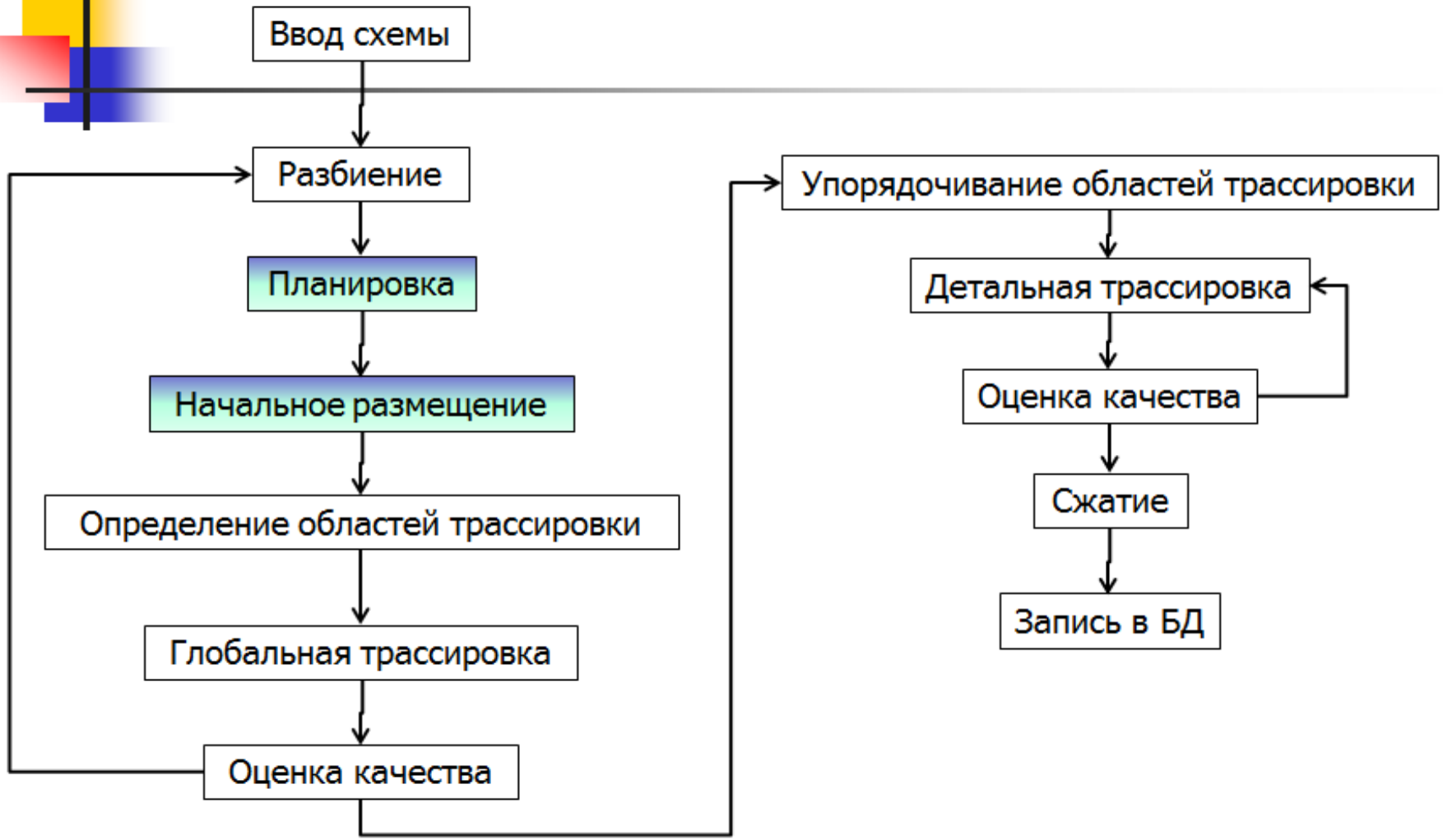
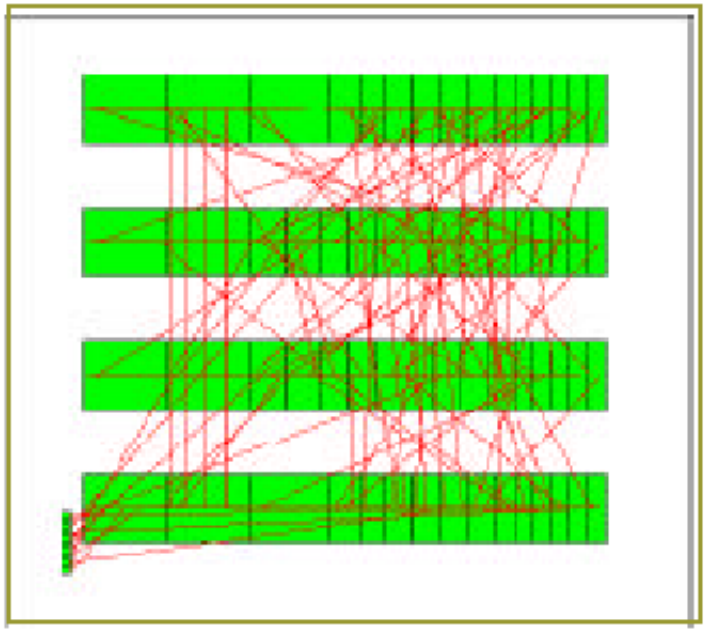
Содержание

- Сведения о КМОП технологии и методология проектирования СБИС
- Задача разбиения электрической схемы
- Задача размещения модулей СБИС
- Задача трассировки соединений

