Оптимизационные методы искусственного интеллекта

Костенко Валерий Алексеевич
МГУ им. М.В. Ломоносова,
факультет ВМК
kost@cs.msu.su
Сайт: https://asvk.cs.msu.ru/
2025 г.

Алгоритмы имитации отжига

Алгоритм основывается на имитации физического процесса, который происходит при отжиге металлов.

Предполагается, что атомы уже выстроились в кристаллическую решётку, но ещё допустимы переходы отдельных атомов из одной ячейки в другую.

Переход атома из одной ячейки в другую происходит с некоторой вероятностью, причём вероятность уменьшается с понижением температуры.

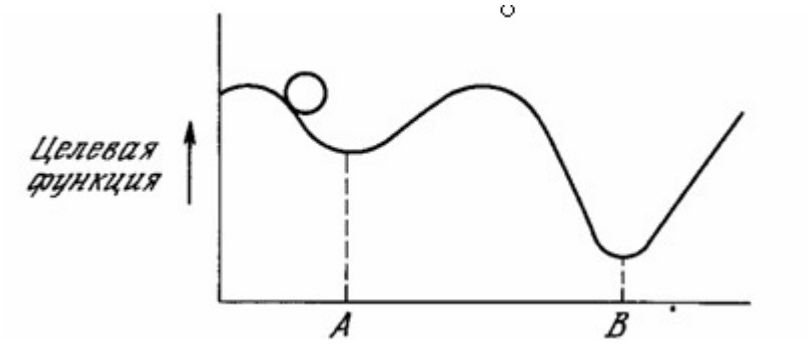
Устойчивая кристаллическая решётка соответствует минимуму энергии атомов, поэтому атом или переходит в состояние с меньшим уровнем энергии, или остаётся на месте.

Вероятностный закон принятия нового решения текущим решением

- Алгоритмы имитации отжига с некоторой вероятностью допускают переход в состояние с более высоким значением целевой функции:

$$P(X^k \to X^{k+1}) = \begin{cases} 1, & \Delta F \le 0 \\ \exp(-\Delta F / T), & \Delta F > 0 \end{cases}$$

- где T - некоторая температура, ΔF - изменение целевой функции.
- В ходе работы алгоритма температура постепенно снижается.



Общая схема

1. Задать начальное корректное решение $X_0 \in S$ и считать его текущим вариантом решения: $X=X_0$.
2. Установить начальную температуру T_0, приняв ее текущей: $T=T_0$
3. Применить операции преобразования решения к X и получить новый вариант решения $X' \in S$,
4. Найти изменение функции оценки качества решения : $\Delta F = F(X') - F(X)$:
   - если $\Delta F \le 0$ (решение улучшилось), то $X=X'$;
   - если $\Delta F > 0$, то принять с вероятностью P в качестве текущего решения новый вариант решения X';
   - если это решение является лучшим из ранее найденных, то запомнить его.
5. Повторить заданное число раз шаги 3 и 4 без изменения текущей температуры.
6. Если критерий останова выполнен, то завершение работы.
7. Понизить текущую температуру в соответствии с выбранным законом понижения и перейти к шагу 3..

Вероятность принятия решения на шаге 4:

$$p = e^{\frac{-\Delta F}{T}}$$

Законы понижения температуры на шаге 7:

- Закон Больцмана: $T = \frac{T_0}{\ln(1+i)}$
- Закон Коши: $T = \frac{T_0}{1+i}$
- $T = T_0 \frac{\ln(1+i)}{1+i}$

i - номер итерации алгоритма

1. Разработать способ представления решения *X* и операций преобразования текущего решения на шаге 3. Должны удовлетворять условиям:
   - замкнутости,
   - полноты.

2. Разработать стратегию применения операций преобразования текущего решения на шаге 3:
   - какую операцию применять,
   - к какому элементу *X*, как его изменять.

3. Выбрать закон понижения температуры на шаге 7.

4. Определить функцию для оценки качества текущего решения на шаге 4.

5. Выбрать критерий останова алгоритма, используемый на шаге 6.

Штрафные и барьерные функции

Пусть ищется минимальное значение функции *F(x)* и переменная *x* ограничена значением, строго меньшим, чем заданная константа *b*.

Штрафная функция *F(x)+g(x)* — непрерывная функция, значение которой при нарушении границы области допустимых решений увеличивается по сравнению со значение целевой функции на значение функции g(x).

Барьерная функция — непрерывная функция, значение которой стремится к бесконечности при приближении точки к границе области допустимых решений.

Что бы алгоритм не выходил из области допустимых решений функция *F(x)* может быть заменена на функцию $F(x) - g(x,b)$.

Для логарифмических барьерных функций *g(x,b)* определяется как $\log(b - x)$ для всех $x<b$ и ∞ в противном случае .

$\log(b - x)$ будет стремится к минус бесконечности, когда $(b - x)$ стремится к нулю.

Асимптотическая скорость сходимости

Чтобы достичь наперед заданной точности, нужно совершить число итераций, пропорциональное квадрату от размера пространства поиска.

Повышение эффективности алгоритма

- Динамические законы понижения температуры.
- Распараллеливание алгоритмов.
- Эвристические (направленные) операции преобразования текущего решения.

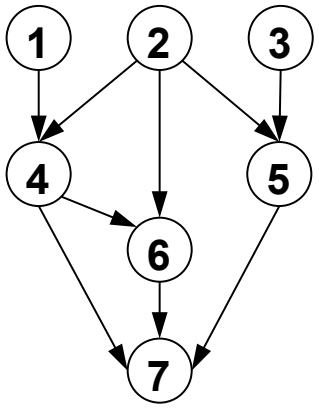
Задача построения расписания

- Модель программы *H задается*

  графом потока данных $H = \{P, L\}$:

  – $P = \{ P_i \}$ – множество работ,

  – L – ограничения на порядок выполнения работ.



- Метки работ p_i:

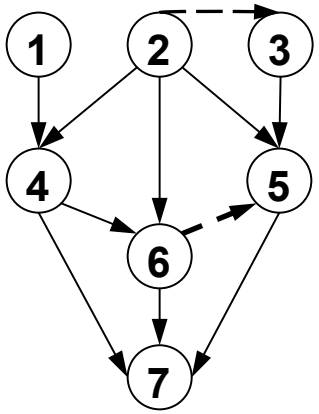
  – N_i – номер,

  – W_i - вычислительная сложность.

