Лекции 8-9

Эйлеровы пути

- Определение 5: *Эйлеровым путем* в графе называется путь $v_1,\ldots,v_{m+1}$, такой, что каждое ребро $e \in E$ появляется в последовательности $v_1,\ldots,v_{m+1}$ в точности один раз как $e=\{v_i,v_{i+1}\}$. Если $v_1 = v_{m+1}$, то такой путь называется *эйлеровым циклом*.
- Теорема 2: Эйлеров путь в графе существует тогда и только тогда, когда граф связный и содержит не более чем две вершины нечетной степени ■.

Алгоритм нахождения эйлерова цикла

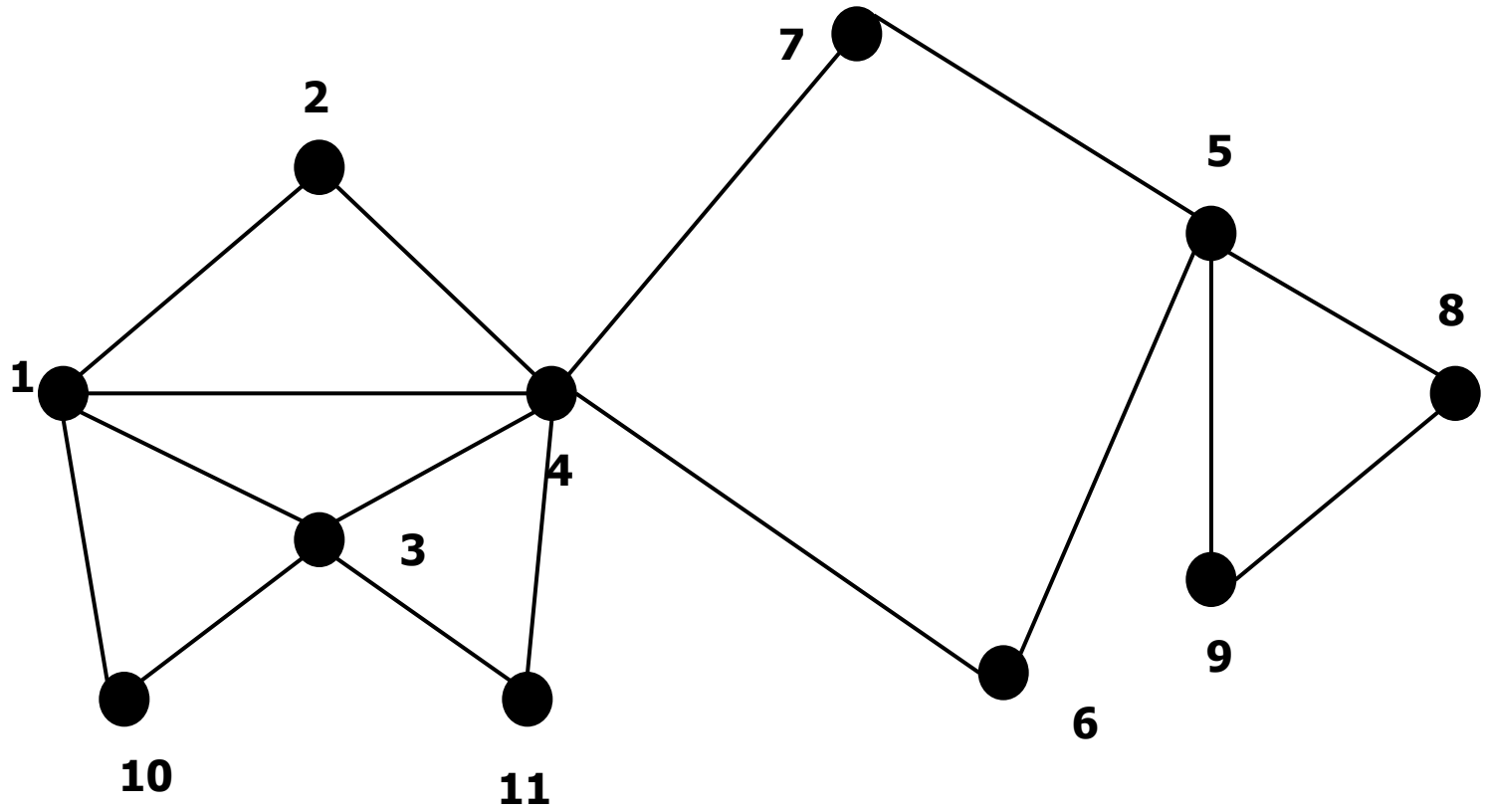
```
begin
    2.   СТЕК ← ∅; СЕ ← ∅;                //СЕ-цикл в виде последовательности вершин
    3.   v ← произвольная вершина графа
    4.   СТЕК ← v
    5.   while СТЕК ≠ ∅ do
         6.   begin v ← top(СТЕК)
         7.   if ЗАПИСЬ[v] ≠ ∅ then
              8.   begin u ← первая из ЗАПИСЬ[v]
                   9.   СТЕК ← u                    //удалить ребро из {v,u} графа
                   10.  ЗАПИСЬ[v] ← ЗАПИСЬ[v] \{u};
                   10.  ЗАПИСЬ[u] ← ЗАПИСЬ[u] \{v};
                   11.  v ← u
              12.  end
         13.  else                                 // ЗАПИСЬ[v] = ∅
         14.  begin v ← СТЕК ; СЕ ← v;
    15.  end
16.  End
end
```

СТЕК={}

СЕ={}

V=

U=



ЗАПИСЬ

1: (2,3,4,10)

2: (1,4)

3: (1,4,10,11)

4: (1,2,3,7,6,11)

5: (6,7,8,9)

6: (4,5)

7: (4,5)

8: (5,9)

9: 5,8()

10: (1,3)

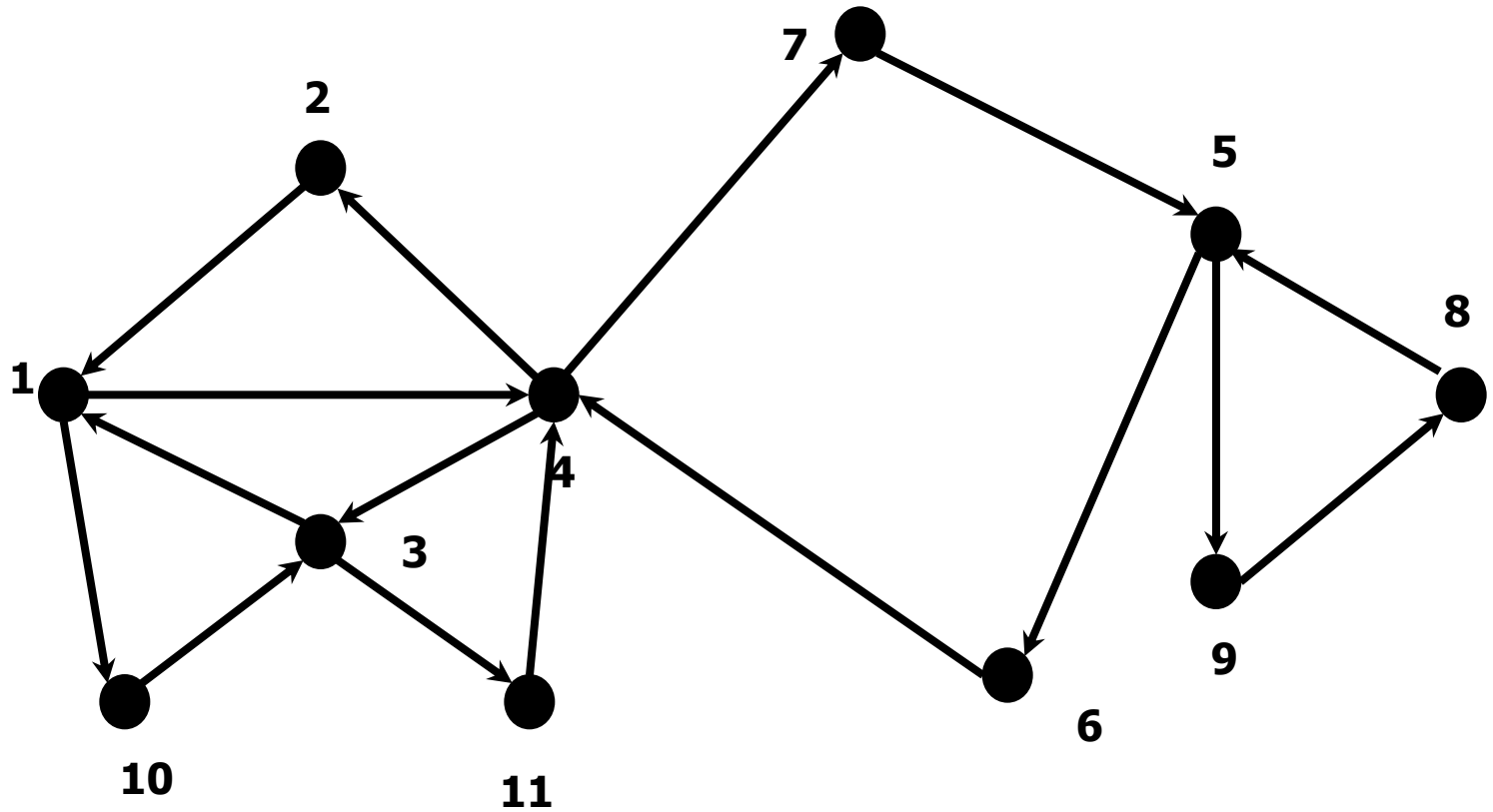
11: (3,4)

СТЕК={}

СЕ={1,10,3,11,4,3,1,4,7,5,9,8,5,6,4,2,1}

V=?

U=?