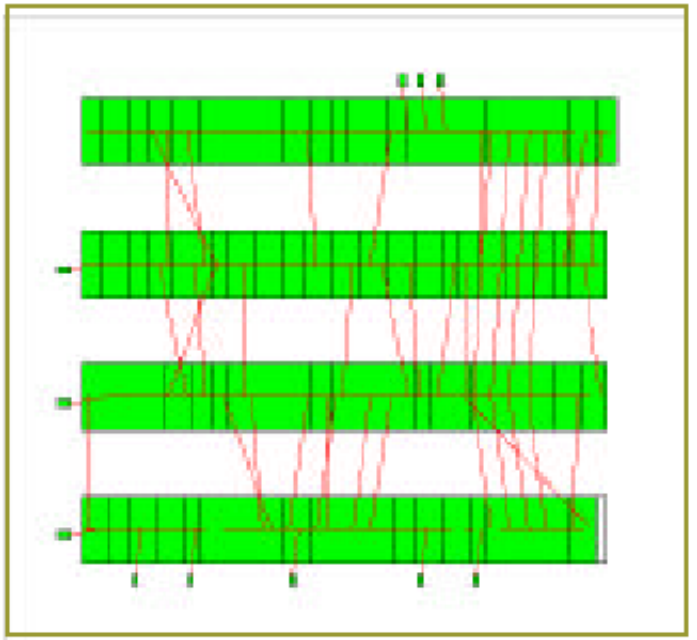
Маршрут проектирования топологии СБИС



Проблема размещения

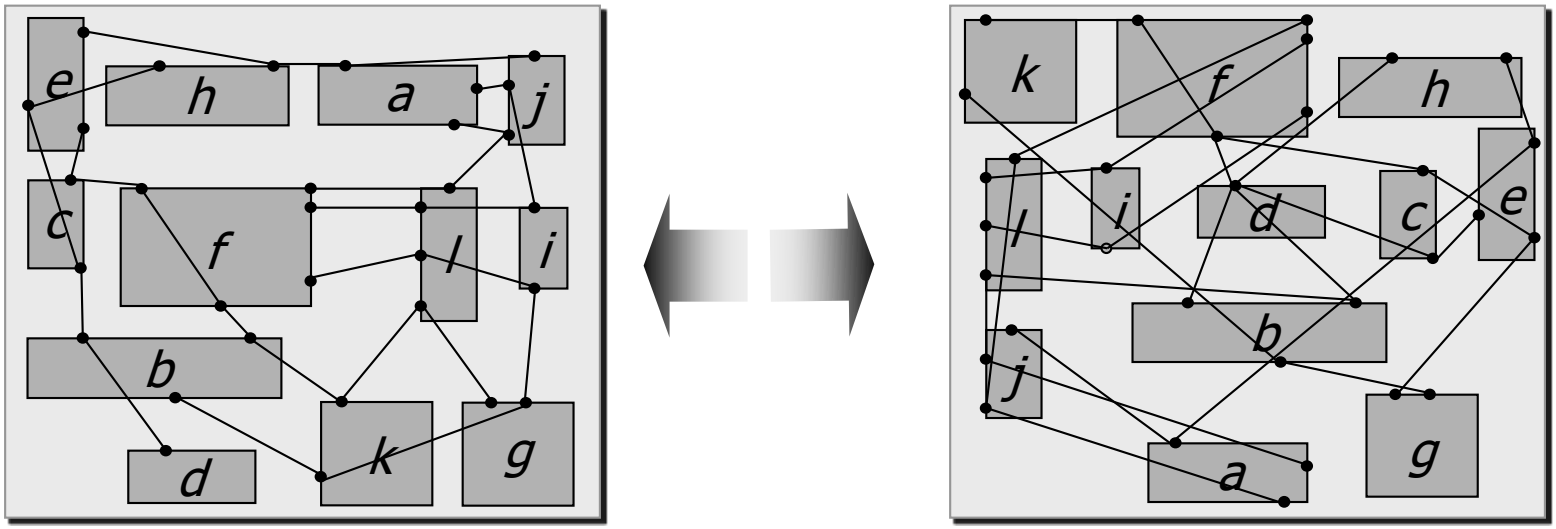


A bad placement

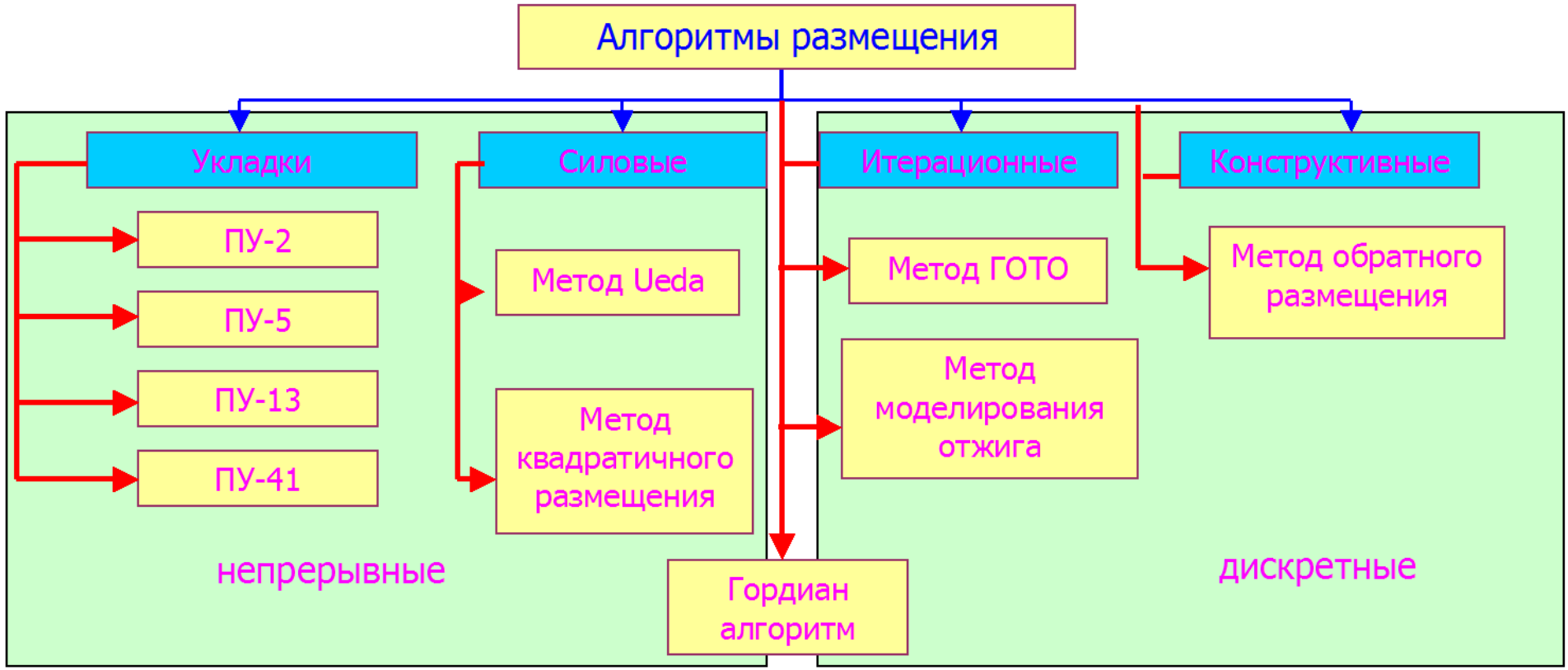


A good placement

Суммарная длина соединений (Total Wirelength)



Классификация алгоритмов размещения

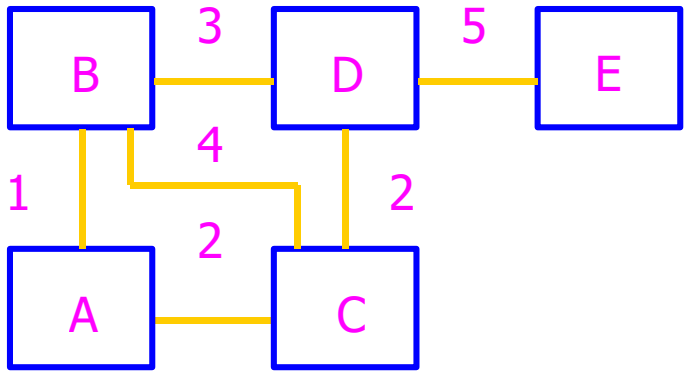


Дискретные методы размещения

- Формулировка задачи
- Обратное размещение
- Метод ветвей и границ

Метод обратного размещения

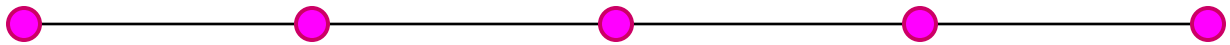
- Классификация взаимодействий между блоками



