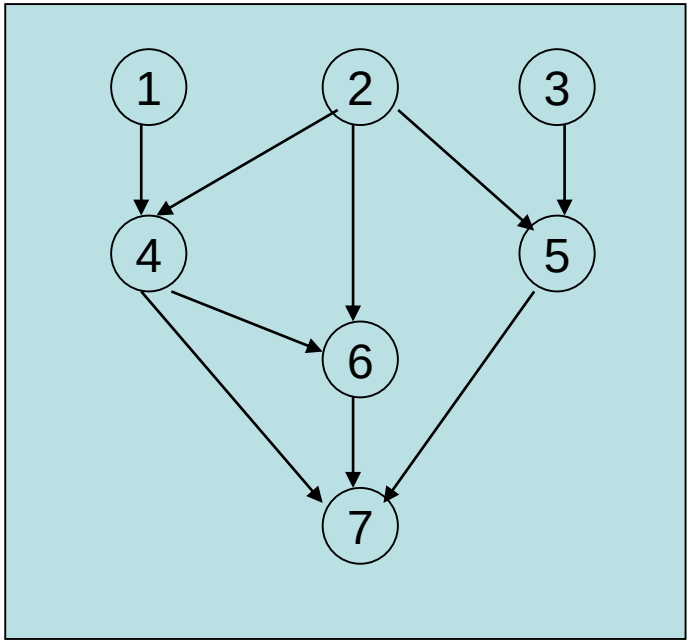
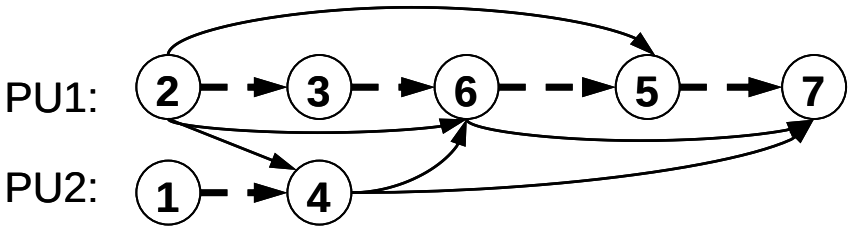
Алгоритмы планирования вычислений в ИУС РВ

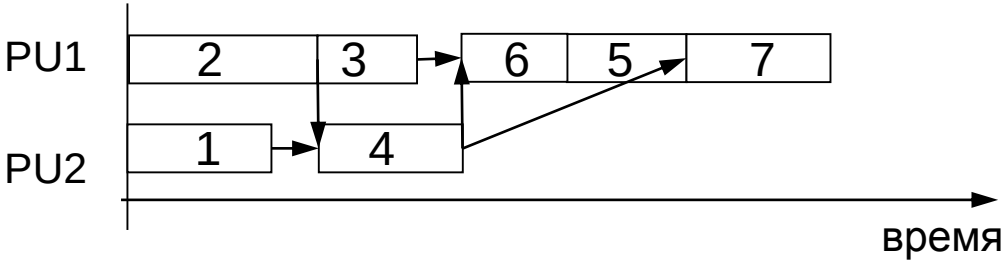
Кафедра АСВК,
Лаборатория Вычислительных Комплексов
Костенко В.А.

Способы представления расписания

1. Временная диаграмма – для каждой работы задано время начала выполнения $s'(t_j)$ и процессор на котором она выполняется.
2. Привязка работ к процессорам и порядковый номер выполнения работы на процессоре.
3. Привязка работ к процессорам и приоритет работы.
4. Статико-динамические расписания.


1
2
3
4
5
6
7


PU1:
PU2:
2
3
6
5
7
1
4


PU1
PU2
2
3
6
5
7
1
4
время

Меры оценки эффективности расписания

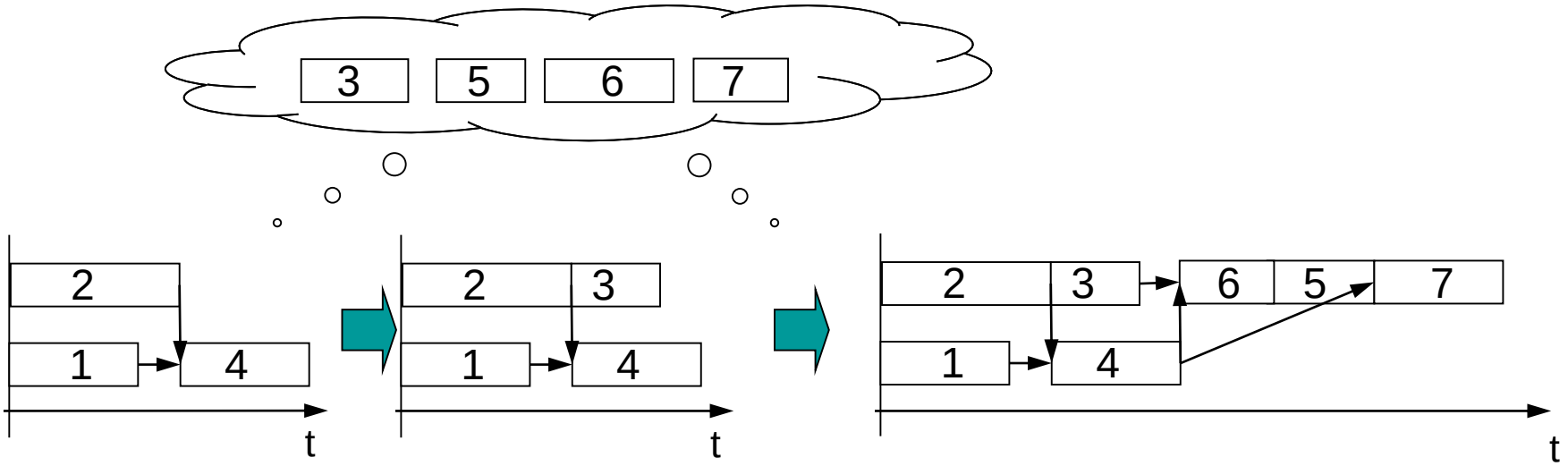
1. Время выполнения расписания.

2. Число используемых процессоров для выполнения множества работ за время не превышающее заданные директивный срок.

3. Максимальное число совместимых работ.