ЗАПИСЬ

1: (2,3,4,10)

2: (1,4)

3: (1,4,10,11)

4: (1,2,3,7,6,11)

5: (6,7,8,9)

6: (4,5)

7: (4,5)

8: (5,9)

9: (5,8)

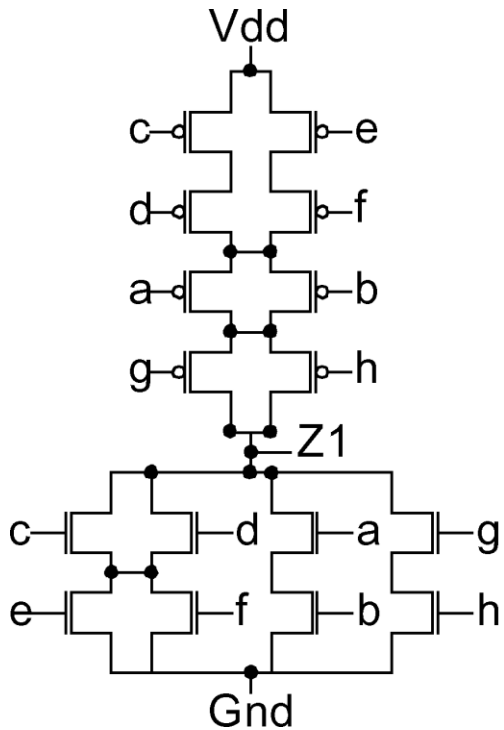
10: (1,3)

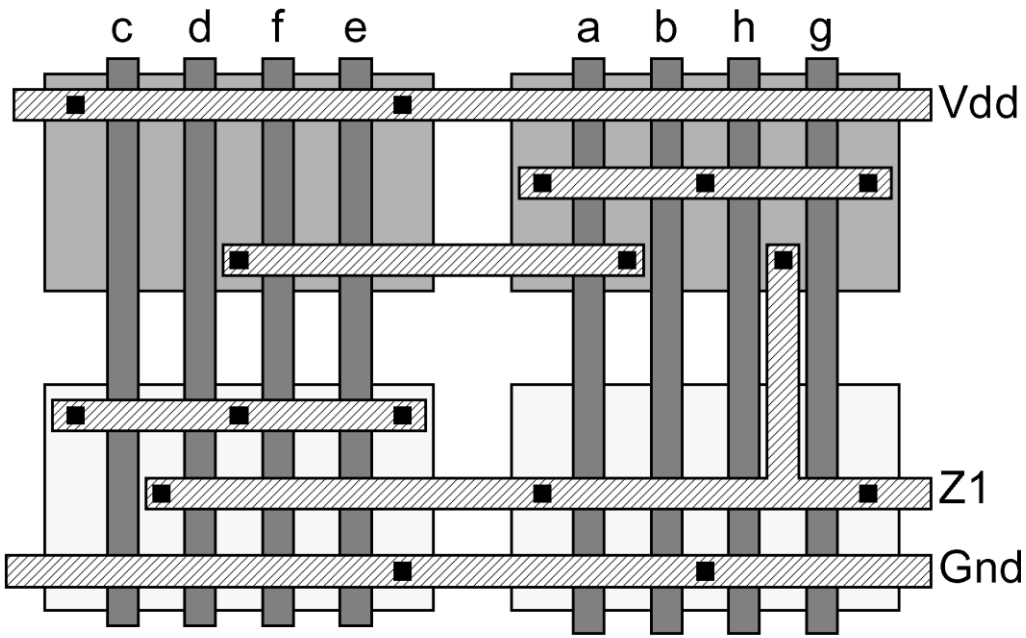
11: (3,4)

- Утверждение 1. Для последовательной цепочки транзисторов Эйлеров путь всегда существует.
- Утверждение 2. Для параллельной цепочки транзисторов Эйлеров цикл всегда существует.

Нахождение Эйлеровых путей в электрической схеме

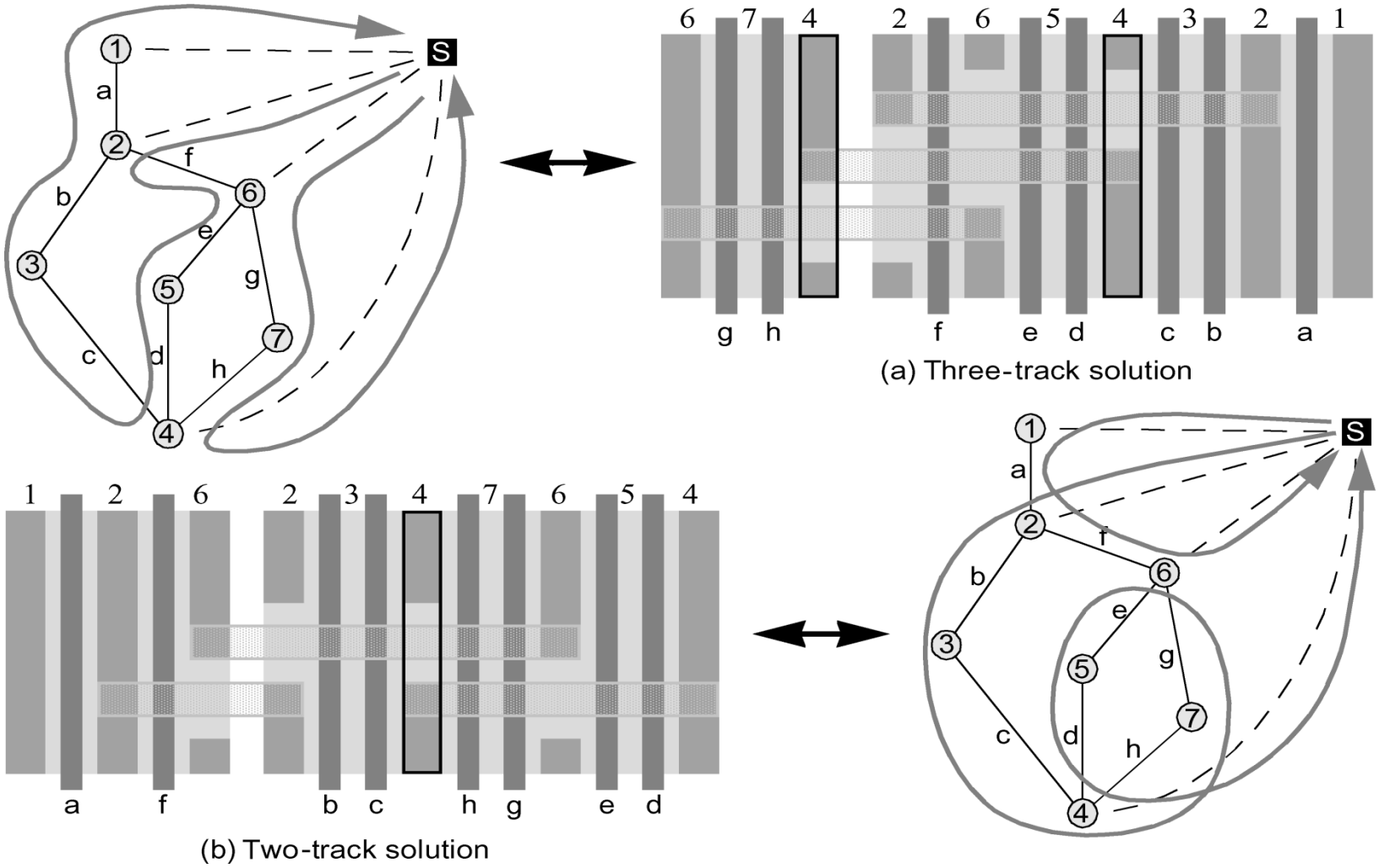
Transistor Placement for Noncomplementary Digital VLSI Cell Synthesis, MICHAEL A. RIEPE and KAREM A. SAKALLAH, ACM Transactions on Design Automation of Electronic Systems, Vol. 8, No. 1, January 2003.





An example of a complex gate designed in the “functional cell” style of Uehara and Van-Cleemput [1981]

Иллюстрация идеи



Нахождение кратчайших путей в графе

Кратчайшие пути

- Определение 7: Ориентированный граф $G=\langle V,E\rangle$ называется взвешенным, если каждой дуге $\langle u,v\rangle\in E$ поставлено в соответствие некоторое вещественное число $a(u,v)$, называемое весом данной дуги, кроме того, $a(u,v)=\infty$, если u не соединена с v. Если последовательность вершин $v_0,\ldots,v_p$ определяет путь в G, то его длина определяется как сумма $\Sigma a(v_{i-1}, v_i)$.

- Лемма 1: При условии положительной длины всех контуров $\forall\ s,t\in V\ \exists\ v : d(s,t)=d(s,v)+a(v,t)$ ■.

Алгоритм нахождения кратчайшего пути

- Данные: расстояния D[v] от вершины s до всех остальных вершин $v \in V$, фиксированная вершина t, матрица весов ребер A[u,v], $u,v \in V$.
- Результаты: СТЕК содержит последовательность вершин, определяющую кратчайший путь из s в t.

```
begin
    СТЕК ← ∅; СТЕК ← t; v ← t;
    while v ≠ s do
        begin
            u ← вершина, для которой D[v] = D[u] + A[u,v];
            СТЕК ← u; v ← u;
        end
end
```

