Эвристические алгоритмы для задачи составления расписаний мультимодальных работ с ресурсными ограничениями

Ю.В. Захарова, М.Ю. Сахно

Омский филиал Института математики им. С.Л. Соболева СО РАН

Исследование выполнено за счет гранта Российского научного фонда № 22-71-10015-П, https://rscf.ru/project/22-71-10015-П/.

Постановка задачи

- Необходимо составить расписание выполнения множества распараллеливаемых работ $\mathcal{J} = \{1, ..., n\}$ на m ядрах процессора.
- Имеется два ресурса возобновимого типа – шина данных и оперативная память.
- Каждая работа может выполняться в разных модах.
- Задан частичный порядок на множестве работ (может быть пустым).
- Необходимо минимизировать общее время завершения всех работ $C_{\max}$.

Постановка задачи

- Для каждой работы $j \in \mathcal{J}$ задан требуемый процент объема оперативной памяти v_j.
- Для каждой работы $j \in \mathcal{J}$ определено множество возможных мод выполнения $\mathcal{M}_j = \{(size_{ji}, \tilde{b_{ji}}, \tilde{p_{ji}}) \mid i \in I_j\}$, где I_j — множество индексов мод работы j; $size_{ji}$ — количество ядер, необходимых для выполнения работы j в моде i; $\tilde{b_{ji}}$ и $\tilde{p_{ji}}$ – процент потребления шины данных и количество единиц времени, необходимые ей для полного выполнения в идеальных условиях в моде i, соответственно.
- Если при выполнении работы j в выбранной моде i ей выделен меньший процент шины данных, чем $\tilde{b_{ji}}$, то она пропорционально замедляется.

Алгоритм 1 Правило расчета замедления работ $j \in J$ в конфигурации $c \in C$

Шаг 1: Положим процент свободной шины данных $freePercent := 100\%$, текущее количество ядер, среди которых распределяется шина данных, $coresCount := \sum_{j,i \in c} size_{ji}$ и текущее количество работ $jobsCount$ равно количеству работ в конфигурации c.

Шаг 2: Пока $jobsCount$ не равно 0, выполнять:

2.1: Вычислить процент шины данных, который может быть выделен каждому ядру: $percent := freePercent/coresCount$.

2.2: Если в конфигурации c присутствует такая работа j, для которой $b_j \leq percent * size_j$, положить:

$z_{jc} := b_j$,
$freePercent := freePercent - b_j$,
$jobsCount := jobsCount - 1$.
$coresCount := coresCount - size_j$.

2.3: Иначе положить $b_{jc} := percent * size_j$ для каждой оставшейся работы j и $jobsCount := 0$.

Шаг 3: Скорость выполнения каждой работы j в конфигурации c пропорциональна отношению z_{jc}/b_j.

Предыдущие исследования

- **Мультимодальные параллельные работы** (Drozdowski, 2009).
- **Быстрые онлайн-эвристики** (Merkel et al., 2010; Zhuravlev et al., 2010). Работы размещаются на ядрах процессора по принципу **комплементарности**, чтобы одновременно выполнялись задачи с максимально разными требованиями к ресурсам.
- **Расчет замедления**: использование методов **ML** (Zacarias et al., 2019), а также **оффлайн-профилирования** и предсказательных моделей (Fedorova et al., 2007; Snavely et al., 2000; Chandra et al., 2005).
- **Критерии оптимизации**: минимизация суммарной деградации всех работ (Jiang et al., 2008) или общего времени завершения (*makespan*) (Tian et al., 2012).
- Подходы, где **ресурсы назначаются ядрам**, а не работам, при условии, что распределение задач по ядрам и их порядок зафиксированы (Althaus et al., 2018).
- Исследование случаев, когда **допустимы прерывания** и непрерывное перераспределение ресурсов (Shioura et al., 2018).

