерациях оператор Swap применяется к следующим из списка парам работ $(b_1, s_1), \ldots, (b_1, s_{n/2-1})$. Каждый раз имеет место случай 2. Критическая машина меняется и

$$\begin{aligned} \triangle(n/2-1+i) = 2(p_{b_1} - p_{s_i}) - \triangle(n/2-1+i-1) \\ = \triangle(n/2-1+i-1) - 2 = n^3 - n + 2 - 2i. \end{aligned}$$

После $(n/2-1)$ итераций имеем решение $I(2(n/2-1)) = (I_1(2(n/2-1)), I_2(2(n/2-1)))$ вида

$$\begin{array}{llllllllll} M_1: & y_1, & b_1, & b_2, & \ldots, & b_{\frac{n}{2}-2}, & s_2, & s_4, & \ldots, & s_{n/2-2}, \\ M_2: & y_2, & b_0, & b_3 & \ldots, & b_{\frac{n}{2}-1}, & s_1, & s_3, & \ldots & s_{n/2-1}, \end{array}$$

$$S_1(2(n/2-1)) > S_2(2(n/2-1)),$$

$$\triangle(2(n/2-1)) = S_1(2(n/2-1)) - S_2(2(n/2-1)) = \mathcal{B} - 2n + 4.$$

На следующих $n/2-1$ итерациях согласно правилу выбора работ будут меняться пары $(b_2, s_1), \ldots, (b_2, s_{n/2-1})$. После этих итераций будут аналогичные итерации с парами $(b_3, s_1), \ldots, (b_3, s_{n/2-1})$, и т. д. до пары $(b_{n/4-1}, s_1)$ из упорядоченного списка пар. Таким образом, алгоритм локального спуска после $(n/4-1)(n/2-1) \approx \Omega(n^2)$ шагов получает решение

$$\begin{array}{llllllllll}
M_1: & y_1, & b_1, & b_3, & \ldots, & b_{\frac{n}{4}-3}, & & & b_{\frac{n}{4}}, & \\
 & & & & \ldots, & b_{\frac{n}{2}-2}, & s_1, & s_3, & \ldots, & s_{n/2-1}, \\
M_2: & y_2, & b_0, & b_2, & \ldots, & b_{\frac{n}{4}-4}, & b_{\frac{n}{4}-2}, & b_{\frac{n}{4}-1}, & b_{\frac{n}{4}+1}, & \\
 & & & & \ldots, & b_{\frac{n}{2}-1}, & s_2, & s_4, & \ldots, & s_{n/2-2},
\end{array}$$

$$S_1((n/4-1)(n/2-1)) > S_2((n/4-1)(n/2-1)),$$
$$\triangle((n/4-1)(n/2-1)) = \mathcal{B} - 2(n/4-1)(n/2-1).$$

Данное решение не является локальным минимумом, поэтому алгоритм локального спуска на этом решении не остановится. Тем не менее на рассмотренном примере алгоритму потребуется $\Omega(n^2)$ шагов. Значит, оценка, полученная в теореме 1, точна.

4. Дальнейшие исследования

В статье показано, что алгоритм локального спуска с рассмотренными окрестностями квадратичной мощности и заданными направлениями выбора соседнего решения достигает локального минимума за полиномиальное число шагов. Для автора представляется интересным исследовать другие окрестности. Например, окрестности с экспоненциальными мощностями, для которых поиск лучшего решения среди соседних осуществляется с полиномиальной трудоёмкостью [3], или окрестности, основанные на идее Лина — Кернигана [9]. Для многих задач теории расписаний вопрос о сложности поиска локального минимума остаётся открытым. Поэтому ещё одно направление дальнейших исследований связано с рассмотрением других задач, например, задачи $Pm \mid\mid \sum w_i c_i$, где в качестве целевой функции берётся взвешенная сумма времён окончания работ, или задачи $Pm \mid prec \mid C_{\max}$, в которой работы должны выполняться в заданном порядке.

ЛИТЕРАТУРА

1. **Brucker P., Hurink J., Werner F.** Improving local search heuristics for some scheduling problems. I // Discrete Appl. Math. — 1996. — Vol. 65, N 1–3. — P. 97–122.

2. **Brucker P., Hurink J., Werner F.** Improving local search heuristics for some scheduling problems. II // Discrete Appl. Math. 1997. Vol. 72, N 1–2. P. 47–69.
3. **Brueggemann T.** Efficiency of local search: Thes.. . . doct. phylosophy . — Univ. Twente, Enschede, 2006. — 167 p.
4. **Glover F.** Tabu search, part I // ORSA J. Comput. — 1989. — Vol. 1, N 3. — P. 190–206.
5. **Glover F.** Tabu search, part II // ORSA J. Comput. — 1990. — Vol. 2, N 1. — P. 4–32.
6. **Graham R. L., Lawler E. L., Lenstra J. K., Rinnoy Kan A. H. G.** Optimization and approximation in deterministic sequencing and scheduling // Ann. Discrete Math. — 1979. — Vol. 5. — P. 287–326.
7. **Hurkens C. A. J., Vredeveld T.** Local search for multiprocessor scheduling: how many moves does it take to a local optimum?// Oper. Res. Lett. — 2003. — Vol. 31. — P. 137–141.
8. **Ibaraki T., Nonobe K., Yagiura M.** Metaheuristics: progress as real problem solvers. — Berlin: Springer-Verl., 2005. — 414 p.
9. **Kernighan B. W., Lin S.** An efficient heuristic procedure for partitioning graphs // Bell Syst. Tech. J. — 1970. — Vol. 49. — P. 291–307.
10. **Van Laarhoven P. J. M., Aarts E. H. L.** Simulated annealing: theory and applications. — Reidel, Dordrecht: Kluwer Acad. Publ., 1987. — 204 p.
11. **Yannakakis M.** Computational complexity. Local search in combinatorial optimization. — Chichester: Wiley, 1997. — P. 19–55.

Великанова Юлия Юрьевна,
e-mail: julia.velikanova@gmail.com

Статья поступила
13 августа 2009 г.

Переработанный вариант —
14 марта 2012 г.