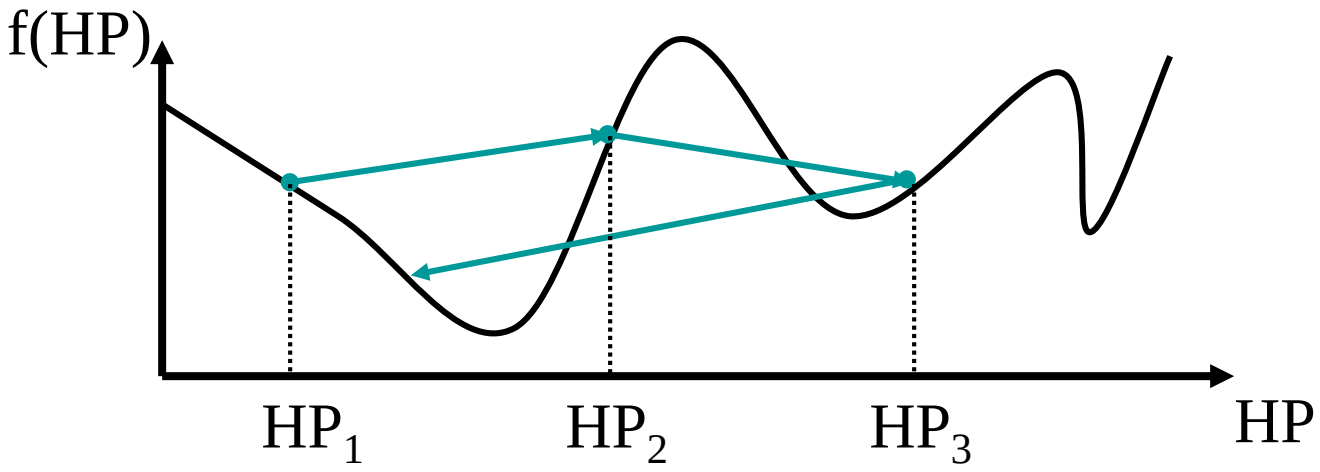
4. Критерии, основанные на использовании функций штрафа за нарушение директивных сроков работ.

Конструктивные алгоритмы построения расписаний



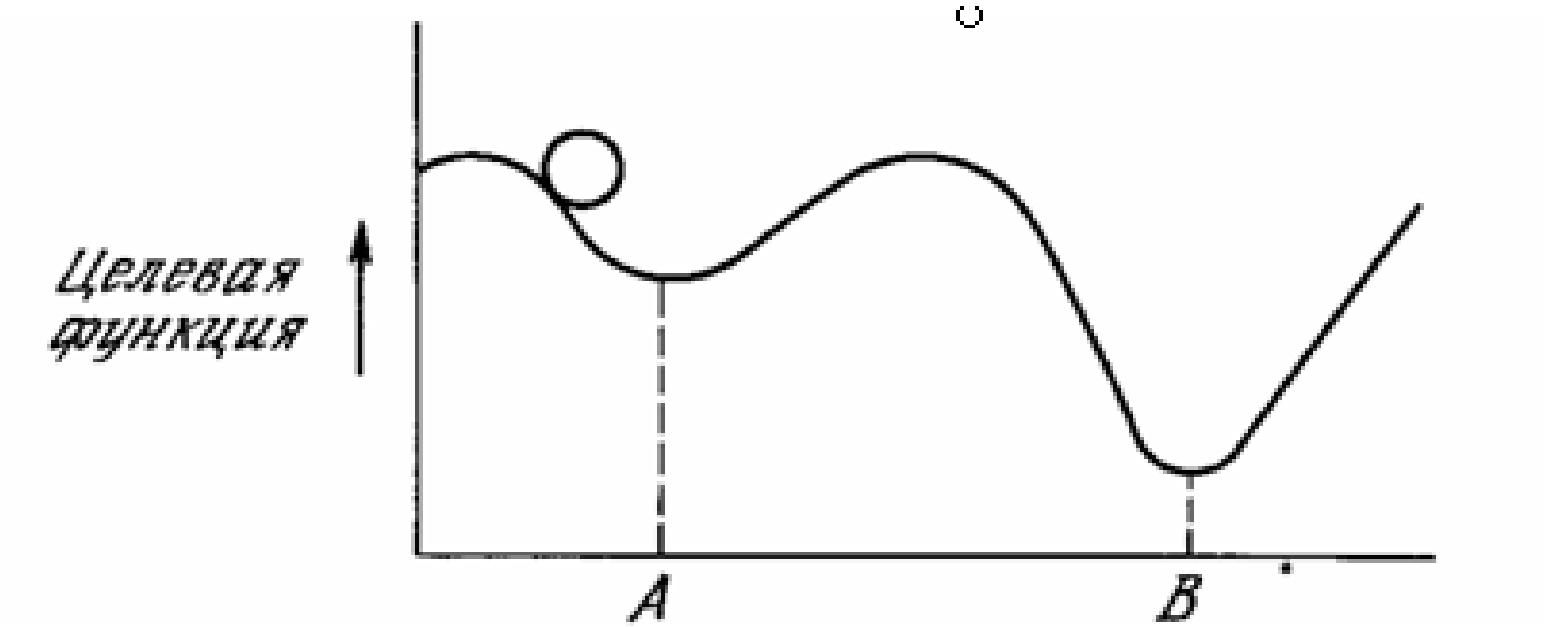
- Жадные алгоритмы
- Метод ветвей и границ
- Алгоритмы сочетающие жадные стратегии и стратегии ограниченного перебора

Итерационные алгоритмы построения расписаний



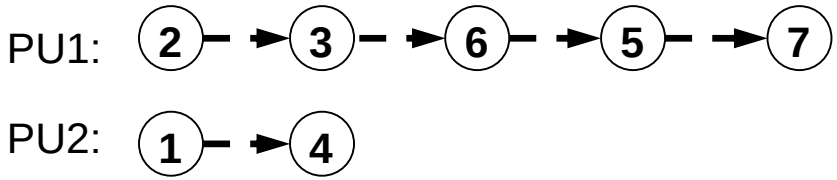
- Алгоритмы случайного поиска
- Имитация отжига
- Алгоритмы локальной оптимизации
- Генетические и эволюционные алгоритмы

Алгоритмы имитации отжига (принцип работы)

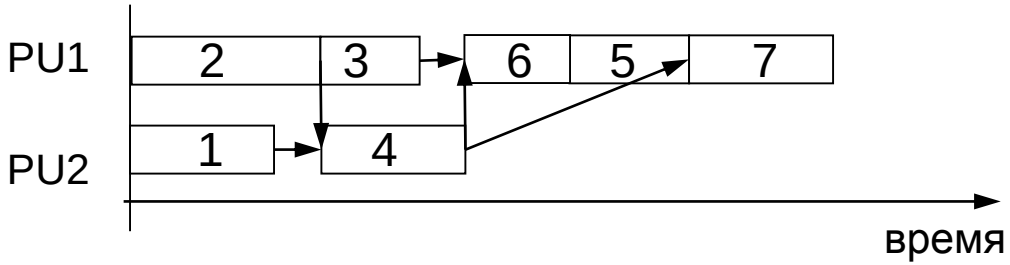


Алгоритмы имитации отжига (общая схема одной итерации)