Нижняя граница

$$LB = \max\left\{P^{\max},\ \frac{\sum_{j\in J} ps_j^{\min}}{m},\ \frac{\sum_{j\in J} pv_j^{\min}}{V},\ \frac{\sum_{j\in J} pb_j^{\min}}{100}\right\},$$

где $ps_j^{\min} = \min_{i\in I_j} p_{ji} size_{ji}$, $pv_j^{\min} = \min_{i\in I_j} p_{ji} V_j$, $pb_j^{\min} = \min_{i\in I_j} p_{ji} b_{ji}$, $j \in J$, значение $P^{\max}$ это длина критического пути в графе частичного порядка с минимальными длительностями работ среди всех мод.

Без частичного порядка

Для задачи составления расписаний с независимыми одномодальными работами можно построить 6-приближенное решение за полиномиальное время.

всегда не менее половины одного из ресурсов:

$$L_r \frac{R}{2} \leq \sum_{j \in J} p_j r_j \leq LB \cdot R.$$

есть интервалы с менее чем половиной всех ресурсов:

$$r' \cdot \Delta + (R - r' + 1) \cdot L'_r \leq LB \cdot R.$$

Приближенные алгоритмы

С частичным порядком

Для задачи составления расписаний одномодальных работ, связанных произвольным частичным порядком, можно построить $\left(\frac{6-3q}{1-q}\right)$-приближенное решение за полиномиальное время, где $q \in (0,1)$ такое, что $size_j \leq qm$, $v_j \leq qV$, $b_j \leq 100q$ for $j \in J$.

декомпозиция на интервалы двух типов:
$L_\sigma = L_1 + L_2$.

интервалы по частичному порядку:
$LB \geq \sum_{l=1}^{k} p_{j_l} = L_1$.

интервалы по дефициту ресурсов:
$L_2 = L_{2m} + L_{2V} + L_{2b}$.

$L_1 + ((1-q)R + 1)\, L_{2r} \leq \sum_{j \in \mathcal{J}} p_j r_j \leq LB$.

Алгоритм 2 Жадный алгоритм GR (идея из [Eremeev & Sakhno, 2020])

Пока присутствуют незавершенные работы, повторять:
1: Выбрать допустмимую конфигурацию работ для выполнения.
2: Вычислить скорости выполнения работ в выбранной конфигурации.
3: Определить, какая из работ завершится первой.

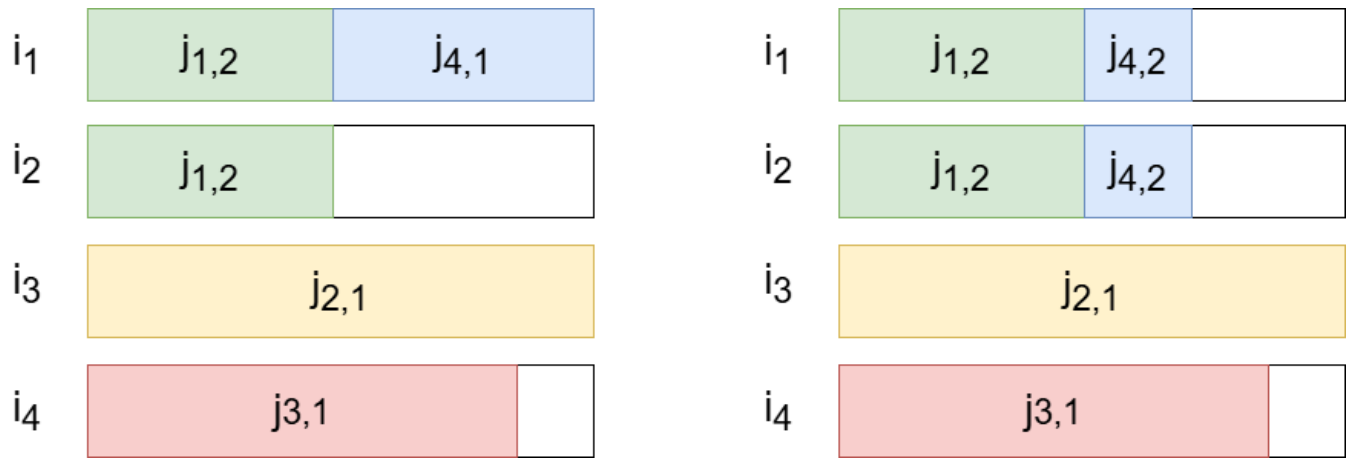
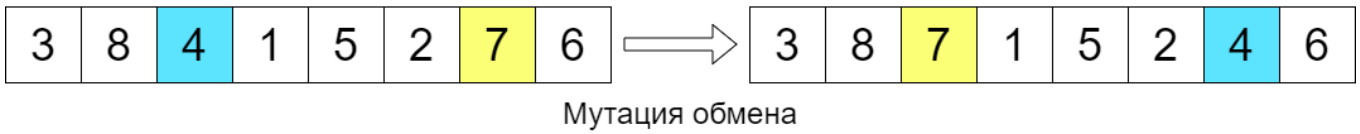