Модель расписания

HP - расписание

$HP = (HP_B, HP_L)$

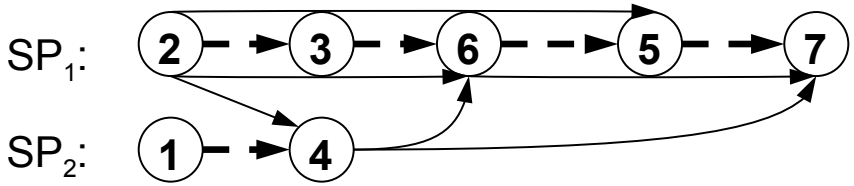


$HP_B : p^* \rightarrow PU$ - привязка работ

p^* - множество всех работ

PU - множество всех процессоров

HP_L - порядок выполнения работ



Условия корректности расписания

- ацикличность

$$(p_i, p_j) \in HP_L^* \Rightarrow (p_j, p_i) \notin HP_L^*,$$

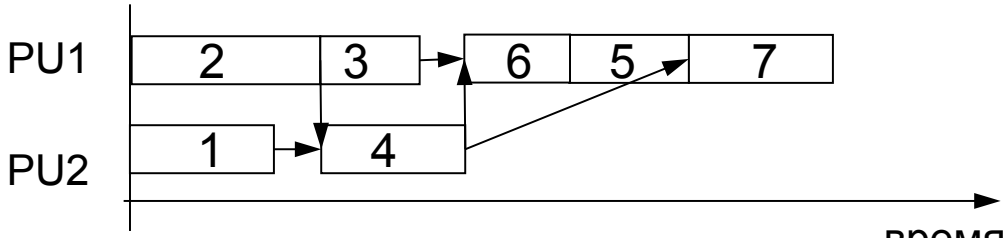
HP_L^* - *транзитивное замыкание* HP_L

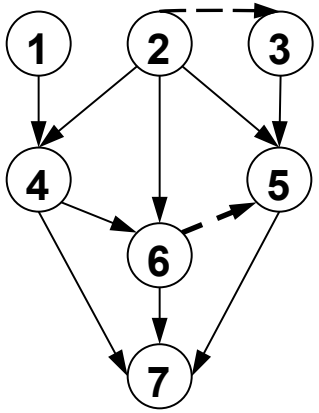
- частичный порядок заданный в *H*, должен быть сохранен в *HP*

$$L \subseteq HP_L$$

- каждая работа должна быть распределена на какой-либо процессор и только на один процессор.

Временная диаграмма выполнения программы





Формальная постановка задачи построения расписаний

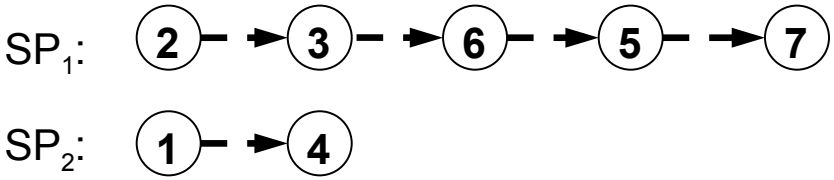
- Дано:
  - *H = (P, L)* - модель программы,
  - *Tr = f(HP)* - функция построения временной диаграммы.
- Задача:
  - построить расписание *HP.*
- Критерий оптимальности:

$$\min_{HP \in HP^*} T(f(H, HP))$$

- Ограничения: $HP \in HP^*$

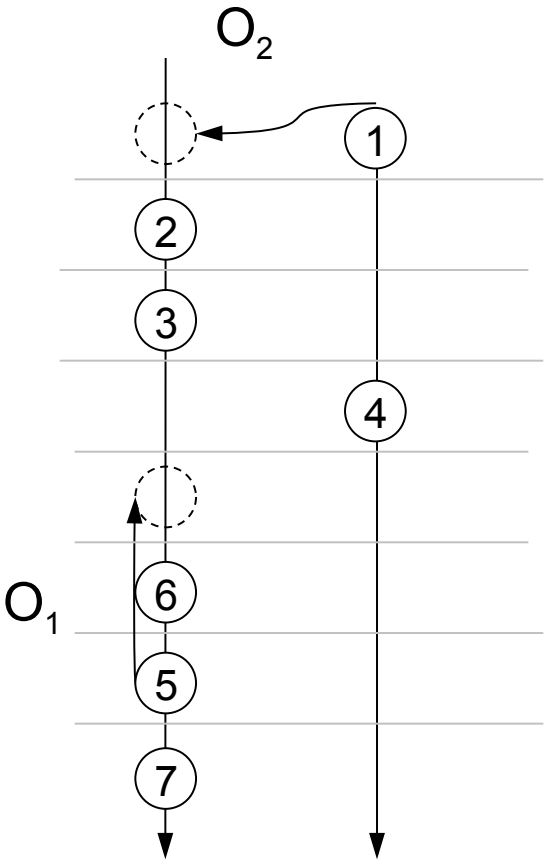
*HP** – множество расписаний удовлетворяющих условиям корректности. (Множество корректных расписаний)

Базовая система операций преобразования расписаний



Теорема. Для любых двух корректных расписаний HP_1 и HP_2 существует цепочка операций $\{O_1, O_2\}$, переводящая расписание HP_1 в HP_2 так, что каждое промежуточное расписание корректно и длина цепочки <=2N.