1. HP=

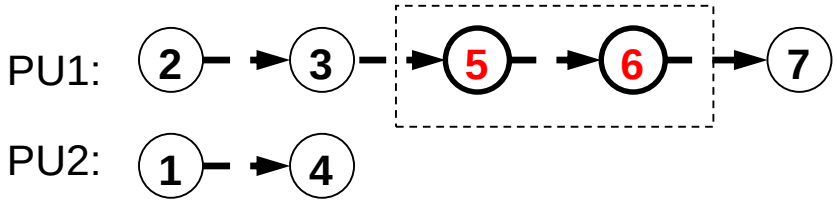


2. f(HP)=

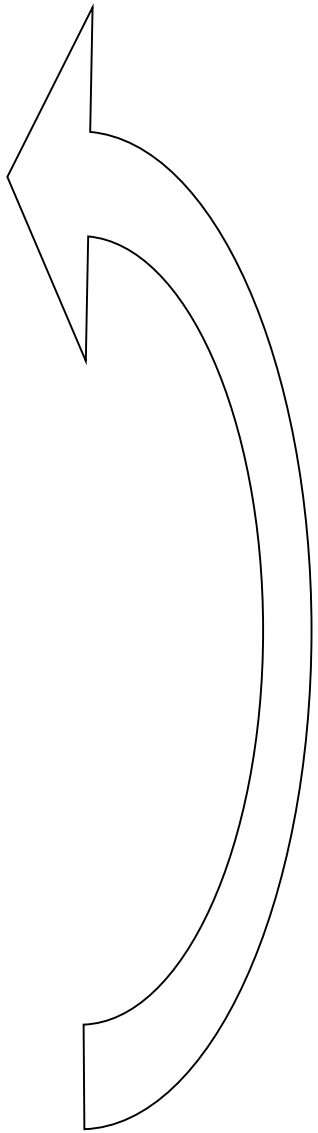


3. Проверка критерия останова

4. HP =



5. Выбор текущего приближения HP

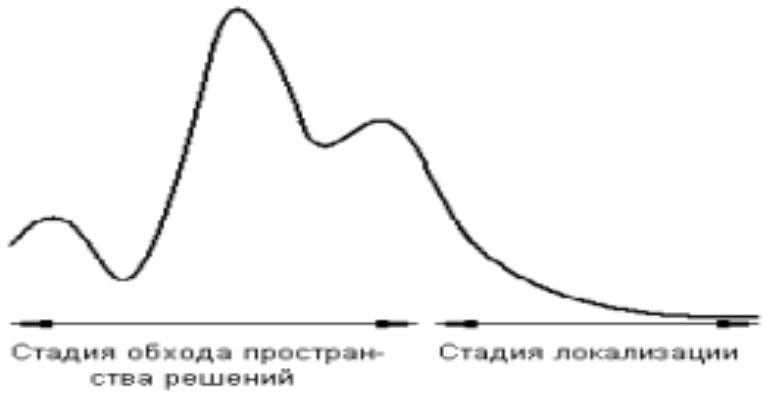


Алгоритмы имитации отжига (общая схема)

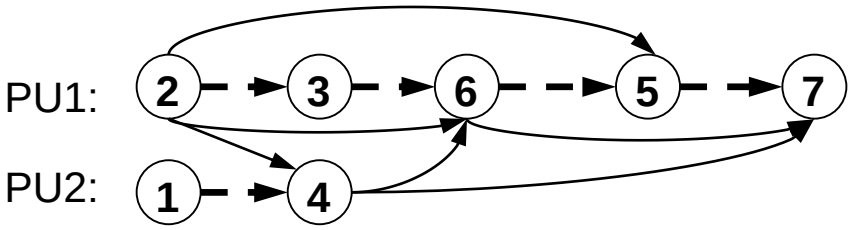
5. Алгоритмы имитации отжига с некоторой вероятностью допускают переход в состояние с более высоким значением целевой функции:

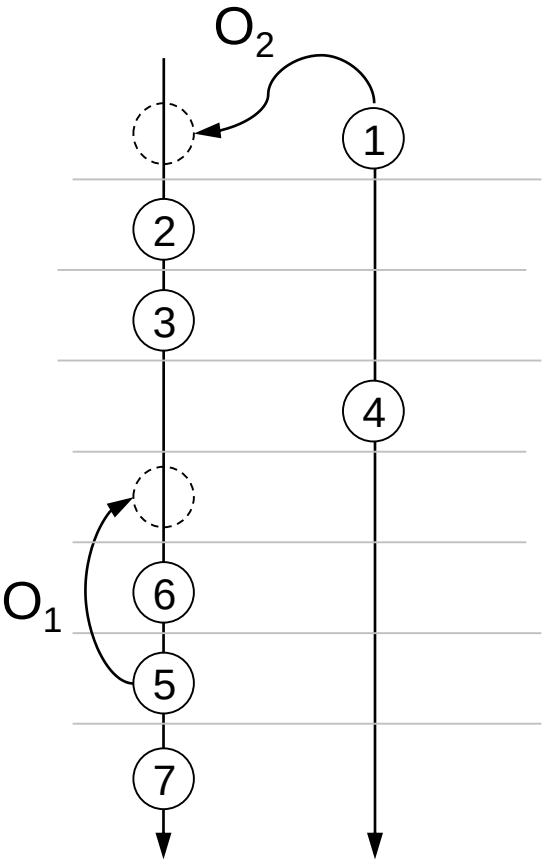
$$P(HP^k \to HP^{k+1}) = \begin{cases} 1, & \Delta f \le 0 \\ \exp(-\Delta f / T), & \Delta f > 0 \end{cases}$$

Поиск продолжается до тех пор, пока алгоритм не попадает в минимум, из которого он уже не может выйти за счет тепловых флуктуаций



Алгоритмы имитации отжига (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 $<=2N$.

Алгоритмы имитации отжига (асимптотическая скорость сходимости)

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

Алгоритмы имитации отжига (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.

Генетический алгоритм Холланда (SGA)

- Holland J.N. Adaptation in Natural and Artificial Systems. Ann Arbor, Michigan: Univ. of Michigan Press, 1975.

Генетический алгоритм Холланда (SGA)

- Основан на использовании механизмов естественной эволюции:
  1. Изменчивость → операция мутации
  2. Наследственность → операция скрещивания
  3. Естесственный отбор → операция селекции

Основные понятия

- ***Популяция*** - это множество битовых строк.
- ***Каждая строка*** - одно из возможных решений задачи.
- По строке может быть вычислена ***функция выживаемости***, которая характеризует качество решения.
- Основные операции алгоритма: ***селекция, скрещивание и мутация*** выполняются над элементами популяции.

Схема ГА

1. Сгенерировать случайным образом популяцию размера P.
2. Вычислить функцию выживаемости для каждой строки популяции.
3. Выполнить операцию селекции.
4. Выполнить операцию скрещивания:
   - 4.1. Выбрать пары для скрещивания.
   - 4.2. Для каждой выбранной пары выполнить скрещивание, получить двух потомков и произвести в популяции замену родителей на их потомков.
5. Выполнить операцию мутации.
6. Если критерий останова не достигнут, перейти к п.2, иначе завершить работу.

Требования к кодированию решений

1. Однозначность: каждая закодированная строка должна соответствовать единственному решению исходной задачи.

2. Возможность кодирования любого допустимого решения.