Рис.: Выбор моды для работы

Используемые операторы мутации

Мутация сдвига



Мутация обмена

В качестве оператора мутации использовались операторы k-сдвига (k-shift) и k-обмена (k-exchange), $k = 1, 2, 4$.

Схемы применения мутации: классическая, тяжелые хвосты (heavy-tailed), перемешивающая (scramble) с распределением Пуассона.

Эволюционная стратегия (1+1)-ES

Алгоритм 3 (1+1)-ES (ES)

1: Сгенерировать начальную перестановку π.
2: Пока не выполнено условие остановки, повторять:
2.1: Построить $\pi' = Mut(\pi)$.
2.2: Если π' по целевой функции лучше π, то $\pi = \pi'$.
3: Вернуть π.

Целевая функция рассчитывается как в жадном, но по перестановке.

(1+1)-ES с адаптивной схемой

Алгоритм 4 (1+1)-ES адаптивный (ESA)

1: Сгенерировать начальную перестановку π.
2: Пока не выполнено условие остановки, повторять:
2.1: Выбрать оператор мутации mut.
2.2: Построить $\pi' = mut(\pi)$.
2.3: Если π' по целевой функции лучше π, то $\pi = \pi'$.
2.4: Пересчитать вес оператора mut как
$R_{mut} = \lambda r_{mut} + (1 - \lambda) R_{mut}$, где

$$r_{mut} = \begin{cases} 1, & \text{если получено лучшее решение,} \\ 0.3 & \text{иначе.} \end{cases}$$

3: Вернуть π.

(1+1)-ES с переменными окрестностями

Алгоритм 5 (1+1)-ES с переменными окрестностями (ESVO)

1: Сгенерировать начальную перестановку π.
2: Пока не выполнено глобальное условие остановки, повторять:
2.1: Выбрать следующий оператор мутации mut.
2.2: Пока не выполнено локальное условие остановки, повторять:
2.2.1: Построить $\pi' = mut(\pi)$.
2.2.2: Если π' по целевой функции лучше π, то $\pi = \pi'$.

3: Вернуть π.

Генетический алгоритм

Алгоритм 6 Генетический алгоритм (GA)

1: Создать начальную популяцию из k особей. Сохранить лучшие n_{elites} особей в качестве элиты на основе значений их целевой функции.

2: Пока не выполнено условие остановки, повторять:

2.1: Выбрать особей из текущей популяции в качестве родителей.

2.2: Применить мутацию и рекомбинацию к выбранным особям.

2.3: Сгенерировать потомство.

2.4: Обновить основную популяцию и группу элиты.

3: Вернуть лучшую особь, найденную в процессе работы.

Теорема

Предложенные эволюционные алгоритмы сходятся к оптимальному решению почти наверное при стремлении числа итераций к бесконечности.

Комментарий

1. При выборе очередной работы с вероятностью ϵ выбирается фиктивная работа (с нулевым потреблением ресурсов), а с вероятностью $1 - \epsilon$ выбирается следующая работа из перестановки.
2. При выборе моды с вероятностью δ выбирается случайная мода, с вероятностью $1 - \delta$ выбирается лучшая мода.