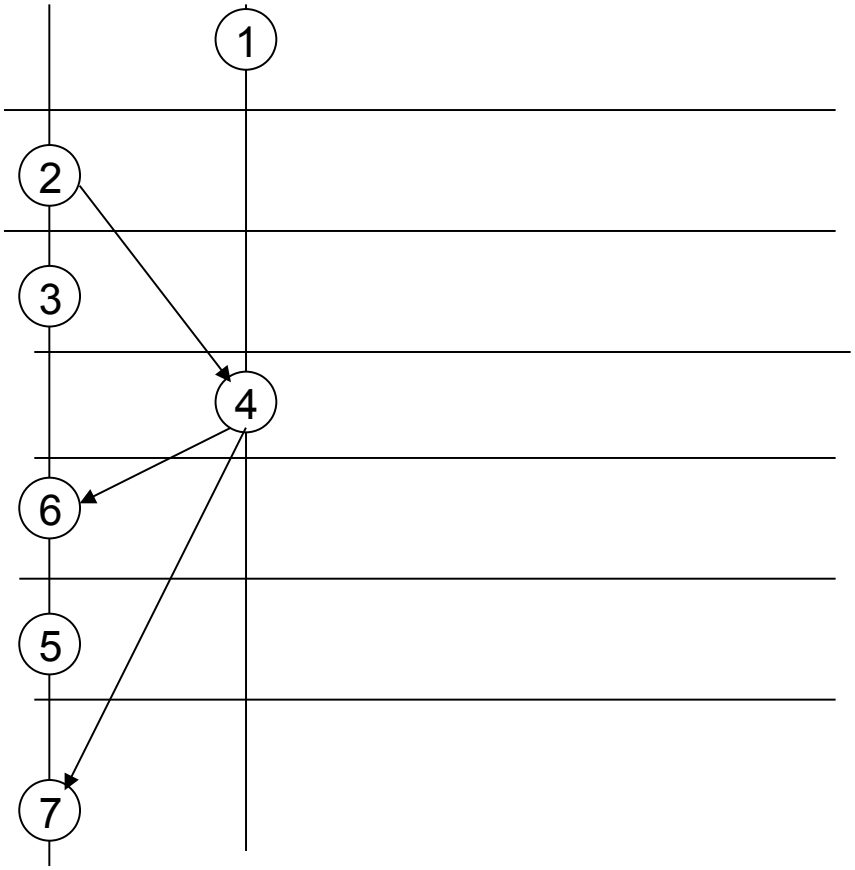
ярусное представление

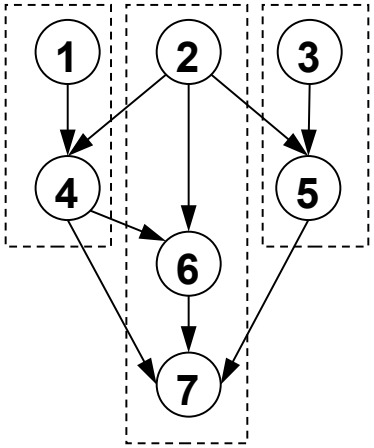


Доказательство теоремы

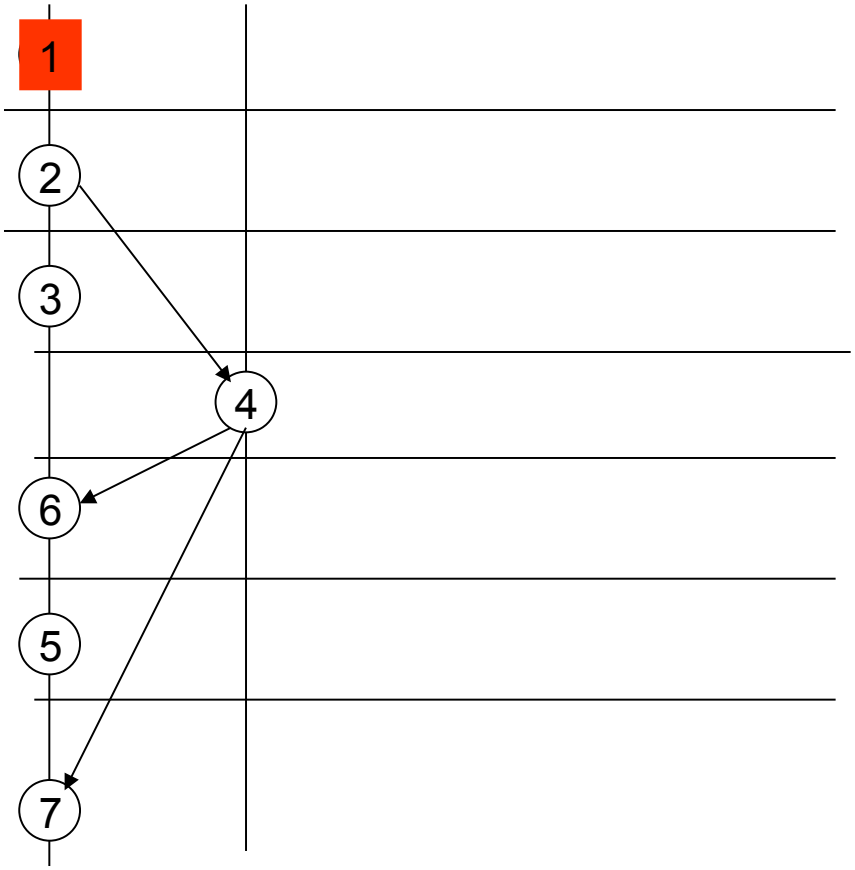
1. Нумерация вершин графа *H* является сквозной, удовлетворяет условиям полной топологической сортировки и предшествинник всегда имеет меньший номер.
2. Введем понятие канонического расписания HP^0: все вершины находятся в SP_1 и порядковые вершин в SP_1 равны номерам вершин в графе *H*.
3. Доказываем, что существует стратегия (последовательность выбора вершин) с числом шагов $K \leq N$ позволяющая перевести произвольное расписание *HP* в расписание HP^0, такая, что все промежуточные расписания *HPi* (полученные после выполнения отдельной операции O_i) являются допустимыми и любая операция из цепочки является обратимой.
4. Поскольку, каждая операция из цепочки обратима, то существует стратегия перехода от HP^0 к произвольному *HP*, такая, что все *K* промежуточных расписаний являются допустимыми и $K \leq N$.

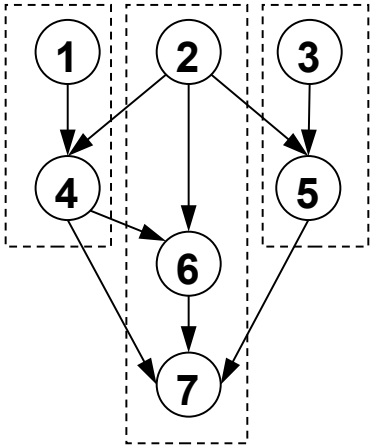
Теорема (граф. ил. доказательства)