3. Получение в результате генетических операций корректных вариантов решений.

4. Возможность перехода от любого корректного решения к любому другому корректному решению.

Требования к кодированию решений

- Для задач непрерывного и целочисленного мат. программирования, оптимизируемые параметры задаются:
  - ***двоичным кодом числа,***
  - ***кодами Грея***.

Создание начальной популяции

- Случайным образом генерируется начальная популяция в пределах допустимых значений (в области поиска):

$$X_1[10100..01], \quad X_2[11100..11], \quad \dots, \quad X_N[01010..10]$$

Функция выживаемости

- Выбирается согласно предметной области
- Определяет качество решения
- Применяется ко всем элементам популяции

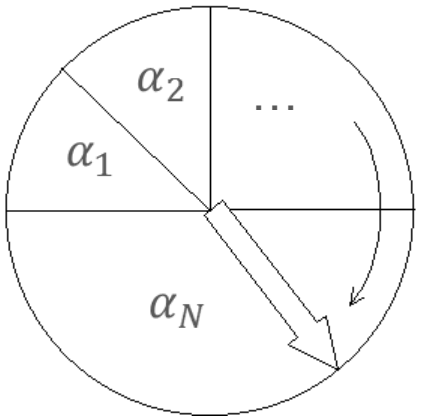
Операция селекции

- Схема пропорциональной селекции:

$$\bar{F} = \frac{\sum F_i}{N}, \quad r_i = \left\lfloor {F_i}/{\bar{F}} \right\rfloor$$

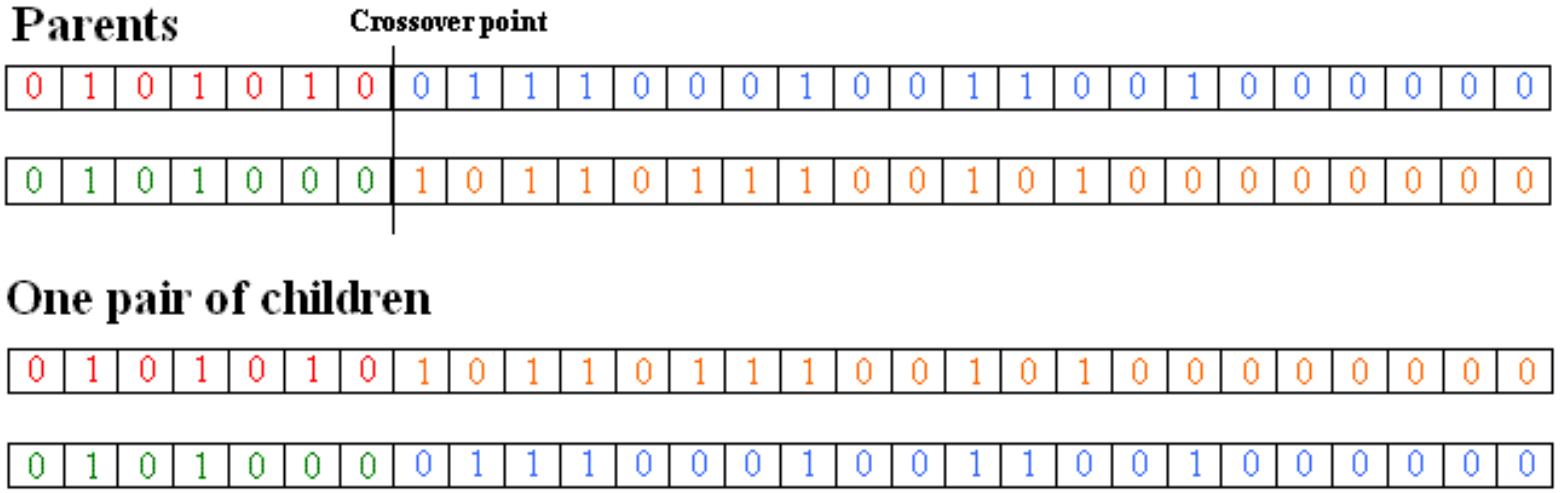
- Схема рулетки:

$$\alpha_i = 2\pi \frac{F_i}{\bar{F}}$$



Операция скрещивания

- Параметр – порог вероятности скрещивания p_{cr}
- Одноточечное скрещивание:



Операция мутации

- Параметр – порог вероятности мутации p_{mut}
- Если $p < p_{mut}$, то бит инвертируется:

$$X = (100110 \ldots 001)$$

$$p < p_{mut}$$

$$\hat{X} = (100010 \ldots 001)$$

Критерий останова

- Процесс продолжается итерационно
- Варианты критерия останова:
  - Выполнение заданного числа итераций
  - Выполнение заданного числа итераций без улучшения
  - Достижение заданного значения функции выживаемости

Схемы

- Схема (*S*) - множество строк, содержащих одинаковые биты в некоторых фиксированных позициях:

$$10 * 1 = \{1001, 1011\}$$

$$1 * 100 = \{10100, 11100\}$$

$$01 ** 0 = \{01000, 01010, 01100, 01110\}$$

Схемы

- Схема (*S*) - множество строк, содержащих одинаковые биты в некоторых фиксированных позициях:

$$10 * 1 = \{1001, 1011\}$$

$$1 * 100 = \{10100, 11100\}$$

$$01 ** 0 = \{01000, 01010, 01100, 01110\}$$

примеры схем

Схемы

- Порядок схемы (*K*)- количество фиксированных позиций в схеме:

$$K\,(10 * 1) = 3$$

$$K\,(1 * 100) = 4$$

$$K\,(01 ** 0) = 3$$

Схемы

- Определяющая длина схемы (L)– расстояние между самыми дальними фиксированными позициями:

$$S = 10 * 1 \qquad L = 3$$

$$S = 1 * 100 \qquad L = 4$$

$$S = * \ 1 * 0 * \qquad L = 2$$

Схемы

- Можно ввести понятие среднего значения $F(S,t)$ целевой функции схемы *S* на шаге t:

$$F(S,t) = \frac{1}{N_S} \sum_{s \in S} F(s)$$

Схемы

- Для любой схемы, представляющей хорошее решение, нужно, чтобы количество ее примеров увеличивалось с увеличением количества итераций

- На преобразование схем влияют операции ГА: мутация, скрещивание и селекция

Теорема схем

$$N(S,t+1) \geq N(S,t)\frac{F(S,t)}{\bar{F}(t)}\left[1 - p_c\frac{L(S)}{l-1} - p_m K(S)\right]$$

- $N(S,t)$ - количество примеров схемы S на шаге t
- $\bar{F}(t)$ - среднее значение целевой функции в популяции на шаге t
- p_c, p_m - вероятности операций мутации и скрещивания
- l– количество бит в строке

Теорема схем

$$N(S,t+1) \geq N(S,t)\frac{F(S,t)}{\bar{F}(t)}\left[1 - p_c\frac{L(S)}{l-1} - p_m K(S)\right]$$