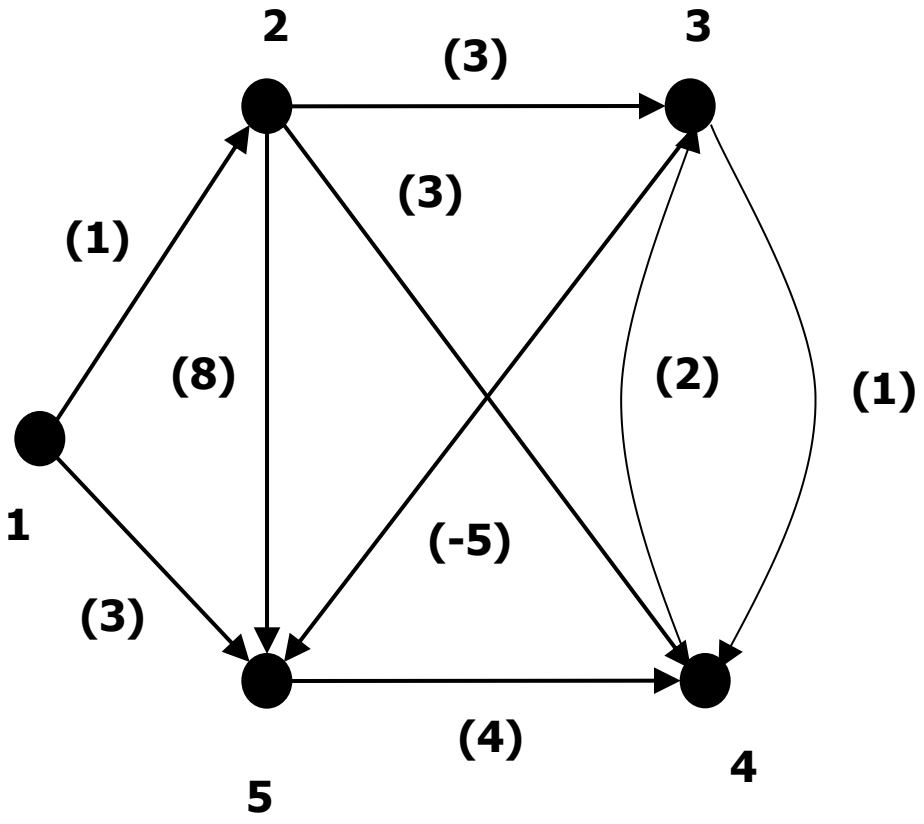
Вычисление расстояний

Алгоритм Форда и Беллмана

- Данные: орграф ⟨V,E⟩ с n вершинами и выделенным источником $s \in V$, фиксированная вершина t, матрица весов дуг A[u,v], $u,v \in V$. Контуры отрицательной длины отсутствуют.
- Результаты: расстояние от источника до всех вершин графа D[v]=d(s,v), $v \in V$.

```
begin
    for v ∈ V do D[v] ← A[s,v]; D[s] ← 0;
    for k ← 1 to n-2 do
        for v ∈ V \ {s} do
            for u ∈ V do D[v] ← min(D[v], D[u]+A[u,v])
end
```

Работа алгоритма Форда и Беллмана



∞	1	∞	∞	3
∞	∞	3	3	8
∞	∞	∞	1	-5
∞	∞	2	∞	∞
∞	∞	∞	4	∞

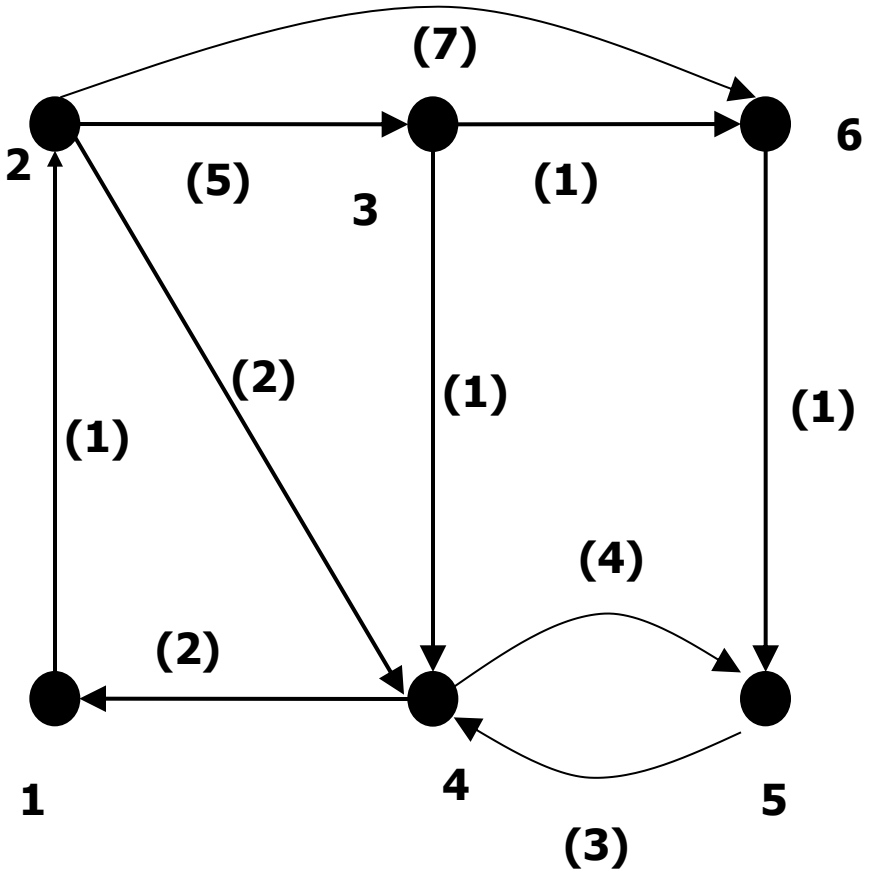
k	D[1]	D[2]	D[3]	D[4]	D[5]
	0	1	∞	∞	3
1	0	1	4	4	-1
2	0	1	4	3	-1
3	0	1	4	3	-1

Алгоритм Дейкстры

- Данные: орграф ⟨V,E⟩ с n вершинами и выделенным источником s ∈ V, фиксированная вершина t, матрица весов дуг A[u,v], u,v ∈ V. Все веса неотрицательны.
- Результаты: расстояние от источника до всех вершин графа D[v]=d(s,v), v ∈ V.

```
begin
    for v ∈ V do D[v] ← A[s,v];  D[s] ← 0;
    T ← V \ {s};
    while T ≠ ∅ do
        begin
            T ← T \ {u}                    // D[u] = min(D[p]: p ∈ T )
            for v ∈ T do D[v] ← min { D[v], D[u]+A[u,v] }
        end
end
```

Работа алгоритма Дейкстры



D[1]	D[2]	D[3]	D[4]	D[5]	D[6]
0	1	∞	∞	∞	∞
0	1	6	3	∞	8
0	1	6	3	7	8
0	1	6	3	7	7
0	1	6	3	7	7

Алгоритм Флойда

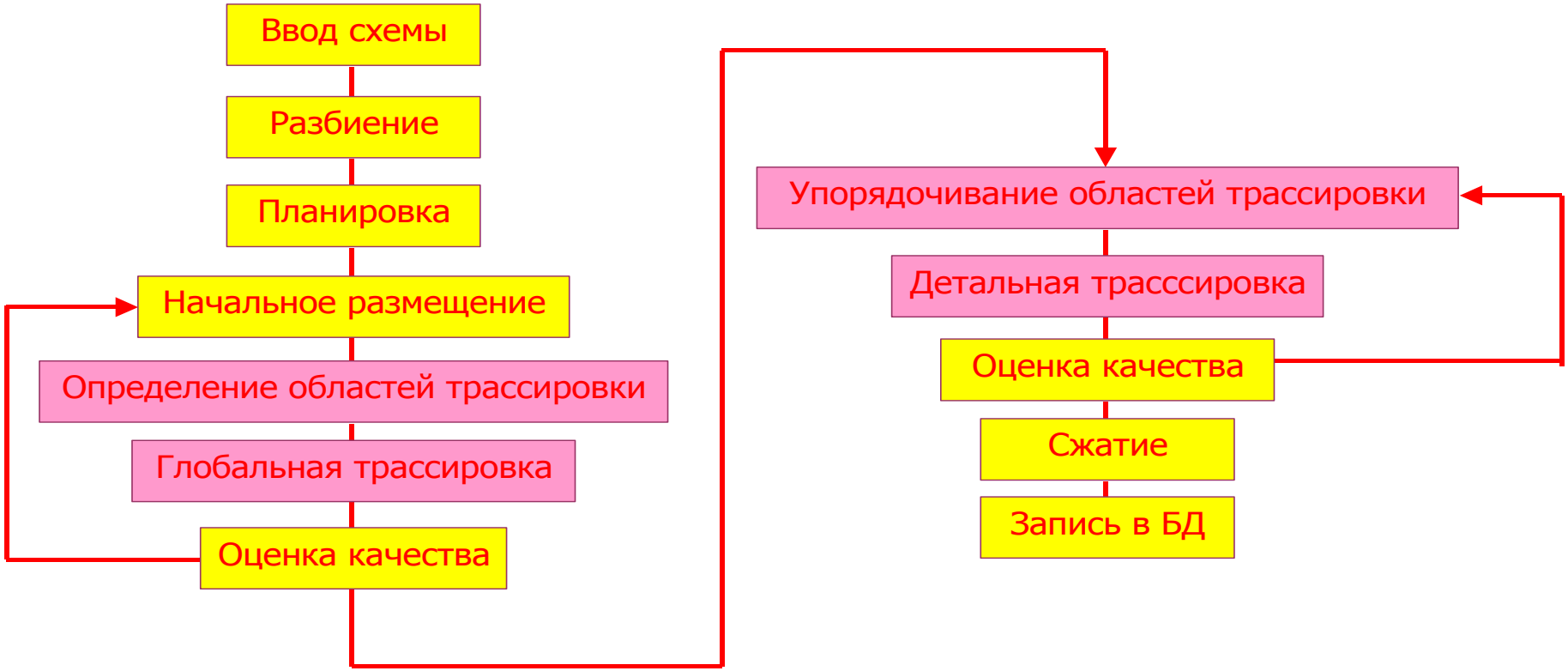
- Данные: матрица весов дуг A[i,j] орграфа без контуров отрицательной длины.
- Результаты: расстояния между всеми парами вершин $D[i,j]=d(v_i,v_j)$,

```
begin
    for i=1 to n do
        for j=1 to n do D[i,j] ← A[i,j];
    for i=1 to n do D[i,i] ← 0
    for m=1 to n do
        for i=1 to n do
            for j=1 to n do
                D[i,j] ← min(D[i,j], D[i,m]+ D[m,j] )
end
```