Генерация тестовых примеров

Каждая работа j описывается параметрами $v_j, size_j, p_j, b_j$:

x	I_1^x	I_2^x	I_3^x
Объем (V)	$[0, 10)$	$[10, 20)$	$[20, 35)$
Ядра ($SIZE$)	$\{1\}$	$\{2\}$	$\{4\}$
Длительность (P)	$[2, 10)$	$[10, 25)$	$[25, 50)$
Шина (B)	$[0, 30)$	$[30, 80)$	$[80, 100)$

Алгоритм генерации:

1. Выбор класса $\mathcal{K} \in \{\text{Small, Medium, Big}\}$ по распределению $\mathbf{P}_{series} = (p_S, p_M, p_B)$.
2. Каждый параметр x для работы выбирается равномерно из интервала I_i^x. Индекс i выбирается с вероятностью α_i, определенной классом работы: $\mathbf{W}_\mathcal{K} = (\alpha_1, \alpha_2, \alpha_3)$.
   - $\mathbf{W}_{\text{Small}} = (0.7, 0.2, 0.1)$;
   - $\mathbf{W}_{\text{Medium}} = (0.15, 0.7, 0.15)$;
   - $\mathbf{W}_{\text{Big}} = (0.1, 0.2, 0.7)$

Типы серий (по 30 примеров в каждой):

- **STD** (Standard): использование весовых векторов $\mathbf{W}_\mathcal{K}$.
- **RND** (Random): равномерное распределение для классов и параметров.
- **INV** (Inverse): инвертированные интервалы для длительности P ($I_1^P \leftrightarrow I_3^P$).

Результаты эксперимента: одномодальные серии

Серии				Алгоритмы					
Тип	p_S	p_M	p_B	**GR**	**ES**	**ESA**	**ESVO**	**GA**	**LB**
STD	0.7	0.2	0.1	6.25	1.25	1.14	0.64	**0.16***	53.43
STD	0.1	0.2	0.7	3.1	0.61	0.29	0.22	**0.11**	74.68
STD	0.33	0.33	0.34	4.15	0.51	0.28	0.22	**0.16**	67.65
STD	0.15	0.7	0.15	3.86	0.51	0.21	0.23	**0.13**	57.1
STD	0.5	0.0	0.5	4.55	1.04	0.45	0.53	**0.21**	65.58
STD	0.3	0.0	0.7	3.6	0.49	0.26	0.32	**0.08***	72.65
STD	0.7	0.0	0.3	5.8	1.09	0.44	0.41	**0.16***	61.21
RND	-	-	-	4.9	0.59	0.34	0.31	**0.1***	86.46
INV	0.7	0.2	0.1	8.71	2.12	1.58	1.62	**0.06***	33.04
INV	0.1	0.2	0.7	4.17	0.69	0.73	0.85	**0.19**	70.88
INV	0.3	0.0	0.7	6.81	3.03	1.2	0.91	**0.22***	52.35
INV	0.7	0.0	0.3	10.73	1.73	1.27	1.3	**0.02***	36.44

Таблица: Среднее относительное отклонение от лучшего известного рекорда по каждому примеру для $n = 100$ в одномодальных сериях, %.

Результаты эксперимента: одномодальные серии и частичным порядком

Серии					Алгоритмы					
Тип	Порядок	p_S	p_M	p_B	**GR**	**ES**	**ESA**	**ESVO**	**GA**	**LB**
STD	rand	0.33	0.33	0.34	16.77	2.33	0.58	0.47	**0.11***	190.11
RND	rand	-	-	-	13.73	1.98	0.45	0.67	**0.08***	352.3
INV	rand	0.7	0.0	0.3	8.08	0.7	0.23	0.09	**0.04***	179.15
STD	bitree	0.33	0.33	0.34	18.72	2.26	1.01	0.73	**0.12***	92.53
RND	bitree	-	-	-	19.41	2.5	0.71	0.94	**0.12***	140.22
INV	bitree	0.7	0.0	0.3	9.68	1.19	0.46	0.38	**0.06***	97.03

Таблица: Среднее относительное отклонение от лучшего известного рекорда по каждому примеру для $n = 100$ в одномодальных сериях с частичным порядком, %.