- Ожидаемое число примеров схемы S в новой популяции

Теорема схем

$$N(S,t+1) \geq N(S,t)\frac{F(S,t)}{\bar{F}(t)}\left[1 - p_c\frac{L(S)}{l-1} - p_m K(S)\right]$$

- Вероятность выживания схемы S после выполения операций мутации и скрещивания

Теорема схем

$$N(S,t+1) \geq N(S,t)\frac{F(S,t)}{\bar{F}(t)}\left[1 - p_c\frac{L(S)}{l-1} - p_m K(S)\right]$$

- Мутация с бо̀льшей вероятностью разрушает схемы высокого порядка

Теорема схем

$$N(S,t+1) \geq N(S,t)\frac{F(S,t)}{\bar{F}(t)}\left[1 - p_c\frac{L(S)}{l-1} - p_m K(S)\right]$$

- Скрещивание с бо̀льшей вероятностью разрушает схемы с большой определяющей длиной

Теорема схем

- Проблема выбора параметров ГА является для многих приложений сложной задачей
- Теоретические результаты для решения данной проблемы на данный момент отсутствуют
- На практике данная проблема решается экспериментальным путем

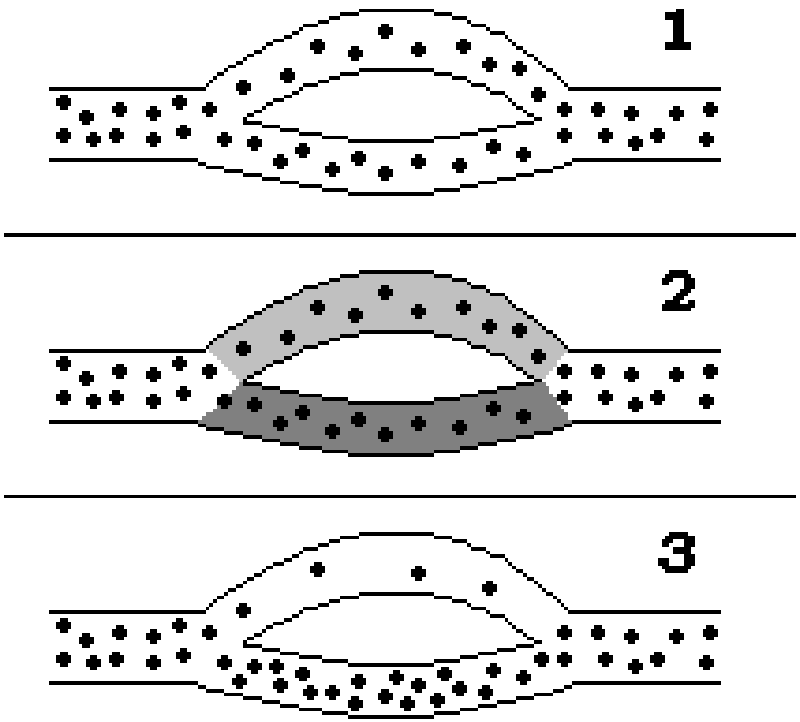
Гипотеза строительных блоков

- Строительные блоки - схемы с низким порядком, малыми определяющими длинами и большими значениями средних целевых функций.

- Гипотеза строительных блоков: комбинирование хороших строительных блоков дает хорошую строку.

Муравьиные алгоритмы (биологическая модель)

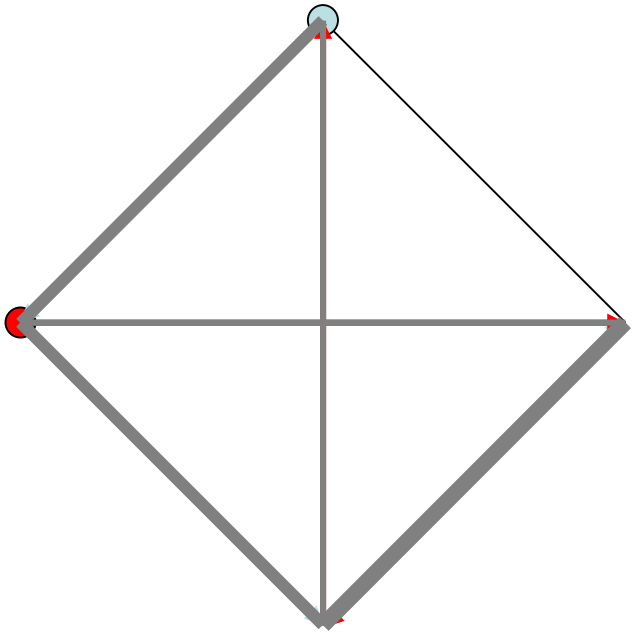
1. Изначально вероятности выбора маршрутов равны
2. Муравьи, выбравшие кратчайший маршрут, возвращаются быстрее, количество феромона на коротких маршрутах больше
3. Вероятность выбора кратчайшего маршрута повышается



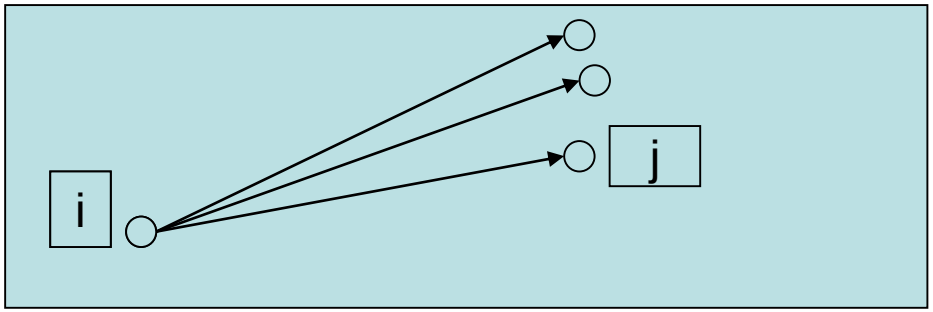
Муравьиные алгоритмы (МА для решения задачи коммивояжера)

Общая схема итерации:

- «Создание» муравьев
- Построение маршрутов муравьями
- Обновление феромонного следа на найденных маршрутах



Муравьиные алгоритмы
(построение маршрута и обновление феромонов)