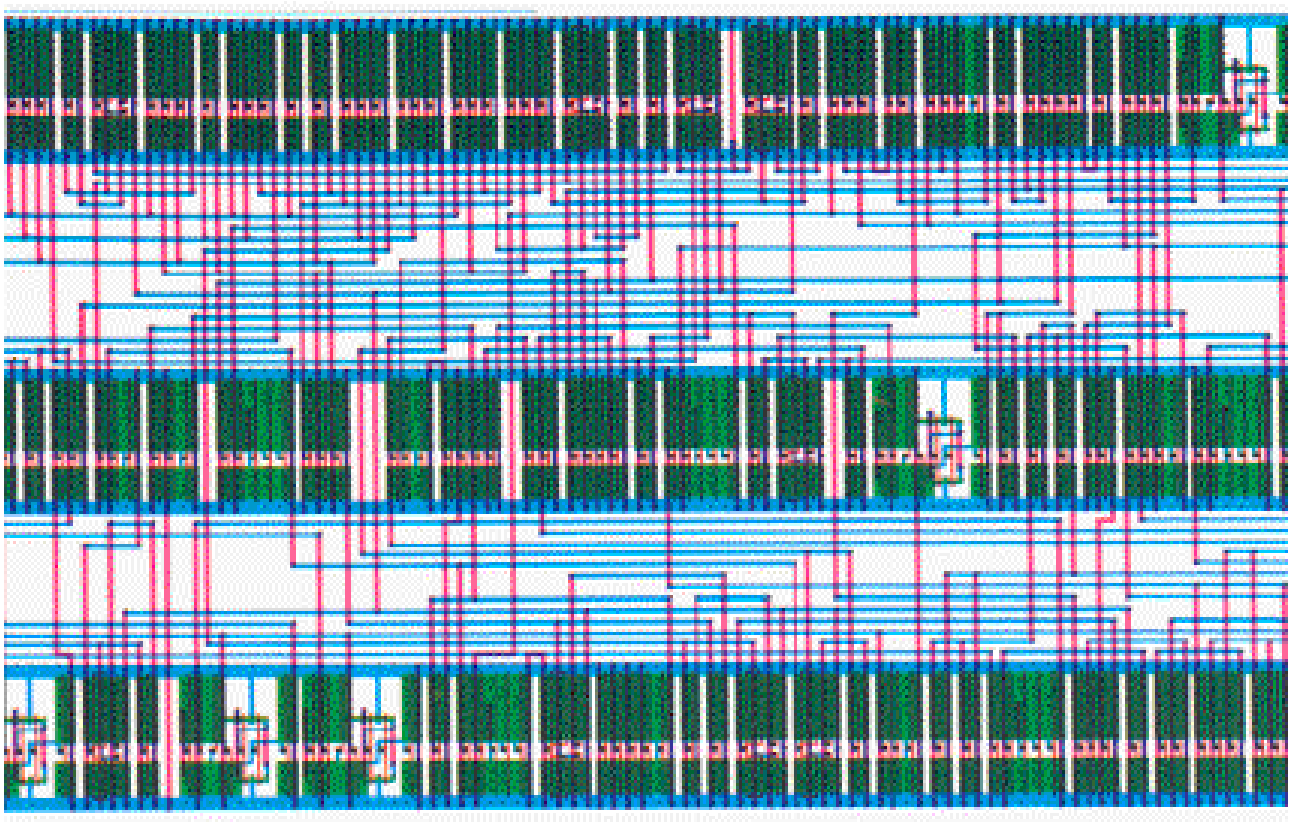
Инкрементальный алгоритм поиска кратчайших путей в графе

- Статическая задача:
  - Алгоритм Флойда-Уоршелла $O(V^3)$
  - Алгоритм Дейкстры $O(V^2)$
- Динамическая задача:
  - Нет эффективных алгоритмов, известные алгоритмы работают слишком долго