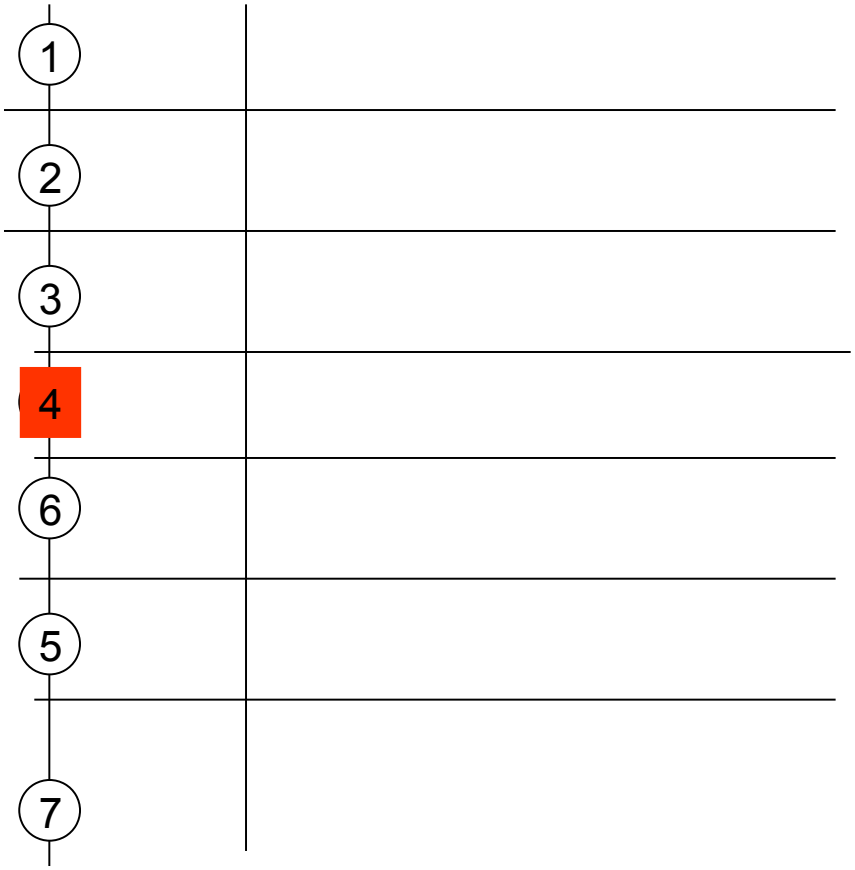


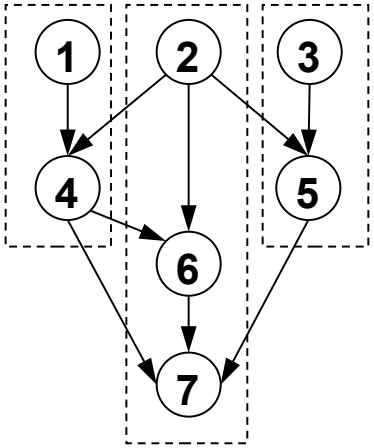
Теорема (граф. ил. доказательства)



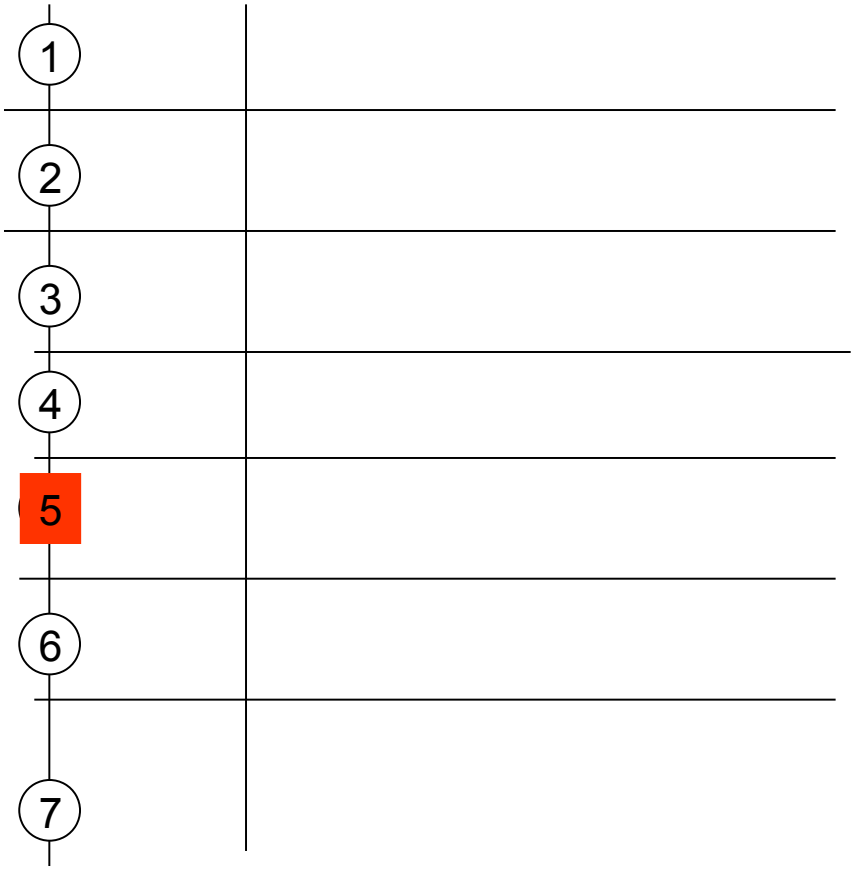


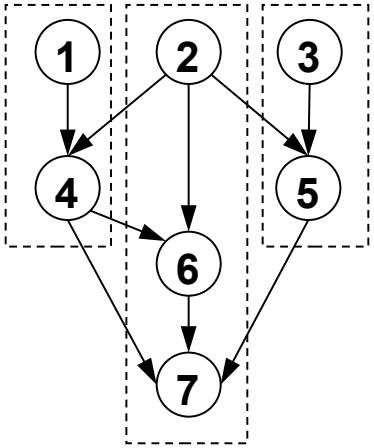
Теорема (граф. ил. доказательства)





Теорема (граф. ил. доказательства)





Костенко В.А. Задача построения расписания при совместном проектировании аппаратных и программных средств // Программирование - 2002. - №3 - С.64-80.

Kostenko V.A. The Problem of Schedule Construction in the Joint Design of Hardware and Software. // Programming and Computer Software – 2002. -Vol. 28 - № 3 - pp.162–173.

Направленные стратегии применения операций

Стратегия уменьшения задержек.

Утверждение. Если время начала выполнения каждой работы равно длине критического пути в графе потока данных, то расписание будет иметь минимальное время выполнения.

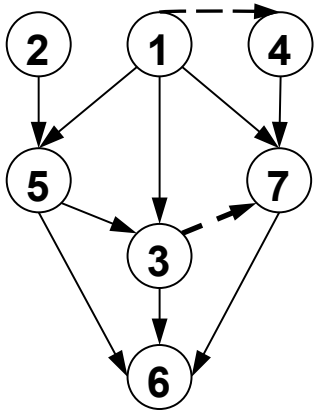
Стратегия заполнения простоев.

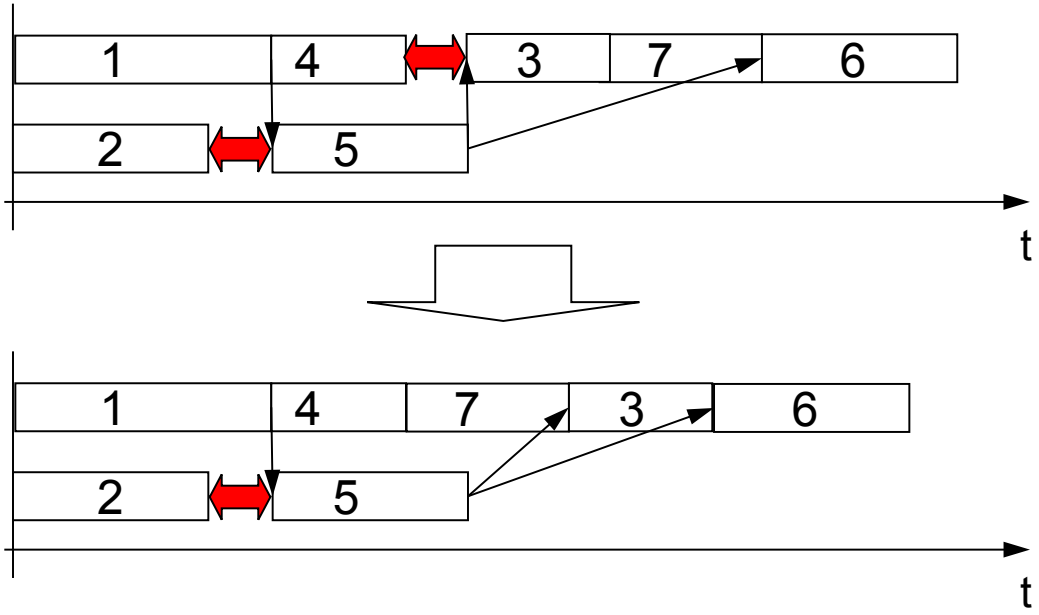
Утверждение. Если простоев процессоров нет, то расписание будет иметь минимальное время выполнения.