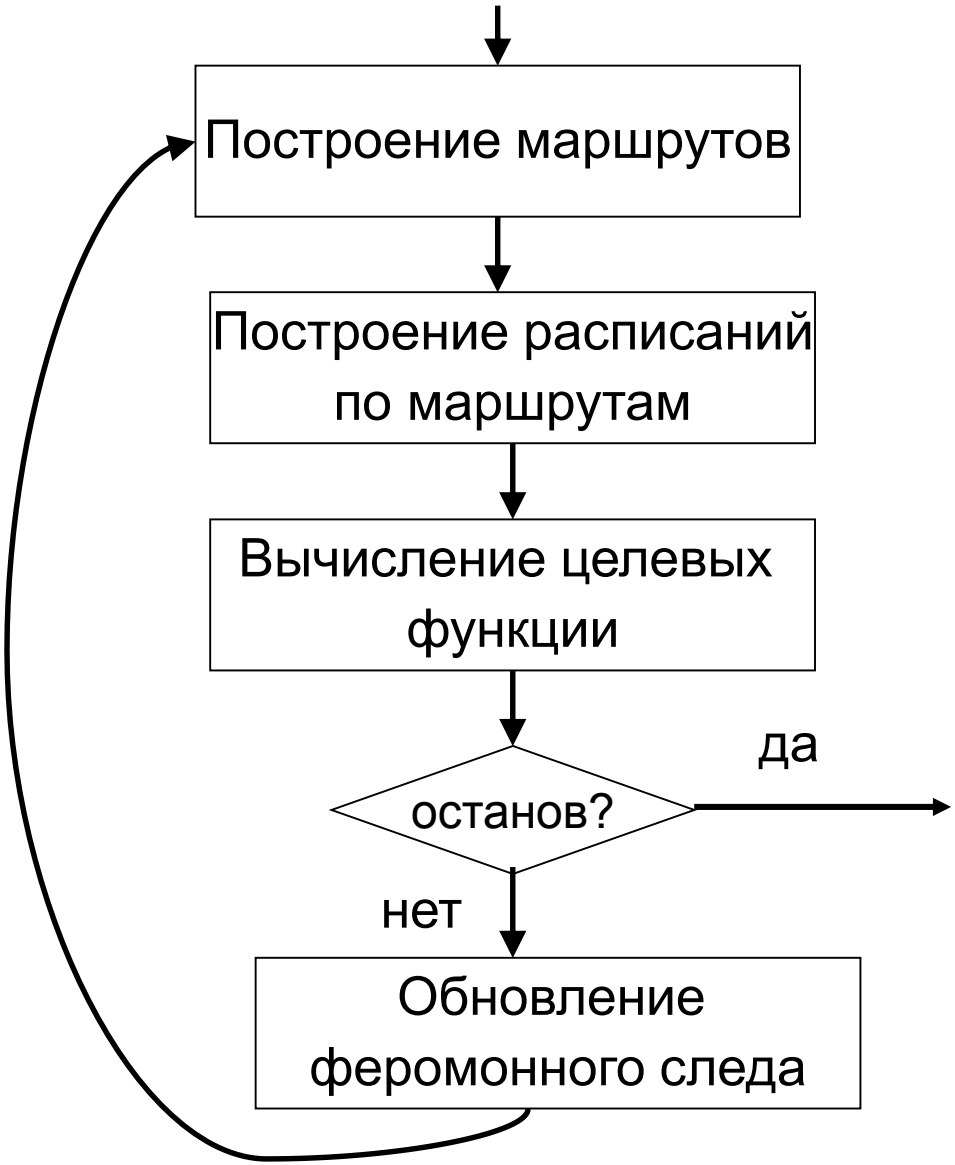


$$P_{ij,k}(t)=\begin{cases}\dfrac{(\tau_{ij}(t))^{\alpha}\cdot(\eta_{ij}(t))^{\beta}}{\sum_{l\in J_k}(\tau_{il}(t))^{\alpha}\cdot(\eta_{il}(t))^{\beta}}, & j\notin L_k\\ 0, & j\in L_k\end{cases}$$

$$\tau_{ij}(t+1)=(1-p)\cdot\tau_{ij}(t)+\sum_{k=1}^{m}\Delta\tau_{ij,k}(t)$$

$$\Delta\tau_{ij,k}(t)=\begin{cases}F(T_k(t)), (i,j)\in T_k(t)\\ 0, (i,j)\notin T_k(t)\end{cases}$$

Муравьиные алгоритмы (использование для построения расписаний)



Алгоритмы случайного поиска

- Основой методов случайного поиска служит итерационный процесс:

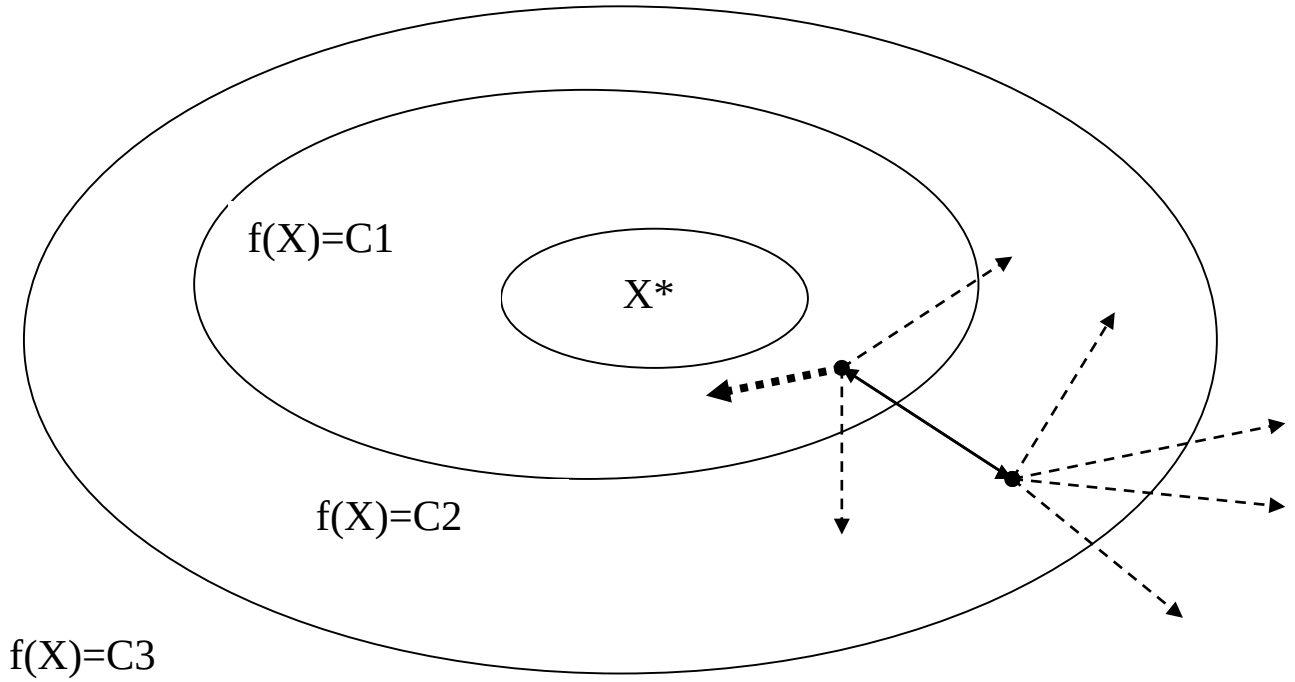
$$X_{k+1} = X_k + \alpha_k \cdot \frac{\xi}{\|\xi\|} \quad , k = 0,1,\ldots,$$

αk – величина шага,

$\xi = (\xi 1, \ldots, \xi n)$ – некоторая реализация n-мерного случайного вектора ξ.