$$C = \begin{bmatrix} 0 & 1 & 2 & 0 & 0 \\ 1 & 0 & 4 & 3 & 0 \\ 2 & 4 & 0 & 2 & 0 \\ 0 & 3 & 2 & 0 & 5 \\ 0 & 0 & 0 & 5 & 0 \end{bmatrix}$$

Метод обратного размещения

- Матрица расстояний:



$$D = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 10 \\ 1 & 0 & 1 & 2 & 3 & 7 \\ 2 & 1 & 0 & 1 & 2 & 6 \\ 3 & 2 & 1 & 0 & 1 & 7 \\ 4 & 3 & 2 & 1 & 0 & 10 \end{bmatrix}$$

Суммы элементов строк

D^+

Метод обратного размещения

Алгоритм

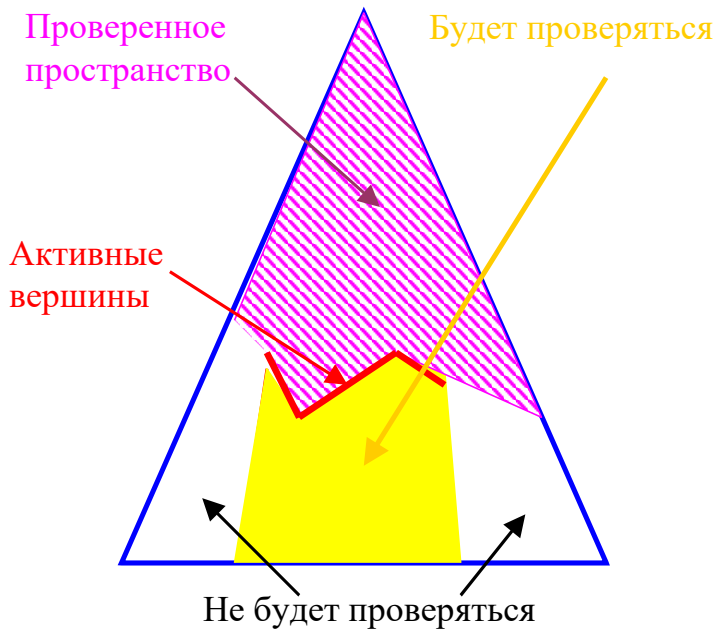
1. Упорядочить E^+ по убыванию
2. Упорядочить D^+ по убыванию
3. Разместить элемент ***i*** на позицию с индексом ***i***.