Пример - Анализ временной диаграммы для устранения простоев процессоров

<u>причины возникновения простоев:</u>

- ожидание работой завершения предшественника
- ожидание освобождения разделяемого ресурса





- передача данных
- простой процессора
- процесс

Метрика в пространстве корректных расписаний

Определение. Расстояние между расписаниями HP_1 и HP_2 определим как число, равное длине минимальной последовательности операций преобразования $\{O_1, O_2\}$, которая переводит расписание HP_1 в расписание HP_2, так, что все промежуточные расписания корректны.

Утверждение. Введенное расстояние между расписаниями является метрикой.

Динамический закон понижения температуры

стандартная функция

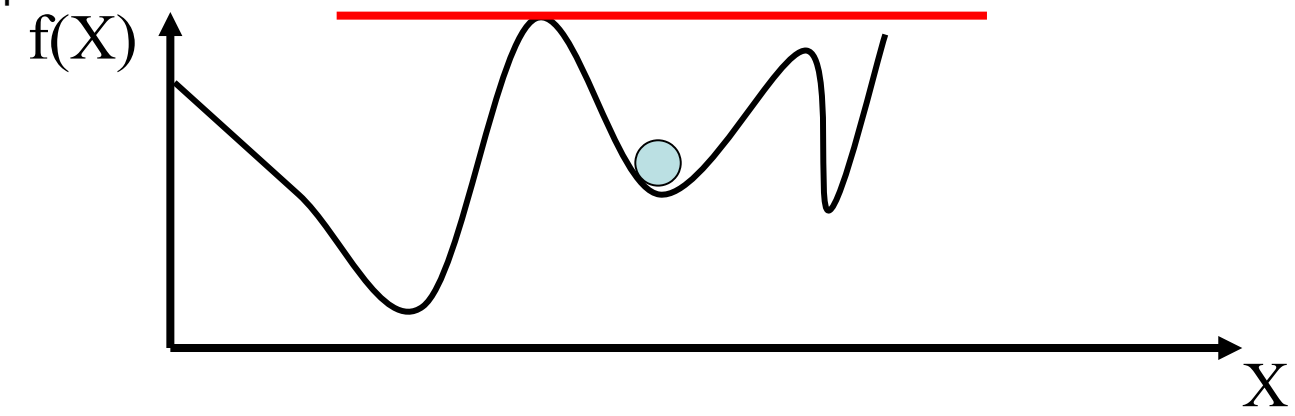
$$T = \frac{T_0}{\ln(1+i)}$$

модификация стандартной функции

$$T = \min\left(\frac{T_0}{\ln(1+i)} + \max(0, c_t(k - k_t)), T_0\right)$$

k_t - порог начала повышения T, k – количество итераций без улучшения решения

Уменьшение количества итераций на 6-10% с сохранением качества решения



Способы распараллеливания алгоритмов

- Универсальные способы распараллеливания итерационных алгоритмов
  - Запуск с различными начальными приближениями
  - Запуск конвейера
- Распараллеливание имитационной модели
- Способы распараллеливания, опирающиеся на особенности задачи
  - Декомпозиции целевой функции
  - Декомпозиции пространства решений