- Л.А. Растригин. Статистические методы поиска.- М.: Наука, 1968.

Алгоритмы случайного поиска



Алгоритмы случайного поиска

- алгоритм с парной пробой,
- алгоритм с возвратом при неудачном шаге,
- алгоритм наилучшей пробы,
- алгоритм с линейной экстраполяцией,
- алгоритм статистического градиента,
- алгоритмы с самообучением.

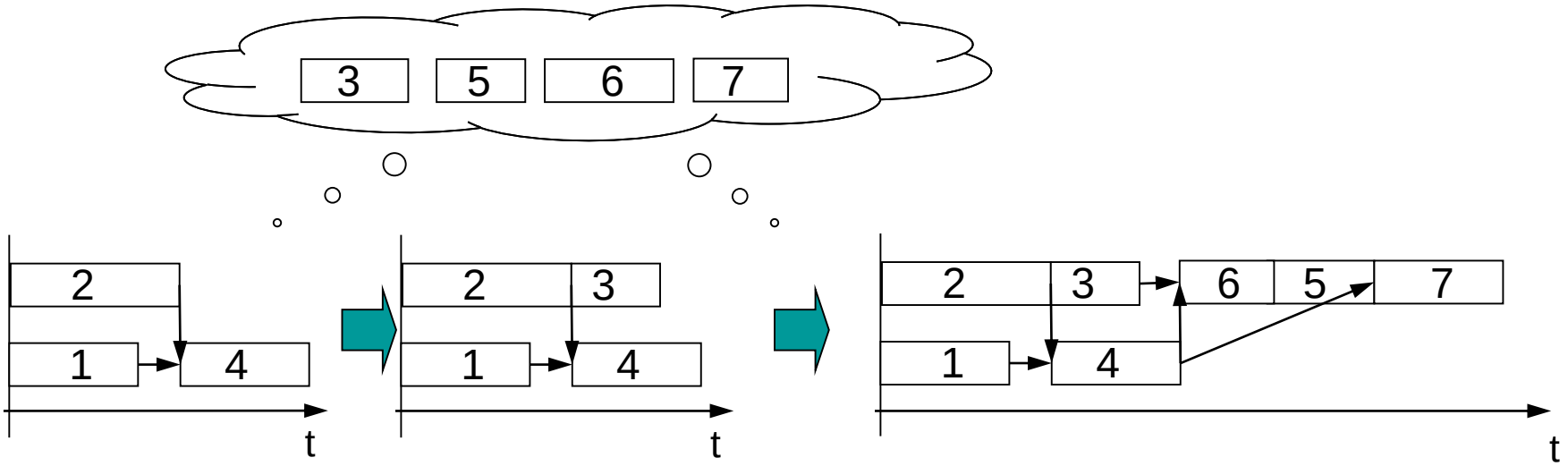
Алгоритмы случайного поиска (параметры алгоритма)

- начальный шаг $\alpha > 0$,
- коэффициент уменьшения шага $\gamma > 1$,
- предельное число неудачных попыток K,
- параметр точности $\varepsilon > 0$,
- начальное приближение X^0

.

Записать схему алгоритма, показать завершимость и достижение параметра точности.

Конструктивные алгоритмы построения расписаний



- Жадные алгоритмы
- Метод ветвей и границ
- Алгоритмы сочетающие жадные стратегии и стратегии ограниченного перебора

Жадные алгоритмы (общая схема)

Разложимые функции:

$$\min_{(x,y)} f(x,y) = \min_{(x)} f_1(x, \min_{(y)} f_2(y))$$

1. Выбрать очередную работу ≡ переменную *x* или *y.*
2. Выбрать в соответствии с локальной функцией (f_1,f_2) место размещения работы ≡ присвоить значение переменной.

Жадные алгоритмы (построение расписания выполнения работ в одноприборном устройстве)

- Для частной задачи:

$$\max_{H \in H^{*'}} |H|$$

$$\forall\ j: t_j = f_j - s_j$$

- известен оптимальный жадный алгоритм сложности $O(n \cdot \log n)$.

Жадные алгоритмы (*GrA* - алгоритм построения расписания для одноприборного устройства)

1. Упорядочиваем работы по возрастанию f_j. Заявки с одинаковым значением f_j располагаем в произвольном порядке. Сложность - $O(n \cdot \log n)$.
2. Размещаем в расписание работу $j=1$.
3. Ищем первую работу для которой $s_i \leq f_j$, размещаем ее в расписание и $j=i$.
4. Шаги 2, 3 повторяем пока список не исчерпан. Количество повторов - $O(n)$.

Жадные алгоритмы (оптимальность алгоритма *GrA*)

Теорема. Алгоритм *GrA* включает в расписание наибольшее возможное количество совместимых работ.

Доказательство.

✓Если в каком-то оптимальном расписании работа 1 отсутствует, то можно в нем заменить заявку с самым меньшим *f* на работу 1.

✓Не нарушится совместимость работ и не изменится их количество в расписании.

Жадные алгоритмы (оптимальность алгоритма *GrA*)

✓Т.е. можно искать оптимальное расписание содержащее работу 1 → существует оптимальное расписание, начинающееся с жадного выбора.

✓Из исходного набора можно удалить все работы несовместимые с 1.

✓Задача сводится, к выбору набора работ из множества оставшихся работ, т.е. мы свели задачу к аналогичной задаче с меньшим числом работ.

✓Рассуждая по индукции, получаем, что делая на каждом шаге жадный выбор, мы придем к оптимальному расписанию.