Метод ветвей и границ

- Определение: Конфигурационное пространство задачи минимизации является древовидным (tree-structured) если можно определить дерево T, с вершинами z которого ассоциируются подмножества конфигураций S_z, такие, что:
  1. Если r – корень, то S_r содержит оптимальное решение
  2. Если z – внутренняя вершина T и $z1,\ldots,zk$ – дочерние вершины z, то $\cup S_{zi} = S_z$
  3. Если z – вершина T, то существует нижняя граница $L_z \leq c(s)$ для $\forall s \in S_z$
  4. Если z – вершина T, то существует верхняя граница такая, что $U_z = \infty$ или $U_z = c(s_z)$, если z – лист, то $L_z \leq U_z$

Метод ветвей и границ (продолжение)

```
begin
  u ← ∞, CurrentBest ← ∅, вставить (r,L_r) в D
  repeat выделить следующий (z,L_z) из D
    l ← ∞
    for z′ дочерней вершины z do
      вычислить L_z′ and U_z′
      if U_z′<u then
        CurrentBest ← s_z′, u ← U_z′
        удалить все (z″,L_z″) такие, что L_z″≥u из D
      if L_z′<u then
      вставить (z′,L_z′) в D
      l ← min(l,L_z′)
  until D =∅
end
```



Непрерывные методы размещения

- Формулировка задачи
- Методы планировки
- Силовые алгоритмы

Гильотинные алгоритмы ПУ-2

1. Определение размеров поля для размещения блоков:

$$L_x = L_y = \sqrt{\sum S_i}$$

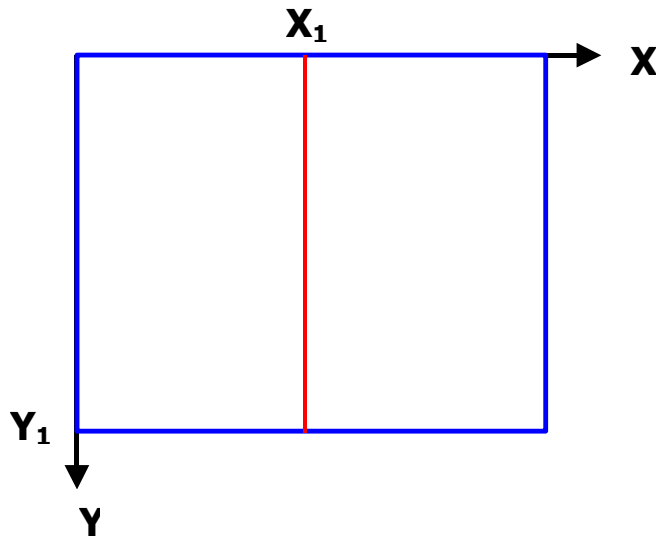
2. Собственно размещение:

$S_1 = x_1 L_y$

$S_2 = (L_x - x_1) L_y$

$L_x = L$

$0 < x_1 < L_x$



Негильотинные алгоритмы ПУ-5

$S_1=x_2y_1$

$S_2=(L_x-x_2)y_2$

$S_3=(L_x-x_1)(L_y-y_2)$

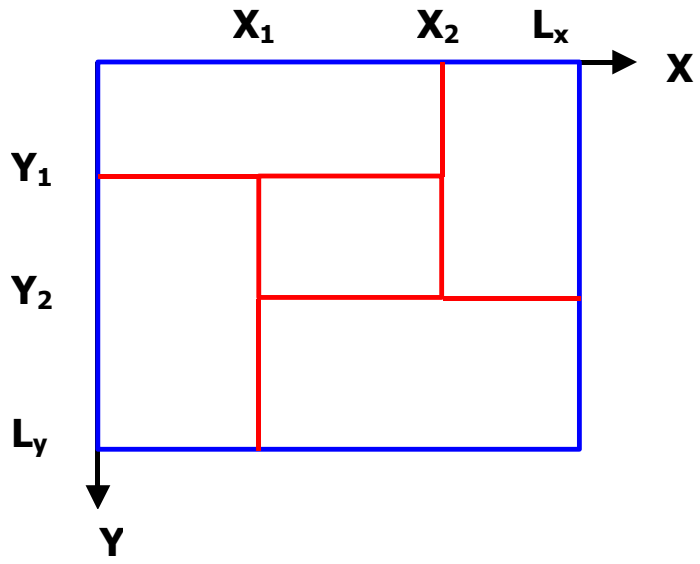
$S_4=x_1(L_y-y_1)$

$S_5=(x_2-x_1)(y_2-y_1)$

$L_x=L$

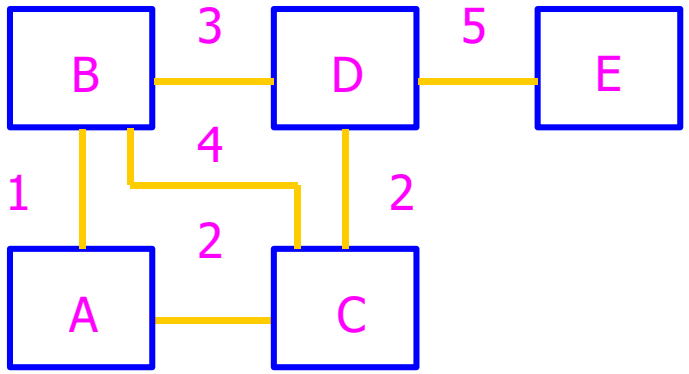
$0<x_1<x_2<L_x$

$0<y_1<y_2<L_x$



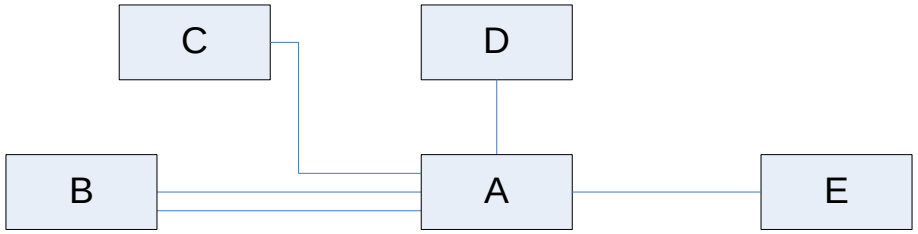
Силовой алгоритм размещения

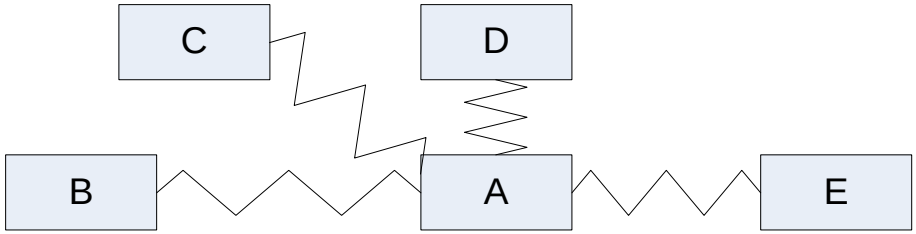
- Классификация взаимодействий между блоками

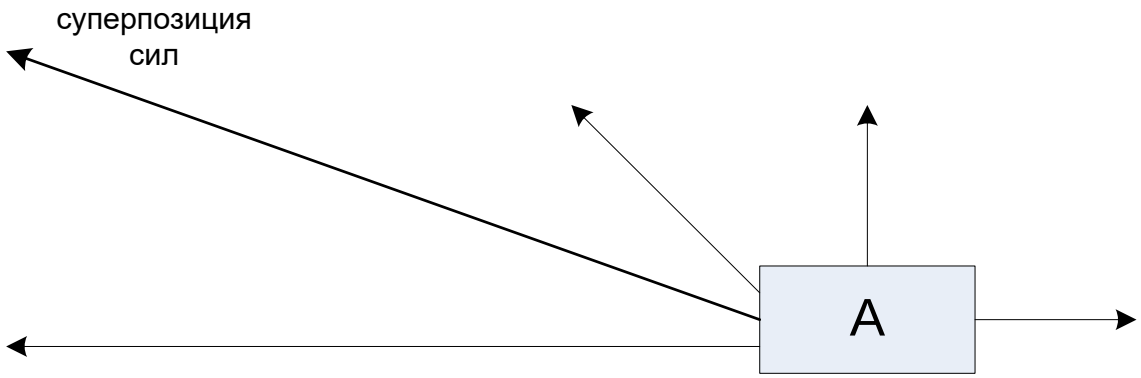


$$C = \begin{bmatrix} 0 & 1 & 2 & 0 & 0 \\ 1 & 0 & 4 & 3 & 0 \\ 2 & 4 & 0 & 2 & 0 \\ 0 & 3 & 2 & 0 & 5 \\ 0 & 0 & 0 & 5 & 0 \end{bmatrix}$$

Силовой алгоритм. Идея – суперпозиция сил.







Силовой алгоритм размещения

- Вычисление расширенной матрицы связей

$$e_{ij} = \begin{cases} \min(p_{ij}), i \neq j \\ 0, i = j \end{cases}$$

$$p_{12}(a,b) = \frac{1}{c_{ab}} = 1$$

$$p_{12}(a,b,c) = \frac{1}{c_{ac}} + \frac{1}{c_{cb}} = \frac{1}{2} + \frac{1}{4} = 0.75$$

$$p_{12}(a,c,d,b) = \frac{1}{c_{ac}} + \frac{1}{c_{cd}} + \frac{1}{c_{db}} = 1.33$$

$$E = \begin{bmatrix} 0.00 & 0.75 & 0.50 & 1.08 & 1.28 \\ 0.75 & 0.00 & 0.25 & 0.33 & 0.53 \\ 0.50 & 0.25 & 0.00 & 0.50 & 0.70 \\ 1.08 & 0.33 & 0.50 & 0.00 & 0.20 \\ 1.28 & 0.53 & 0.70 & 0.20 & 0.00 \end{bmatrix}$$

Силовой алгоритм размещения

- Вычисление матрицы отношений между блоками

$$\overline{E} = \frac{1}{n^2 - n} \sum_{i=1, i \neq j}^{n} \sum_{j=1}^{n} e_{ij}$$

$$r_{ij} = \begin{cases} e_{ij} - \overline{E}, i \neq j \\ 0, i = j \end{cases}$$

$$R = \begin{bmatrix} 0.00 & 0.14 & -0.11 & 0.47 & 0.67 \\ 0.14 & 0.00 & -0.36 & -0.28 & -0.08 \\ -0.11 & -0.36 & 0.00 & -0.11 & 0.09 \\ 0.47 & -0.28 & -0.11 & 0.00 & -0.41 \\ 0.67 & -0.08 & 0.09 & -0.41 & 0.00 \end{bmatrix}$$