Схема параллельного асинхронного алгоритма

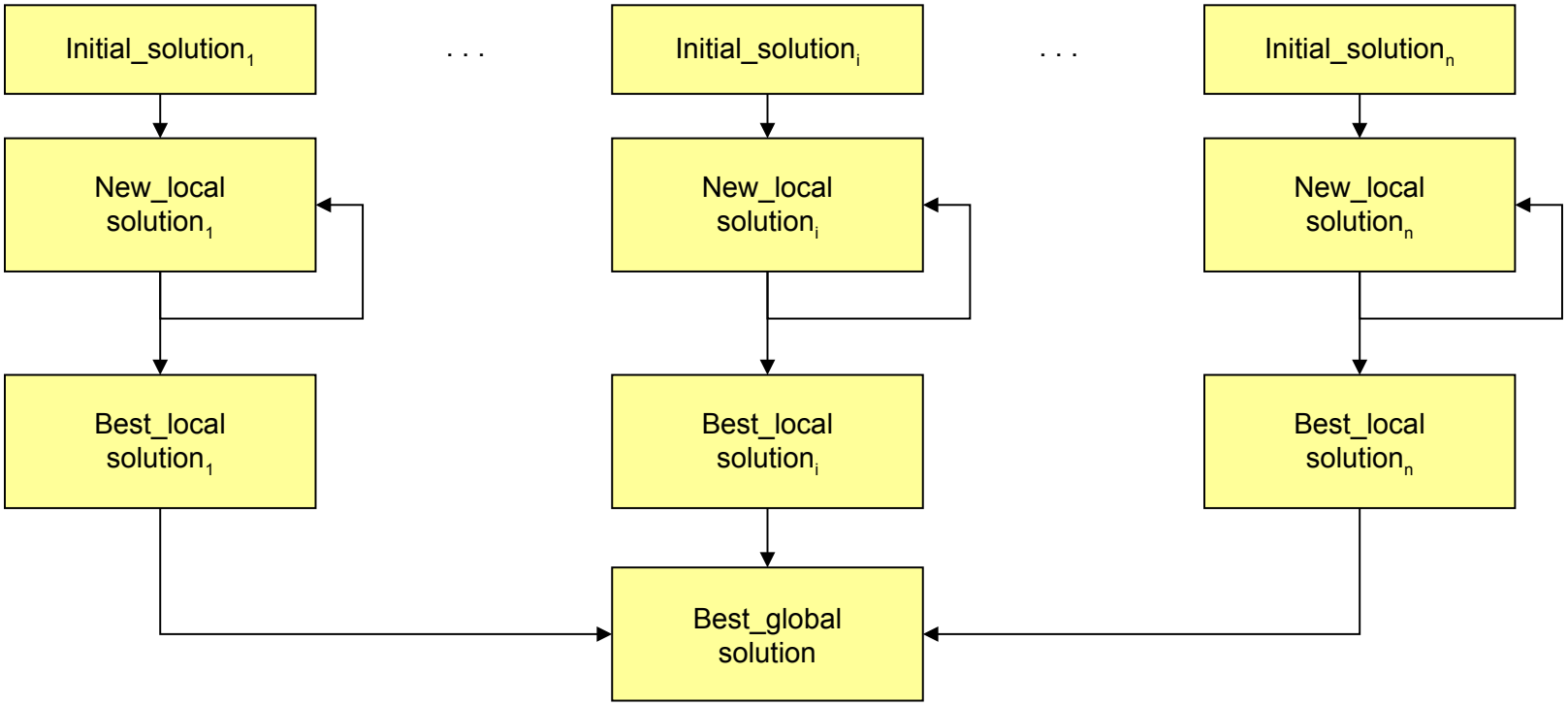
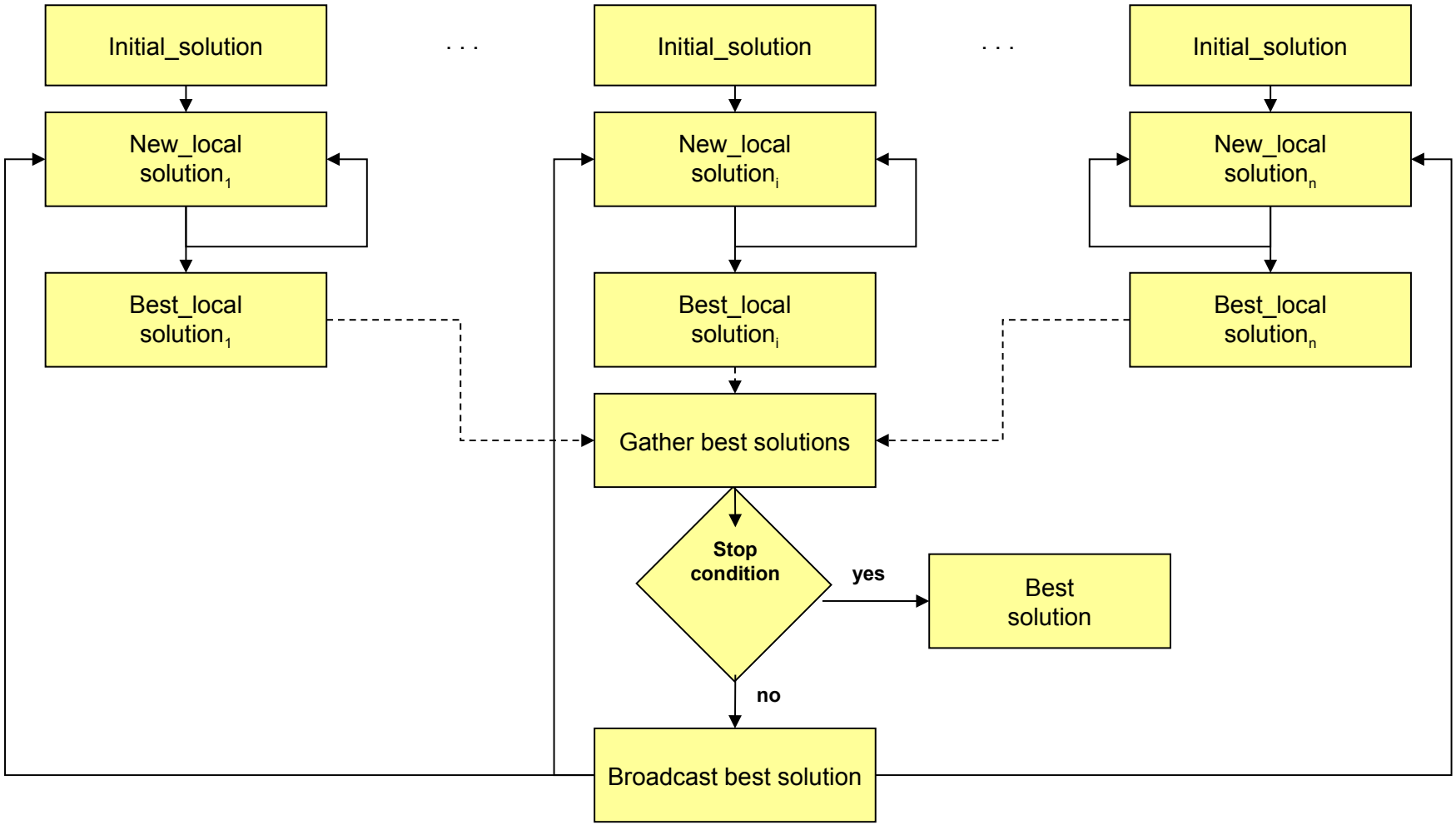


Схема параллельного алгоритма с синхронизацией



Способы обмена полученными решениями и стратегии принятия решения в качестве текущего

- Процесс *i* посылает процессу *i+1* своё текущее решение. Процесс *i+1* принимает это решение в качестве текущего, если $F(X_i) < F(X_{i+1})$.

- Широковещательный обмен. Принятие *j*-ым процессом решения от *i*-ого процесса в качестве текущего, если $F(X_i) < F(X_j)$ и $i < j$.

- Отправка текущих решений координатору, выбор из них произвольного (возможно с разными вероятностями, в зависимости от значений целевой функции) и принятие его в качестве текущего всеми процессами.