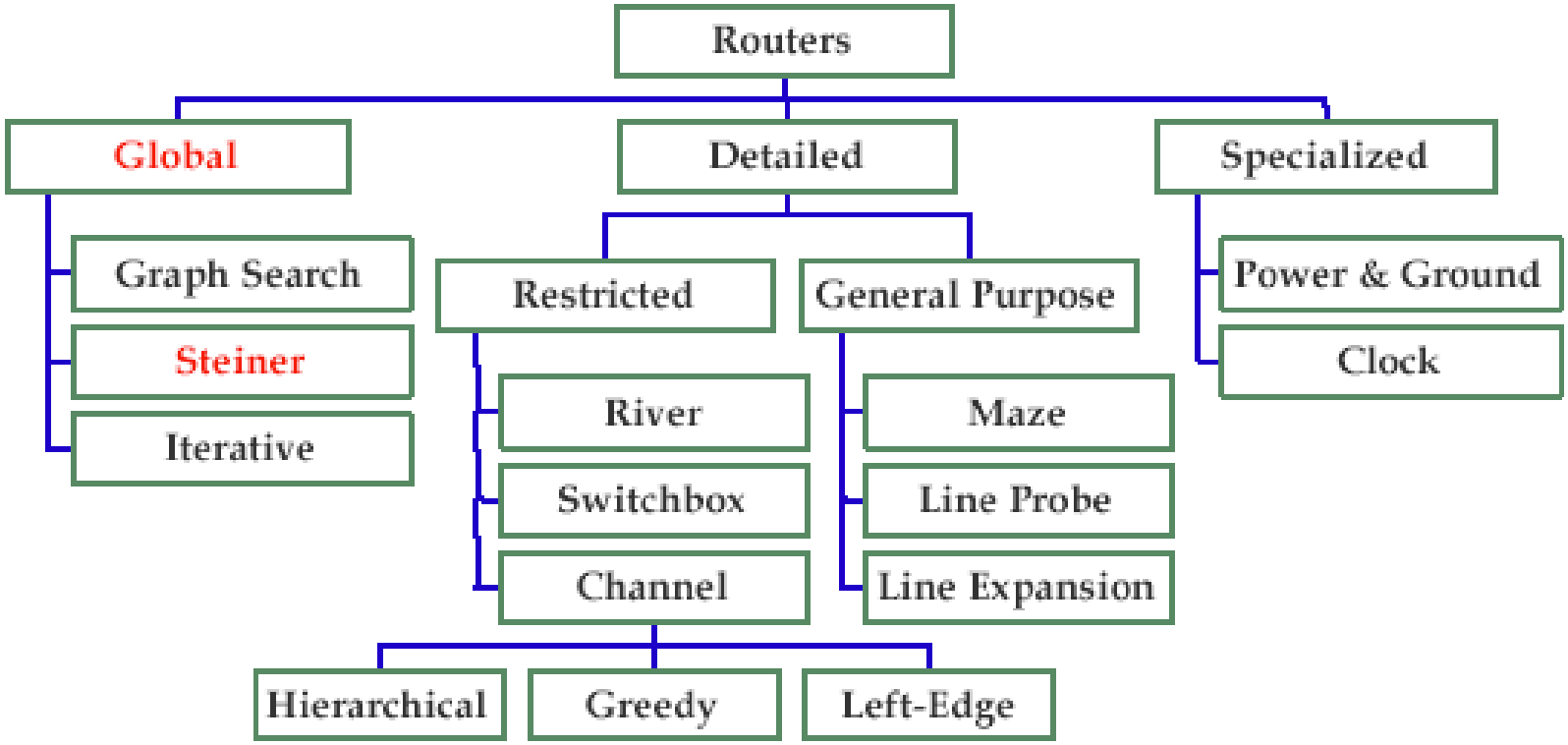
Маршрут проектирования топологии СБИС



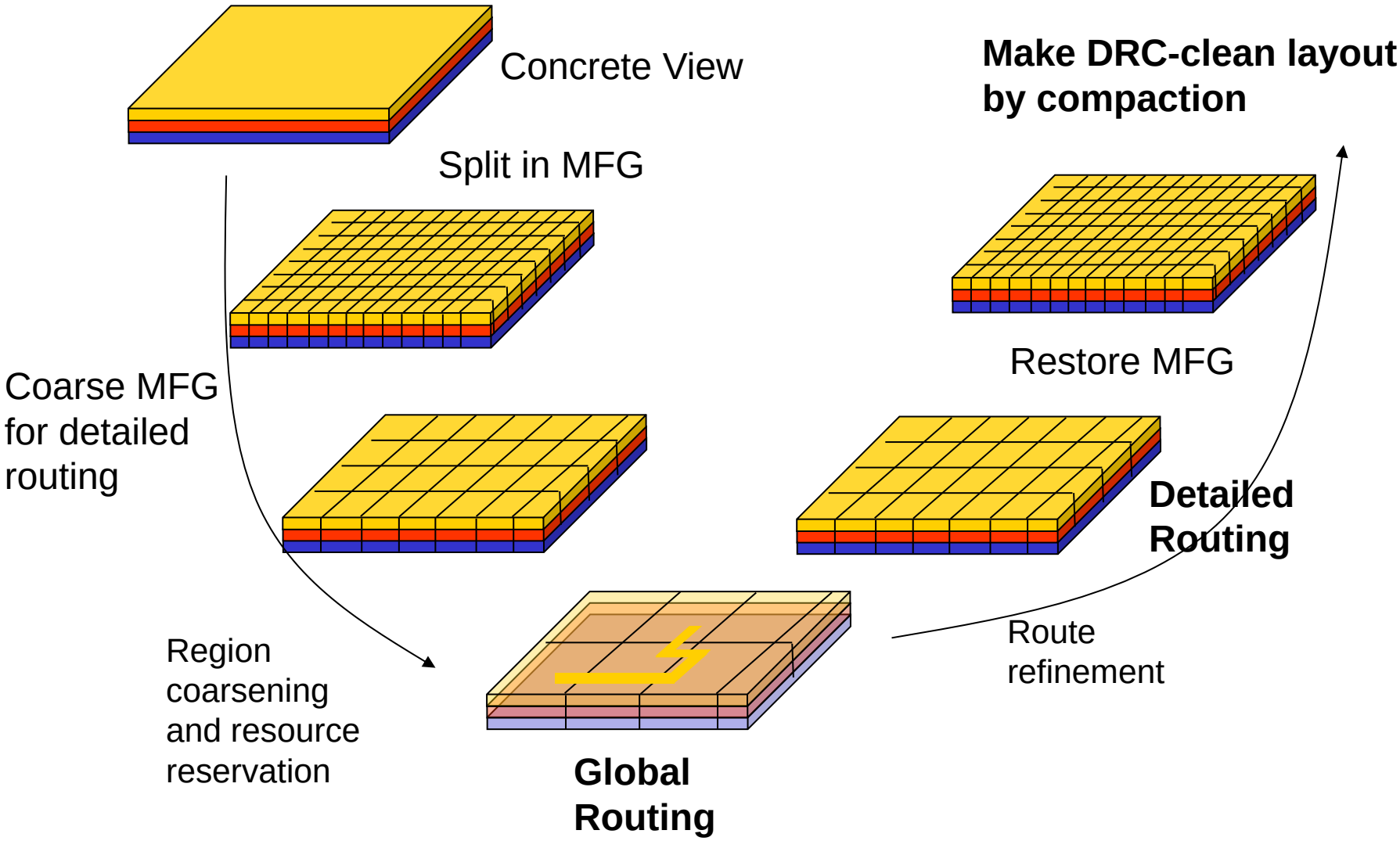
Топология на основе стандартных ячеек



Классификация алгоритмов трассировки

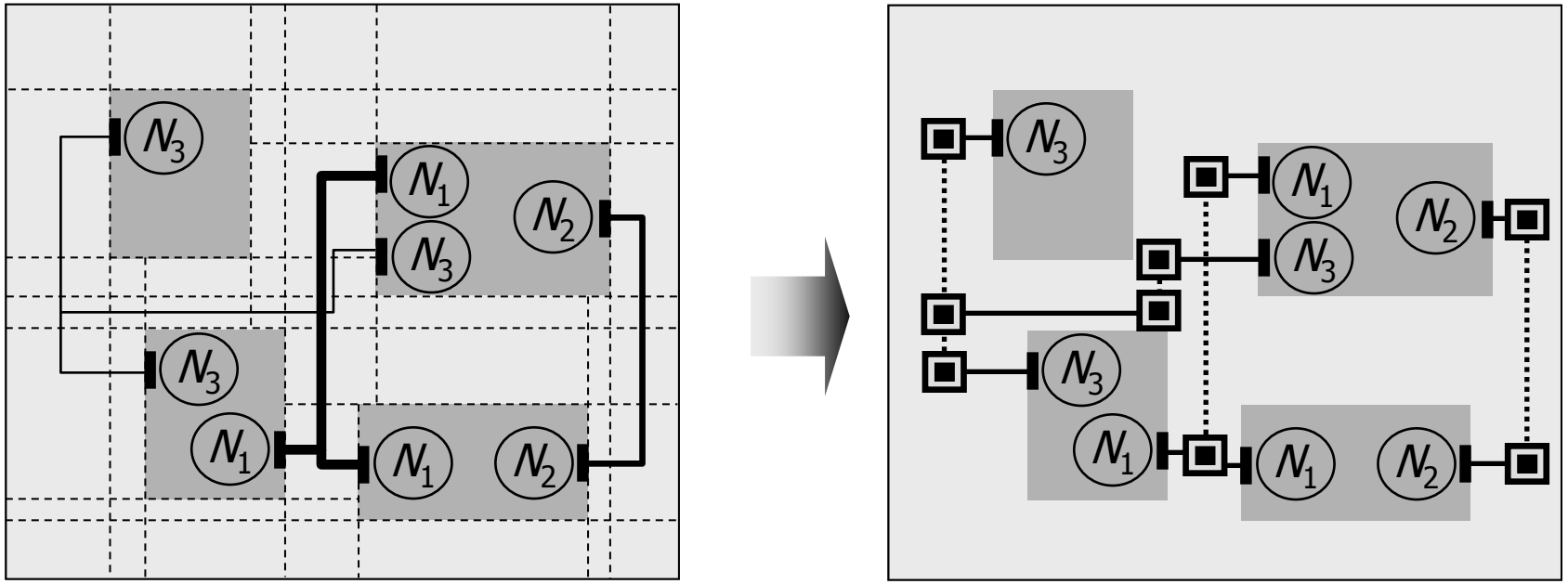


V-based methodology



Глобальная трассировка

Детальная трассировка



Horizontal Segment

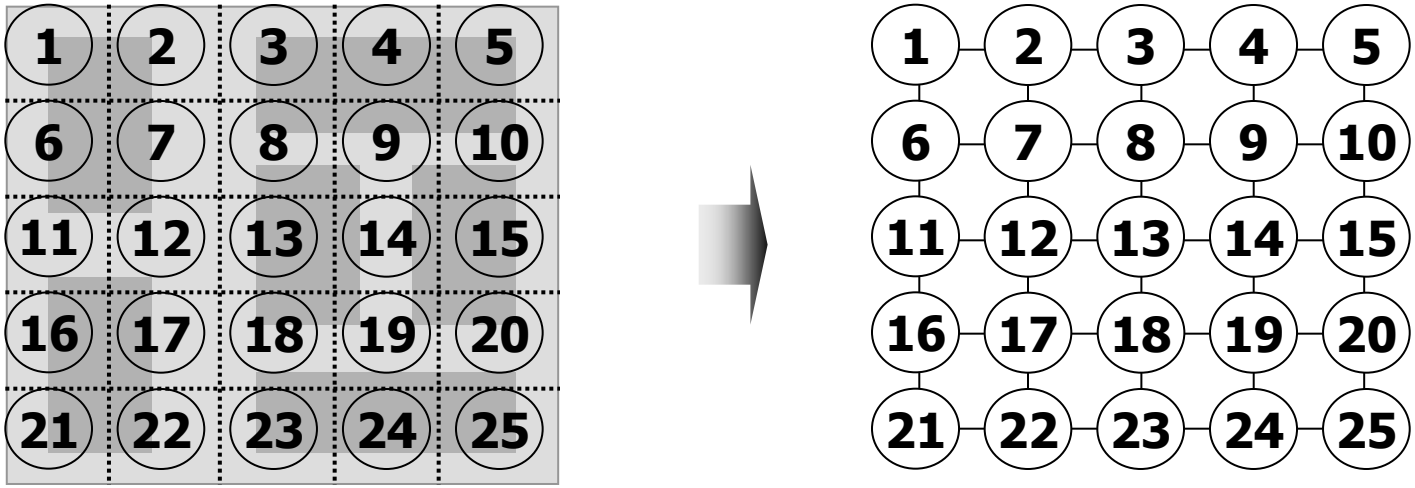
Vertical Segment

Via

*литературный источник[6] слайды 49-50

Представление областей трассировки

Граф трассировки (Grid graph model)

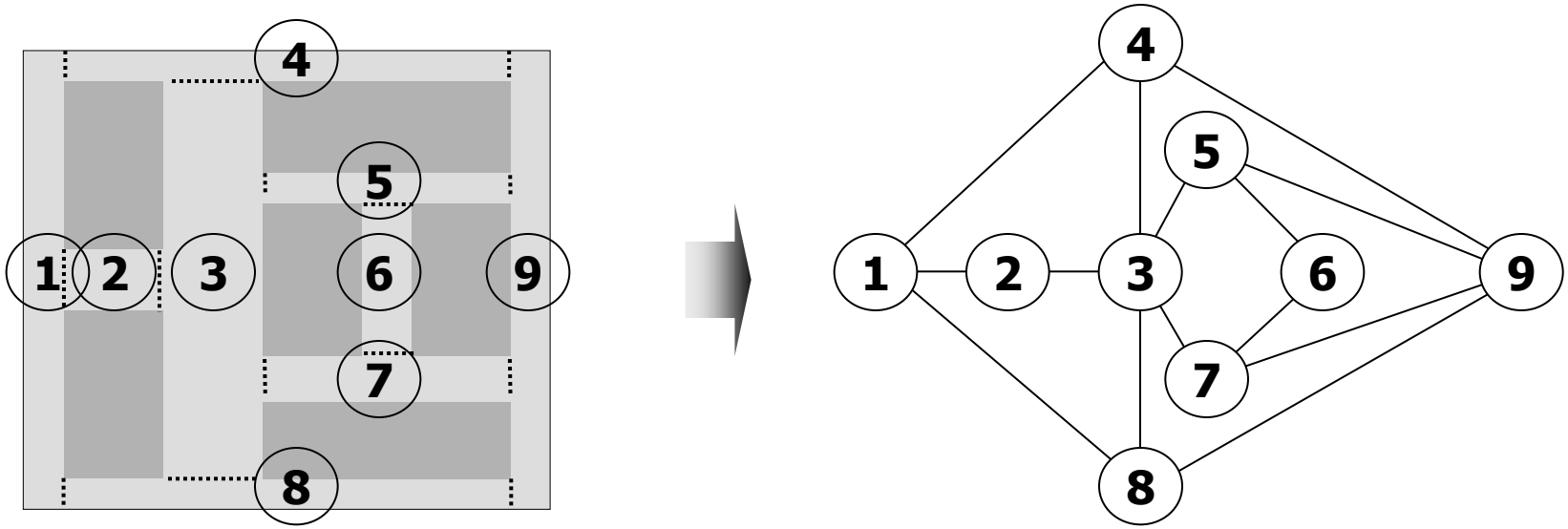


ggrid = (*V*,*E*), where the nodes $v \in V$ represent the routing grid cells (*gcells*) and the edges represent connections of grid cell pairs (v_i, v_j)

*литературный источник[6] слайды 53-55

Представление областей трассировки 2

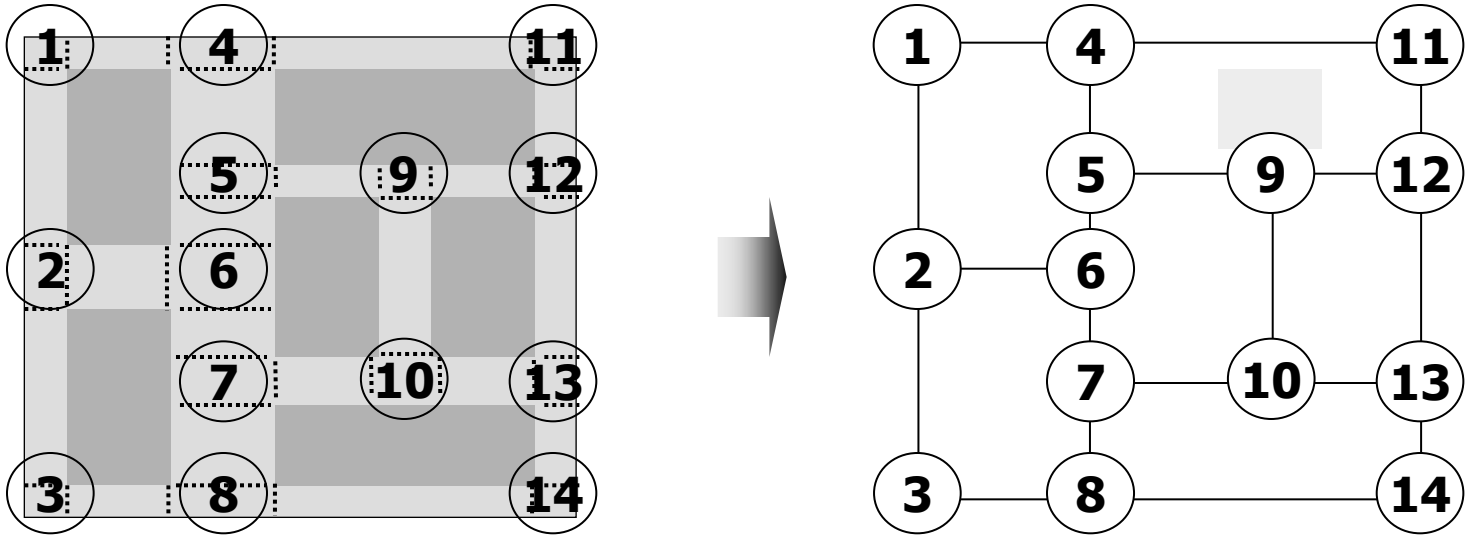
Граф связности каналов (Channel connectivity graph)