Критерий сильной связности: $r_{ij} < 0$

Силовой алгоритм размещения

- Вычисление сил притяжения и отталкивания между блоками

$$F^{+} = \begin{cases} k(r-a), r > a \\ 0, 0 \le r \le a \end{cases}$$

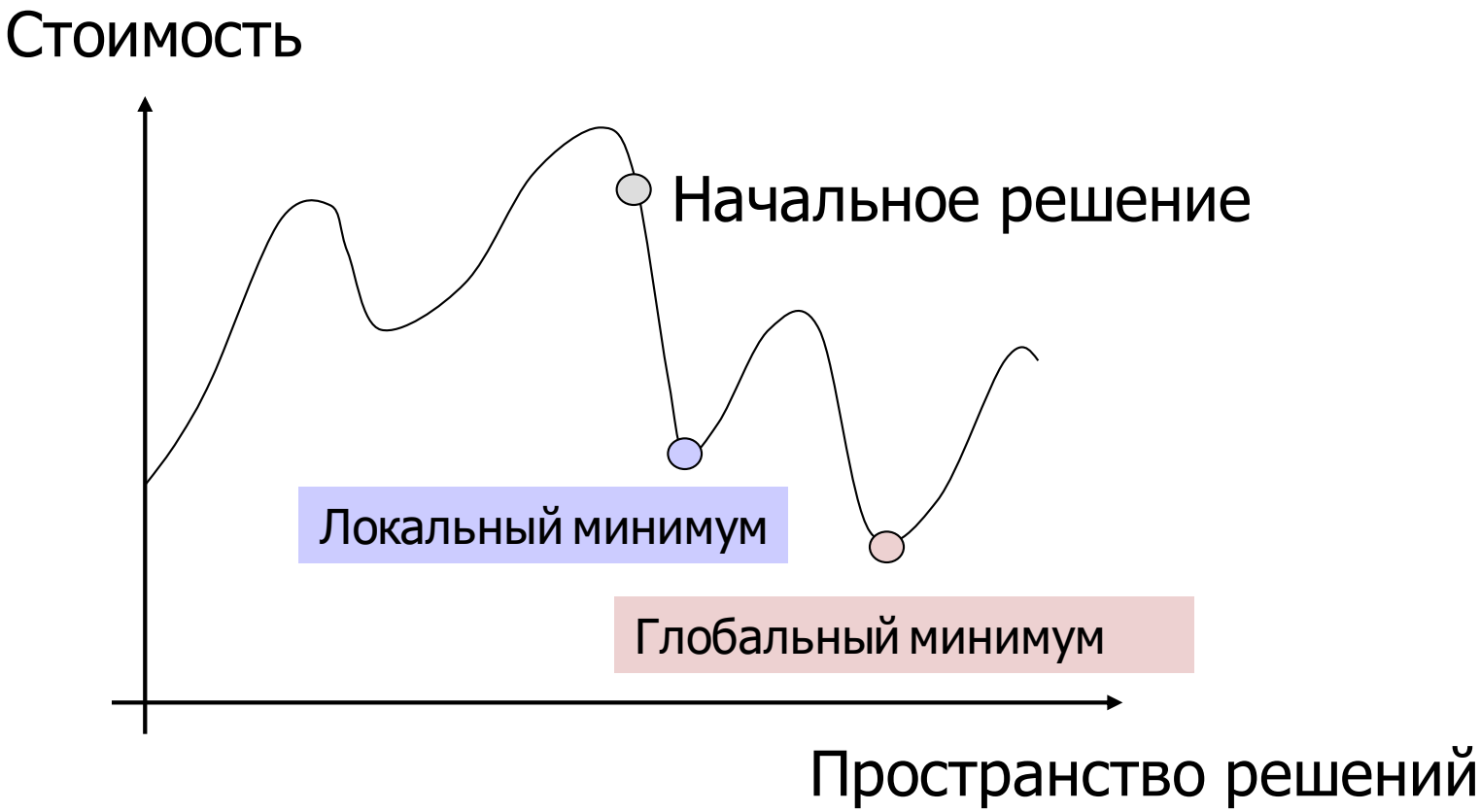
$$F^{-} = \begin{cases} -k(r-a), 0 < r < a \\ 0, r \ge a \end{cases}$$

Simulated Annealing (Моделирование отжига)

- Это итерационный алгоритм.
- В отличие от жадных алгоритмов, SA может принимать в качестве кандидата решение с худшим значением целевой функции.