Алгоритм, основанный на декомпозиции целевой функции

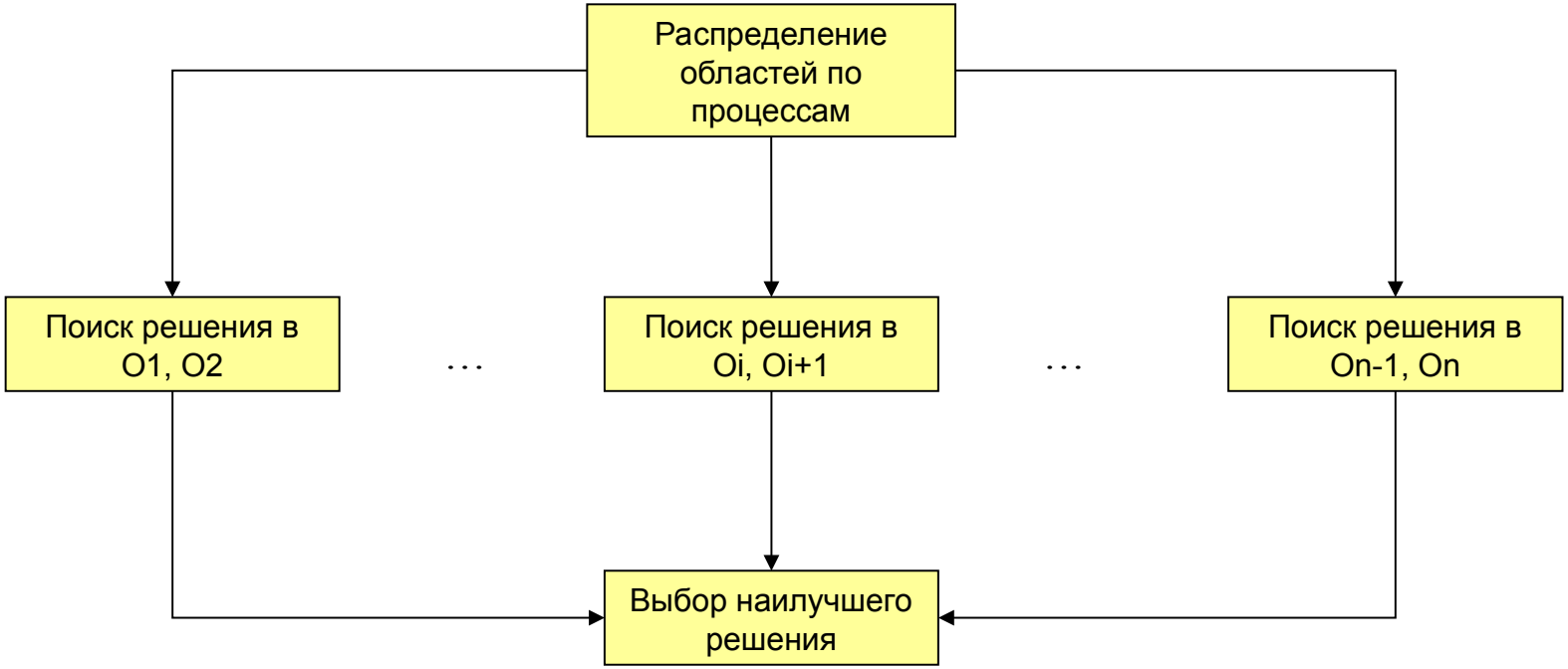
Большую часть времени алгоритм имитации отжига затрачивает на вычисление целевой функции.

Если целевая функция является декомпозируемой

$F(x_1, x_2, \ldots, x_i, \ldots, x_n) = F(x_1) + \ldots + F(x_i) + \ldots + F(x_n)$,

то возможно параллельное её вычисление на m ($m \leq n$) процессорах.

Подходы, основанные на декомпозиции пространства решений



Распараллеливание на основе декомпозиции пространства решений

- **Задачи, требующие решения :**
  - Разбиение пространства корректных расписаний на непересекающиеся области.
  - Модификация операций преобразования расписания.
  - Поиск оптимального расписания в каждой области отдельно.
  - Отсечение областей, которые заведомо не содержат оптимального решения.
  - Распределение областей по узлам ВС.