$G = (V,E)$, where the nodes $v \in V$ represent channels, and the edges E represent adjacencies of the channels

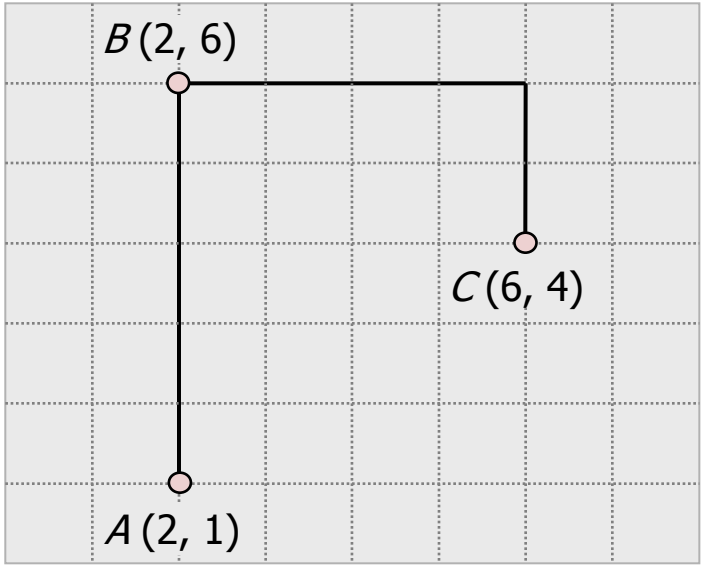
Представление областей трассировки 3

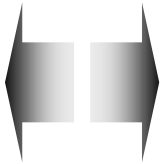
Граф связности пересений каналов (Switchbox connectivity graph)

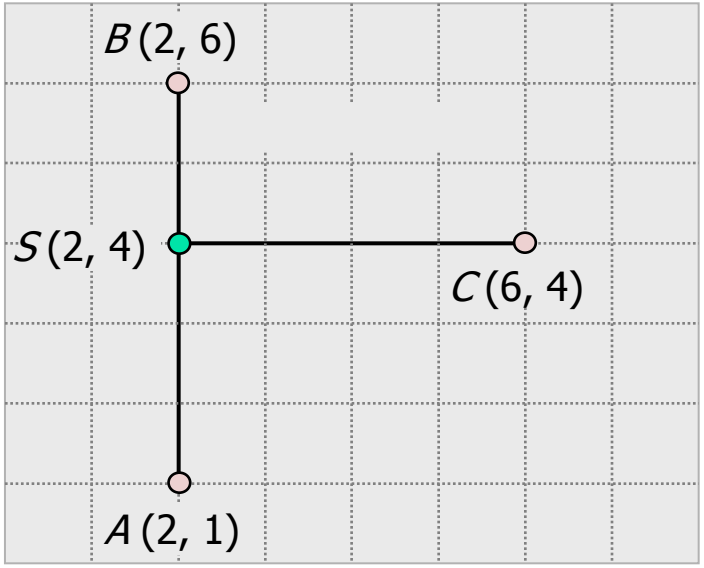


$G = (V, E)$, where the nodes $v \in V$ represent switchboxes and an edge exists between two nodes if the corresponding switchboxes are on opposite sides of the same channel

Прямоугольная трассировка (Rectilinear Routing)







Rectilinear minimum spanning tree (RMST)

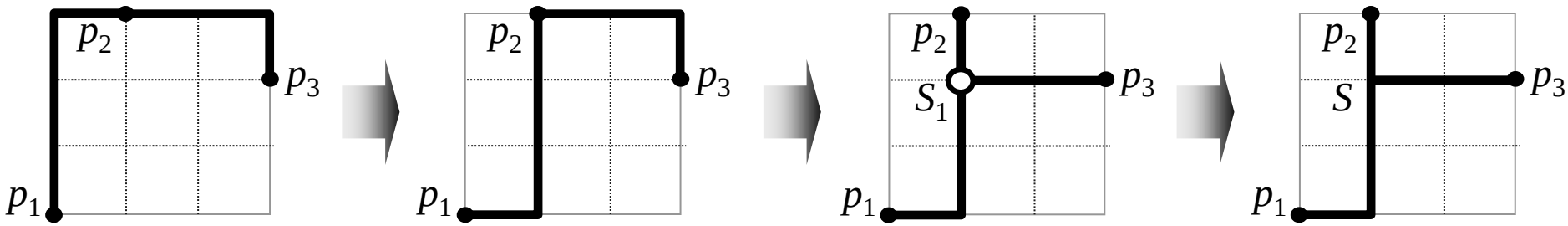
Rectilinear Steiner minimum tree (RSMT)

*литературный источник[6] слайды 59-60

Дерево Штейнера минимальной длины (SMT)

- **Теорема Ханана:** дерево Штейнера минимальной длины существует на Ортогональной Опорной Сетке (ООС) ■.
- Оценка числа точек Штейнера: **s ≤ n-2.**
- Оценка длины дерева Штейнера: **MST/SMT ≤ 3/2.**
- Простой эвристический алгоритм – использовать сканирующую линию на ООС

Преобразование связывающего дерева в дерево Штейнера



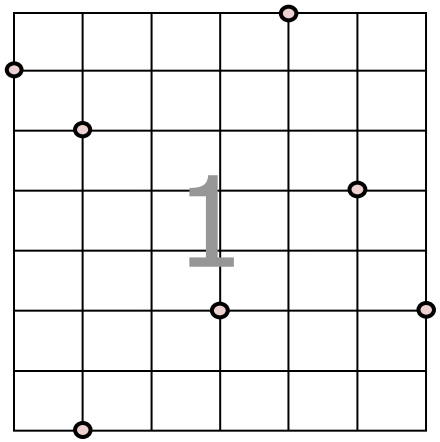
Construct *L*-shapes between points with (most) overlap of net segments

Final tree (RSMT)

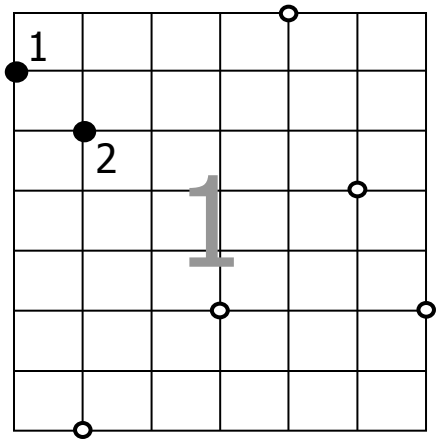
A Sequential Steiner Tree Heuristic

1. Find the closest (in terms of rectilinear distance) pin pair, construct their minimum bounding box (MBB)
2. Find the closest point pair (p_{MBB}, p_C) between any point p_{MBB} on the MBB and p_C from the set of pins to consider
3. Construct the MBB of p_{MBB} and p_C
4. Add the *L*-shape that p_{MBB} lies on to *T* (deleting the other *L*-shape). If p_{MBB} is a pin, then add any *L*-shape of the MBB to *T*.
5. Goto step 2 until the set of pins to consider is empty

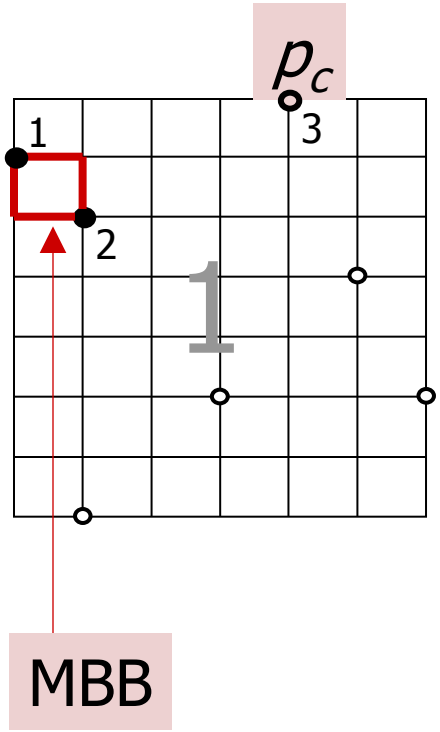
Пример. Исходные данные.



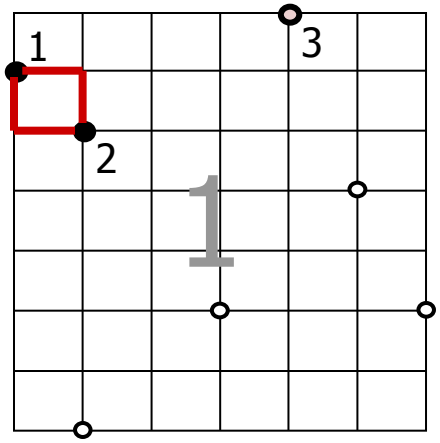
Пример. Шаг 1.

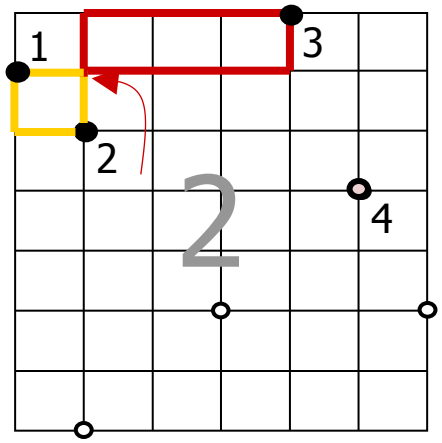


Пример. Шаг 1.