21

Моделирование отжига



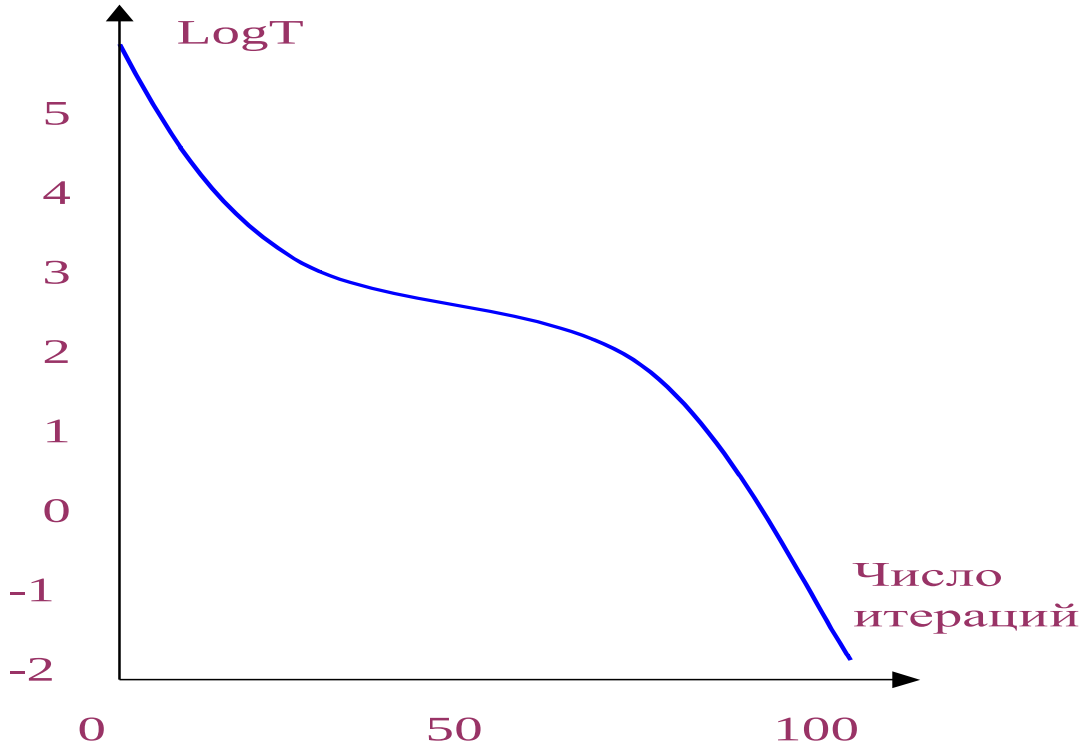
Моделирование отжига

- Generate an initial solution S_{init}, and evaluate its cost.
- Generate a new solution S_{new} by performing a random walk
- S_{new} is accepted or rejected based on the temperature T
  - Higher T means a higher probability to accept S_{new} if $COST(S_{new}) > COST(S_{init})$
  - T slowly decreases to form the final solution
- Boltzmann acceptance criterion, where r is a random number [0,1)