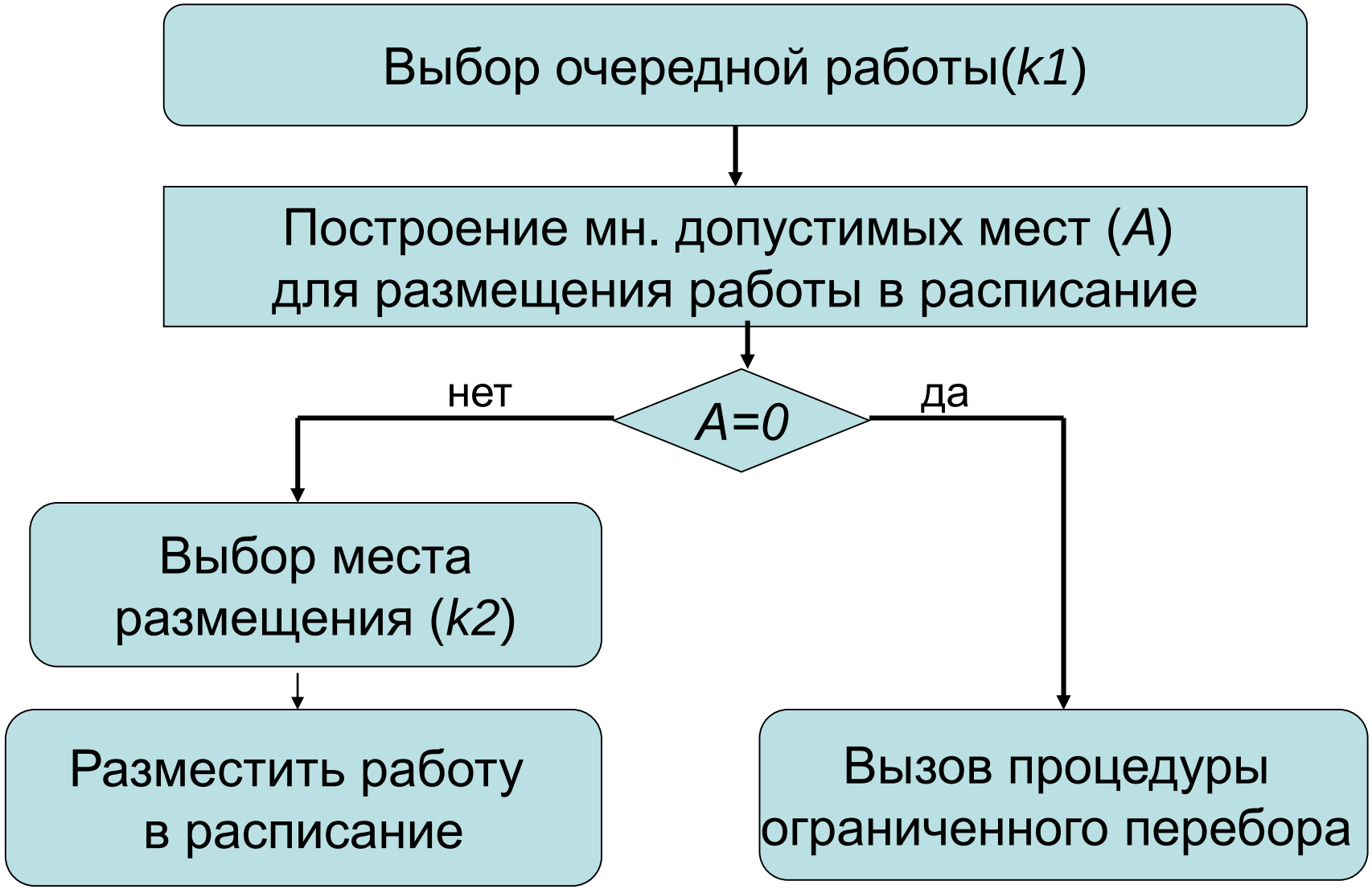
Жадные алгоритмы (как доказать, что алгоритм получает оптимальное решение)

1. Доказываем что жадный выбор на первом шаге не исключает возможности получения оптимального решения.

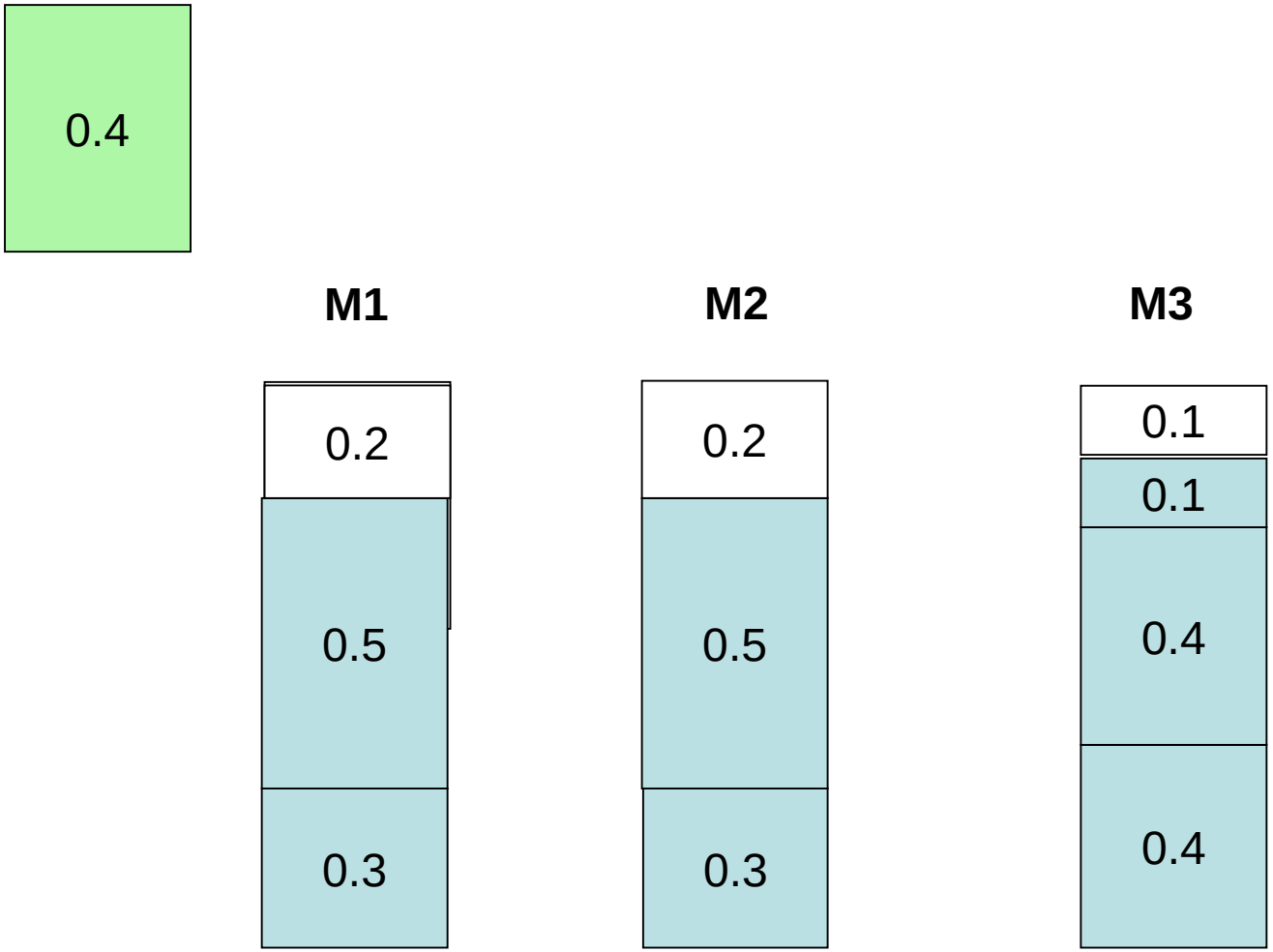
2. Показываем, что подзадача, возникающая после жадного выбора на первом шаге, аналогична исходной.

3. Рассуждение завершается по индукции.

Алгоритмы сочетающие жадные стратегии и стратегии ограниченного перебора



Пример процедуры ограниченного перебора



Материалы курса

- http://lvk.cs.msu.su/courses/
  - Презентации лекций
  - Вопросы к экзамену (скоро)
  - Оттиски статей