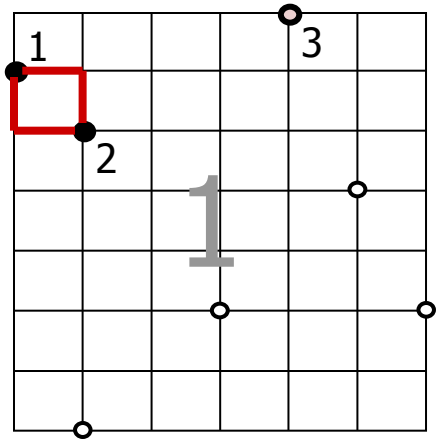
Пример. Шаг 2.

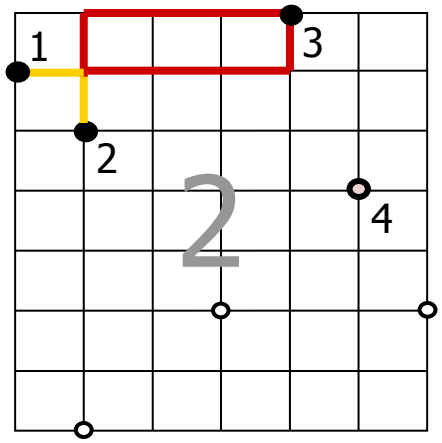


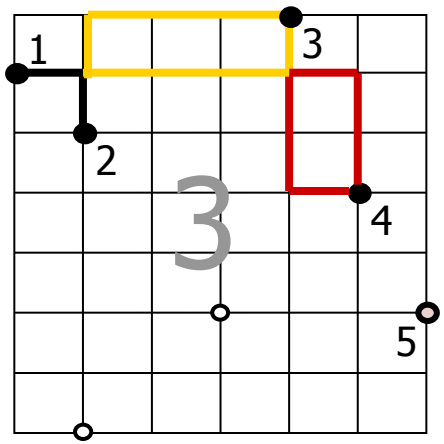


p_{MBB}

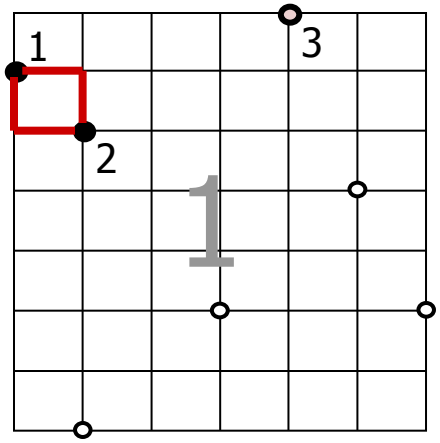
Пример. Шаг 3.

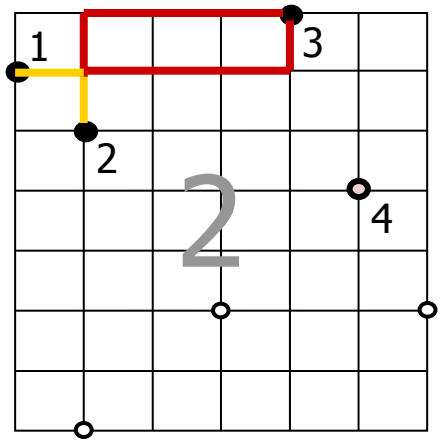


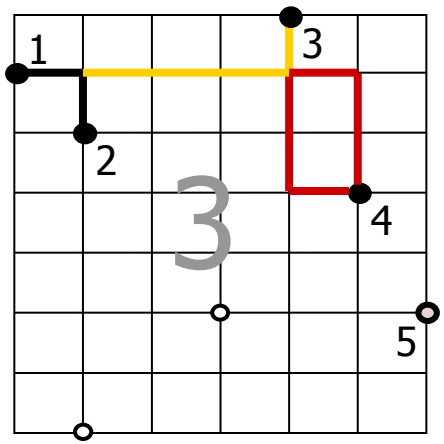


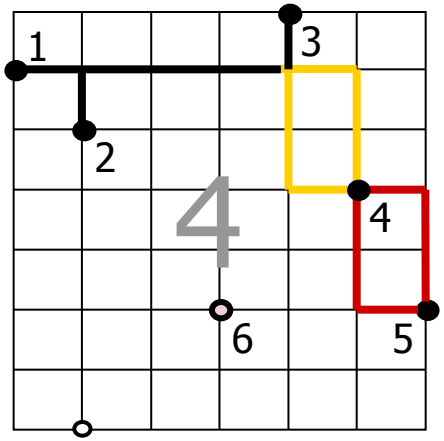


Пример. Шаг 4.

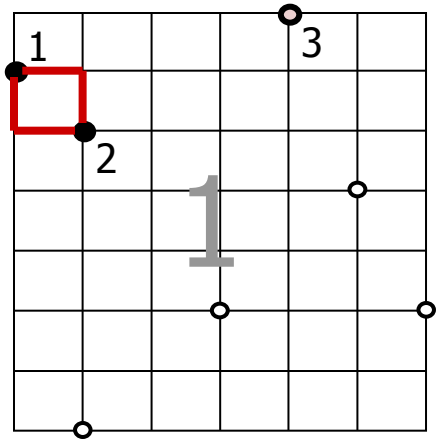


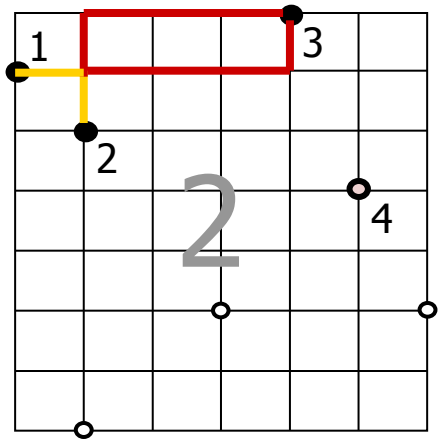


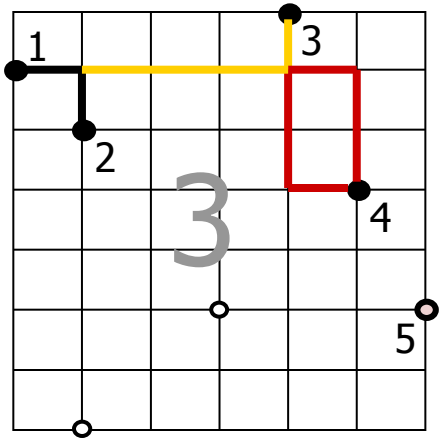


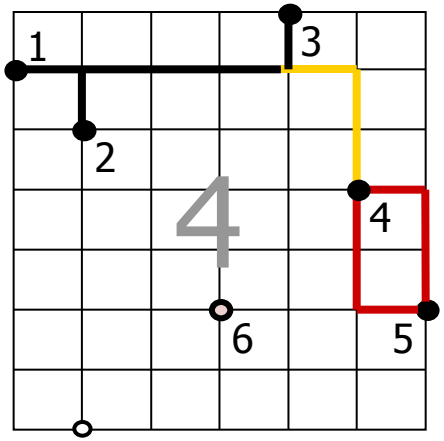


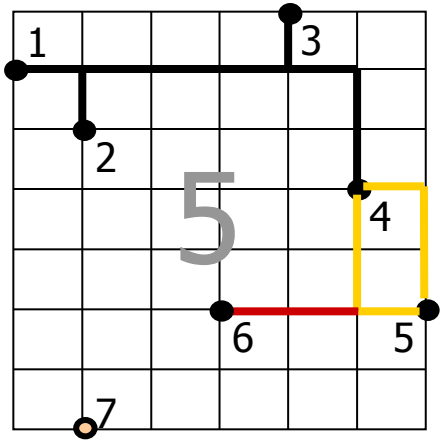
Пример. Шаг 5.