$$e^{\frac{COST(S_{init})-COST(S_{new})}{T}} > r$$

23

Моделирование отжига, дополнение

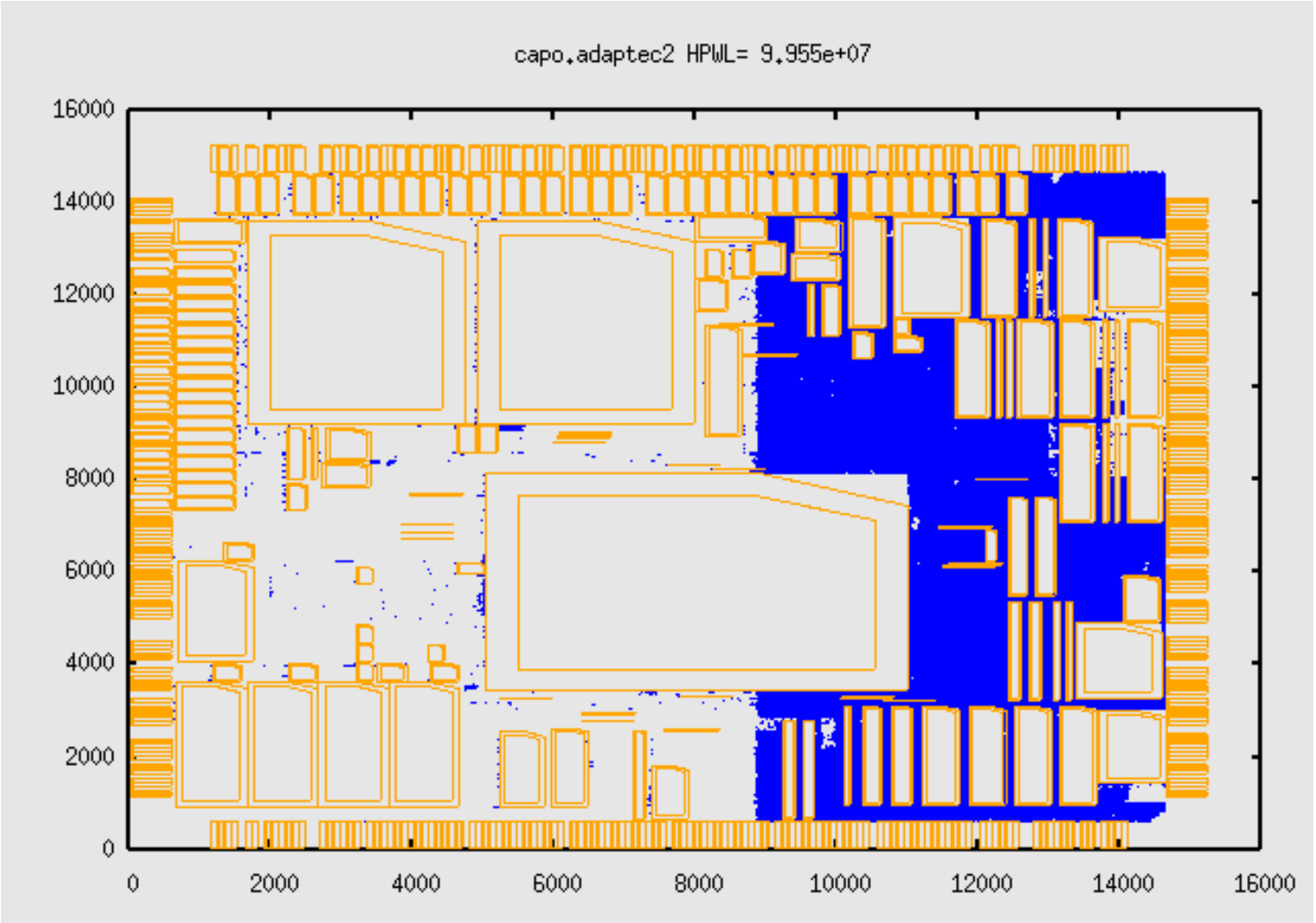


Современное состояние проблемы

- Классификация алгоритмов
- Примеры

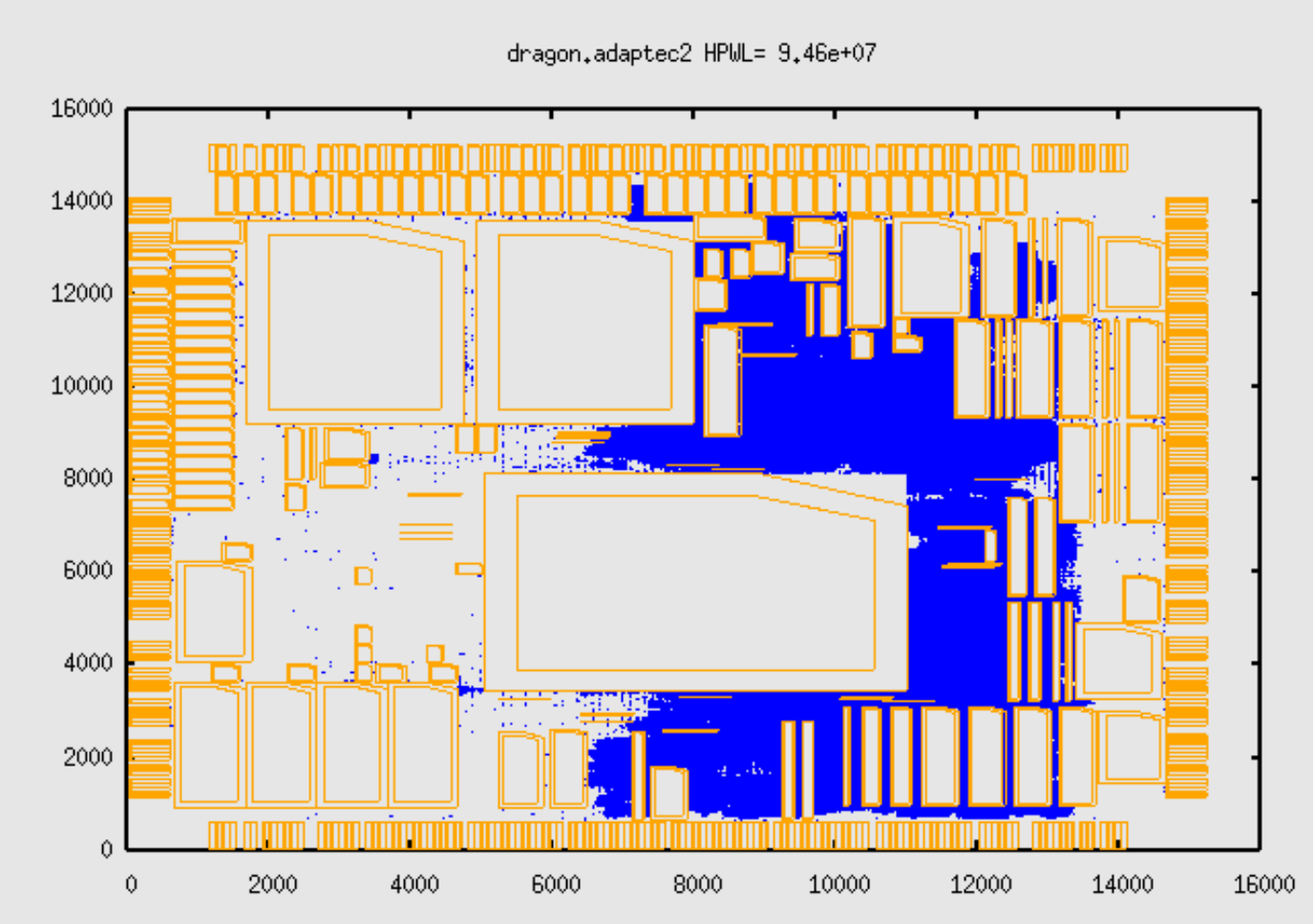
Дихотомические – Capo

- Jarrod A. Roy, David A. Papa, Saurabh N. Adya, Hayward H. Chan, Aaron N. Ng, James F. Lu, Igor L. Markov “Capo: Robust and scalable open-source min-cut floorplacer”, ISPD’05, 2005, p.224-226



Моделирования отжига – Dragon

- Taraneh Taghavi, Xiaojian Yang, Bo-Kyung Choi “Dragon2005: large-scale mixed-size placement tool”, ISPD’05, 2005, p.245-247



Dragon разработан в северо-западном университете (Northwestern University) и доступен в Интернете

- Dragon использует рекурсивное деление на четыре части с минимальным разбиением (recursive min-cut quadrisection), и затем, для детального размещения используется моделирование отжига.
- Первая стадия аналогична алгоритму используемому в Capo: на первой стадии схема и площадь кристалла разбиваются на четыре части, каждая из которых потом также разбивается на четыре части, и т.д. до тех пор, пока для полученных малых частей нельзя будет решить задачу точно

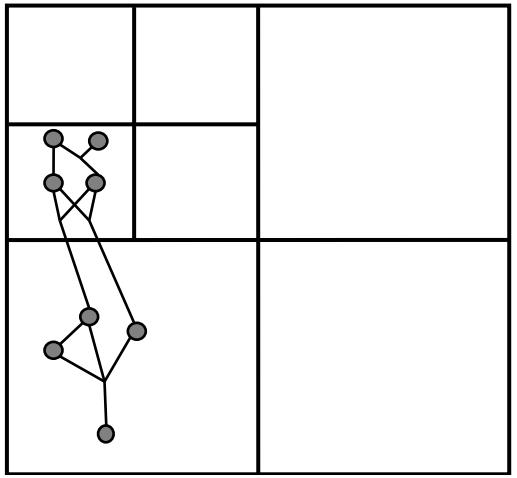


Рис. : Рекурсивное разбиение с минимальным пересечением (Dragon)