Разбиение на области

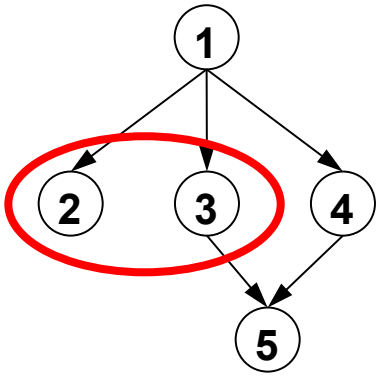
Область 1

$$HP_B(p_i) \neq HP_B(p_j)$$

Исходная область

Выбор двух работ, не связанных отношением порядка

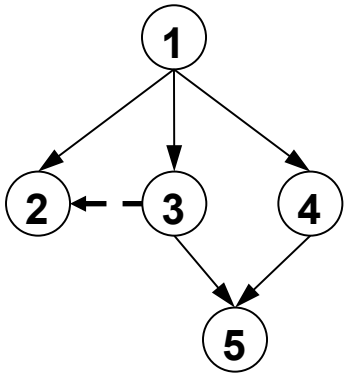
$$p_i = 2, p_j = 3$$



Область 2

$$(p_j, p_i) \in HP_L$$

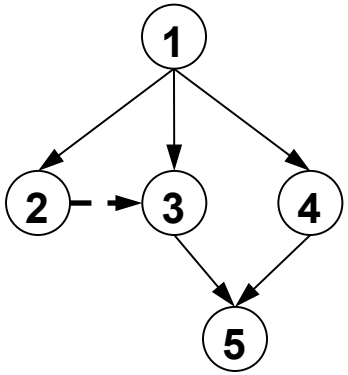
$$HP_B(p_i) = HP_B(p_j)$$



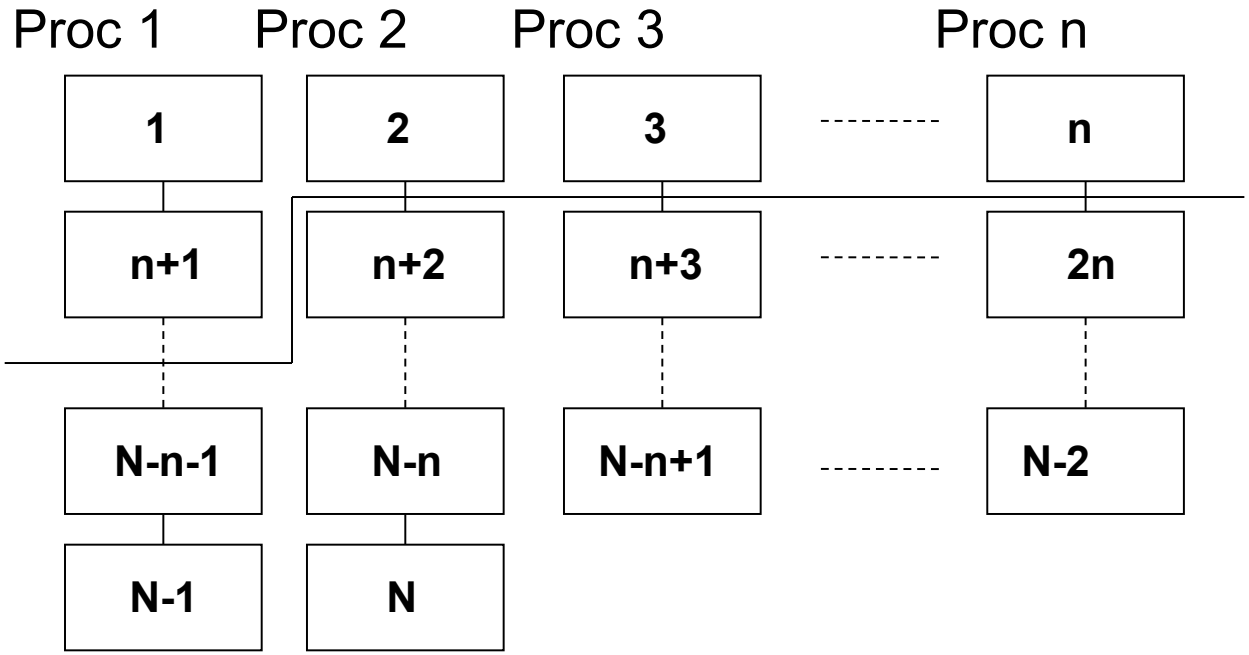
Область 3

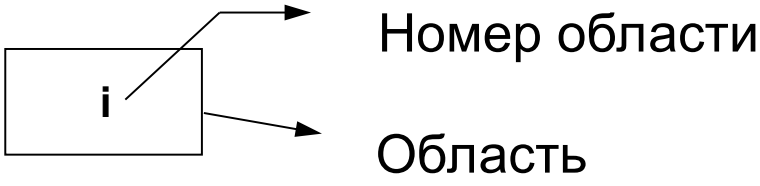
$$(p_i, p_j) \in HP_L$$

$$HP_B(p_i) = HP_B(p_j)$$



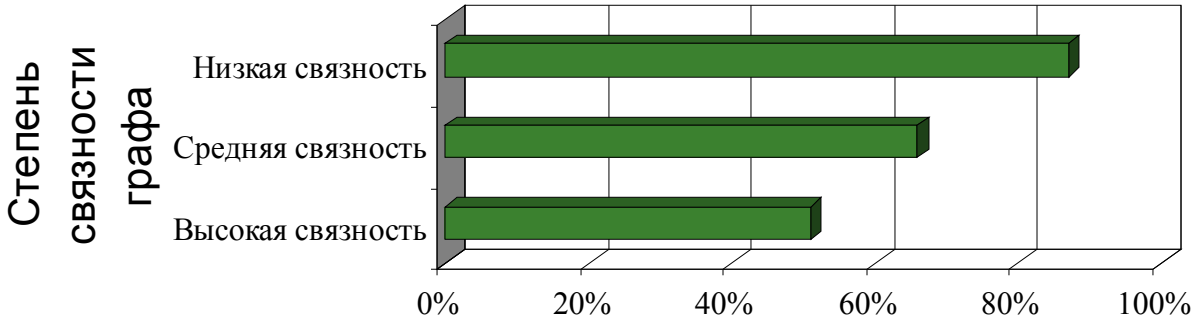
Распределение по узлам



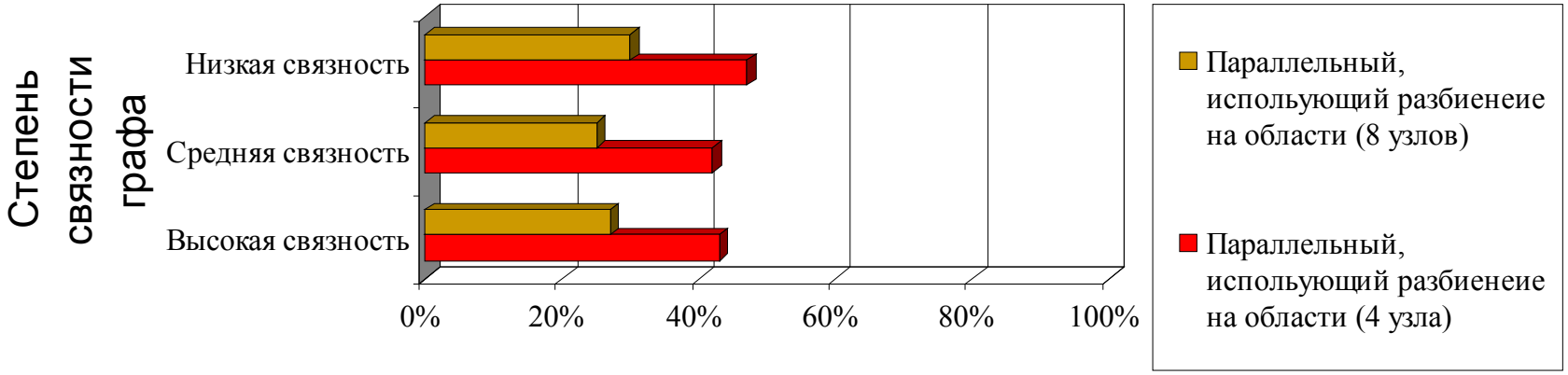


Области отсортированы в порядке возрастания нижней оценки целевой функции

Результаты исследования параллельного алгоритма имитации отжига



Сравнение классического алгоритма и алгоритма, использующего разбиение на области. (100% - время работы классического алгоритма)



Сравнение параллельного алгоритма с последовательным, использующим разбиение на области (100% - время работы последовательного алгоритма)

Костенко В.А. Задача построения расписания при совместном проектировании аппаратных и программных средств // Программирование - 2002. - №3 - С.64-80.

Kostenko V.A. The Problem of Schedule Construction in the Joint Design of Hardware and Software. // Programming and Computer Software – 2002. -Vol. 28 - № 3 - pp.162–173.

Калашников А.В., Костенко В.А. Параллельный алгоритм имитации отжига для построения многопроцессорных расписаний // Известия РАН. Теория и системы управления. - 2008. -№ 3 - С.133-142.

A. V. Kalashnikov and V. A. Kostenko. A Parallel Algorithm of Simulated Annealing for Multiprocessor Scheduling // Journal of Computer and Systems Sciences International. – 2008. - Vol. 47 - № 3- pp.455-463.

Костенко В.А. Алгоритмы построения расписаний для вычислительных систем реального времени, допускающие использование имитационных моделей // Программирование - 2013. - №5 - С.53-71.

Kostenko V.A. Scheduling Algorithms for Real-Time Computing Systems Admitting Simulation Models // Programming and Computer Software, 2013, Vol. 39, № 5, pp.255–267.

Д.А.Зорин, В.А.Костенко. Алгоритм имитации отжига для решения задач построения многопроцессорных расписаний // Автоматика и телемеханика, 2014, № 10, С. 97-110.

D. A. Zorin, V. A. Kostenko. Algorithm to Simulate Annealing in Problems of Multiprocessor Scheduling // Automation and Remote Control, 2014, Vol. 75, No. 10, pp. 1790–1801. DOI: 10.1134/S0005117914100063

Задание 3 и 4:

Доказать, что введенное расстояние между расписаниями является метрикой.

Доказать замкнутость и полноту модифицированной системы операции в областях.