Результаты эксперимента: мультимодальные серии

Серии				Алгоритмы					
Тип	p_S	p_M	p_B	**GR**	**ES**	**ESA**	**ESVO**	**GA**	**LB**
STD	0.7	0.2	0.1	6.76	1.63	0.62	0.5	**0.47**	53.95
STD	0.1	0.2	0.7	3.06	0.5	0.24	0.21	**0.2**	74.89
STD	0.33	0.33	0.34	4.38	0.61	**0.19**	0.29	0.32	67.81
STD	0.15	0.7	0.15	4.52	0.65	0.29	**0.17**	0.2	57.25
STD	0.5	0.0	0.5	5.83	0.95	0.39	0.44	**0.21**	65.94
STD	0.3	0.0	0.7	3.68	0.62	0.28	0.22	**0.21**	72.81
STD	0.7	0.0	0.3	8.01	1.06	0.68	0.52	**0.28**	61.14
RND	-	-	-	5.57	0.52	0.29	**0.22**	0.23	86.58
INV	0.7	0.2	0.1	10.32	1.63	1.07	1.15	**0.13***	34.3
INV	0.1	0.2	0.7	4.97	1.23	0.78	0.35	**0.32**	71.09
INV	0.3	0.0	0.7	9.78	3.3	0.73	1.17	**0.61***	53.08
INV	0.7	0.0	0.3	12.91	1.64	1.06	0.94	**0.08***	36.99

Таблица: Среднее относительное отклонение от лучшего известного рекорда по каждому примеру для $n = 100$ в мультимодальных сериях, %.

Результаты эксперимента: мультимодальные серии и частичным порядком

Серии					Алгоритмы					
Тип	Порядок	p_S	p_M	p_B	**GR**	**ES**	**ESA**	**ESVO**	**GA**	**LB**
STD	rand	0.33	0.33	0.34	16.27	2.18	0.47	0.44	**0.16**	189.37
RND	rand	-	-	-	14.35	2.27	0.52	0.38	**0.14***	352.04
INV	rand	0.7	0.0	0.3	8.49	1.15	0.39	0.28	**0.04***	178.13
STD	bitree	0.33	0.33	0.34	20.01	2.49	0.65	0.86	**0.07***	92.46
RND	bitree	-	-	-	20.72	2.92	1.19	0.98	**0.17***	141.05
INV	bitree	0.7	0.0	0.3	12.77	0.96	0.43	0.42	**0.07***	95.88

Таблица: Среднее относительное отклонение от лучшего известного рекорда по каждому примеру для $n = 100$ в мультимодальных сериях с частичным порядком, %.

Результаты эксперимента: PSPLIB

	GR	**ES**	**ESA**	**ESVO**	**GA**
$n = 60$	206.62	16.17	14.35	14.4	**13.5***
$n = 90$	230.42	15.01	13.81	13.83	**12.67***
$n = 120$	557.84	42.96	40.59	40.54	**37.71***

Таблица: Среднее относительное отклонение от длины критического пути для серий из PSPLIB, %.

Заключение

- Получены теоретические оценки в одномодальном случае.
- Предложены жадный алгоритм и метаэвристики для решения задачи в мультимодальном варианте.
- Лидером на различных сериях примеров чаще всего оказывается генетический алгоритм.

Дальнейшие планы

- Исследовать различные стратегии учета ресурсов в жадном алгоритме.
- Исследовать различные функции замедления работ.
- Улучшение теоретических оценок точности алгоритма за счет учета замедления при потреблении шины данных.
- Добавить учет NUMA-архитектуры.

Спасибо за внимание!