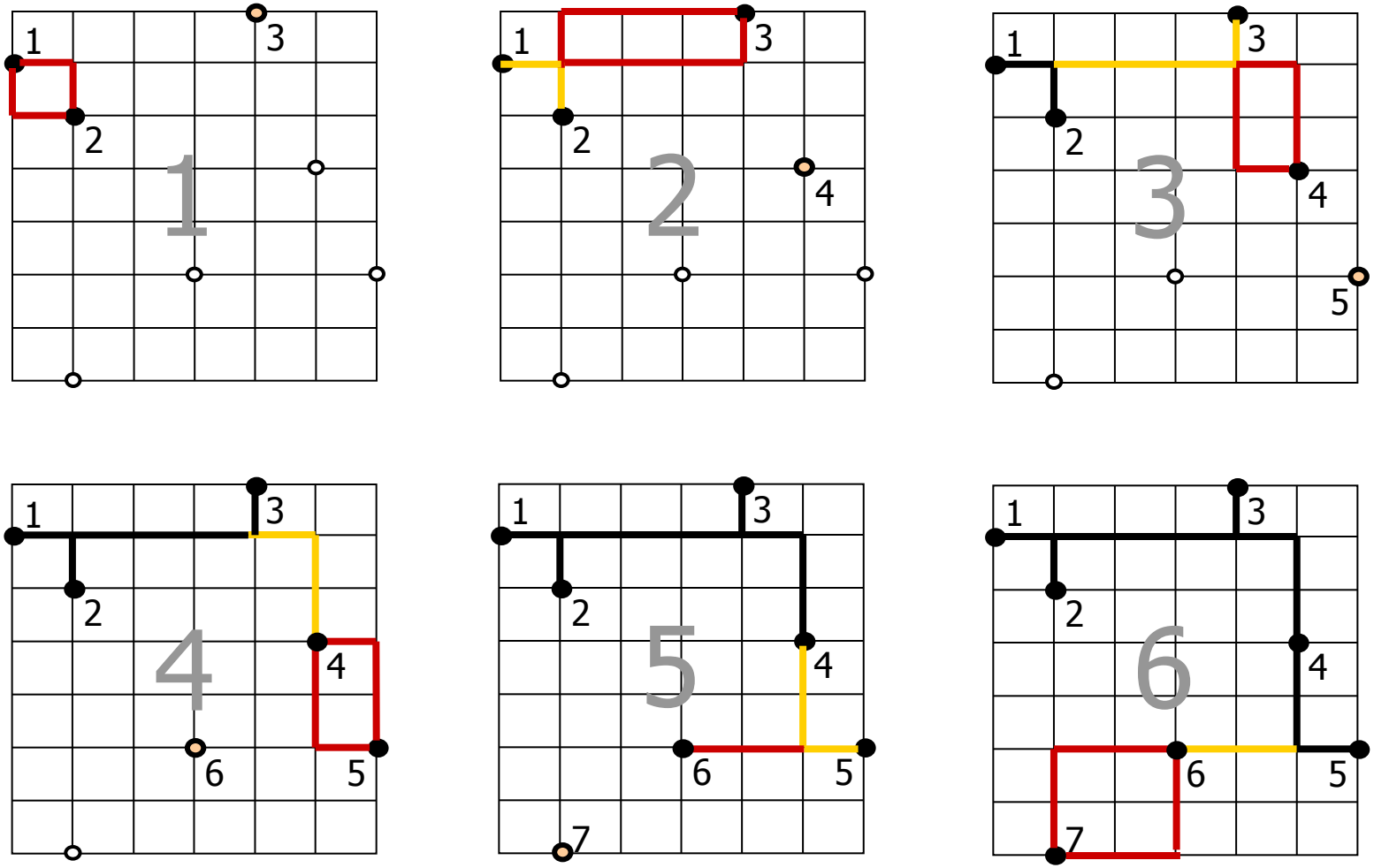




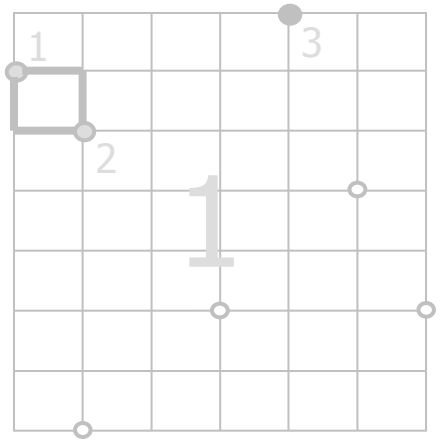


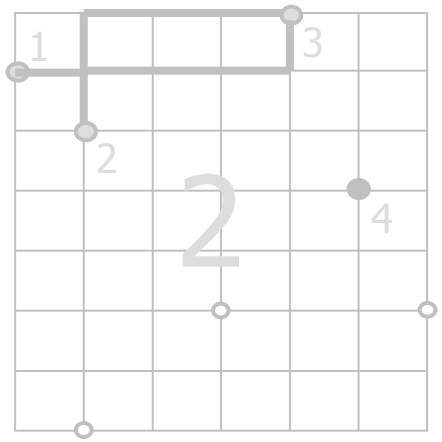


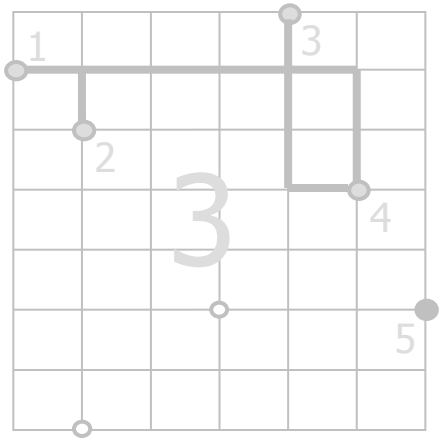
Пример. Шаг 6.

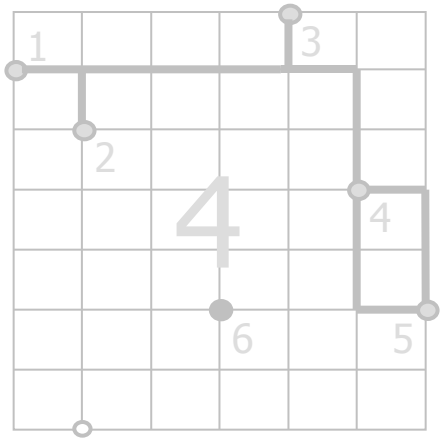


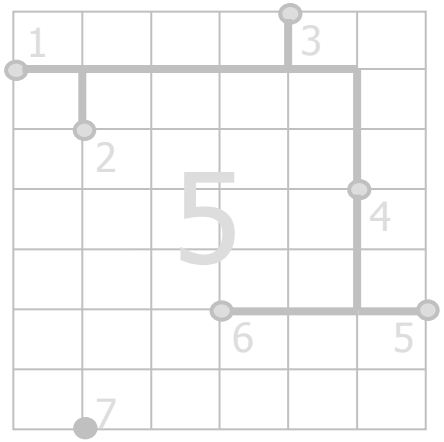
Пример. Решение.

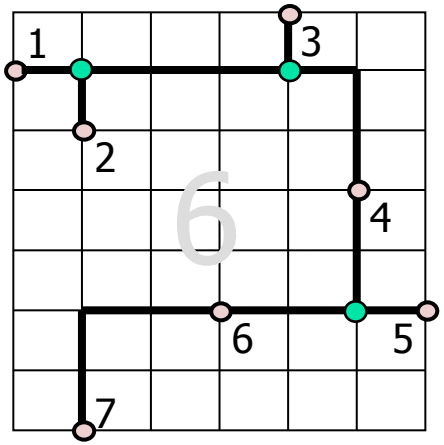




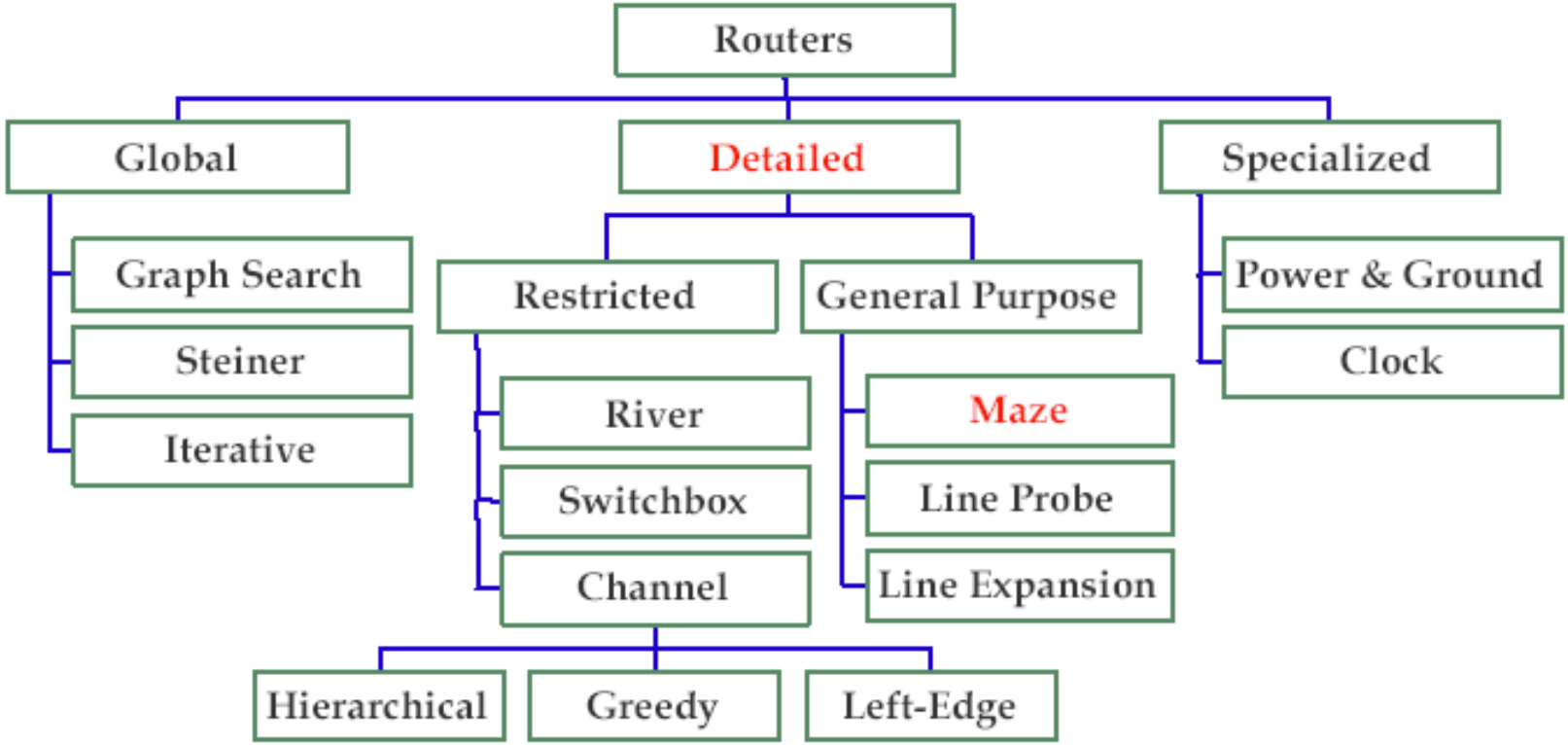








Классификация алгоритмов трассировки



Волновая трассировка (Maze)

Основная идея метода – волновой алгоритм (Lee, 1961)

- **Используется поиск в ширину (BFS)**
- **Путь определяется обратным проходом с минимизацией по дополнительному критерию (число поворотов)**
- **Всегда находится кратчайший путь, если он существует**

4	3	2	3	4	5	6	7	8	9	10	11
3	2	1	2	3	4	5	6	7	8	9	10
2	1	A	1		5	6	7	8			
3	2	1	2		6	7	8	9	10	11	12
4	3	2	3							12	13
5	4	3	4		14				B	13	14
6	5			13	14					14	
7	6	7		11	12	13	14				
8	7	8	9	10	11	12	13	14			
9	8	9	10	11	12	13	14				

Проблемы волновой трассировки

- **Большой расход памяти**
- **Выч. Сложность $O(n^2)$**
- **Последовательный характер трассировки**