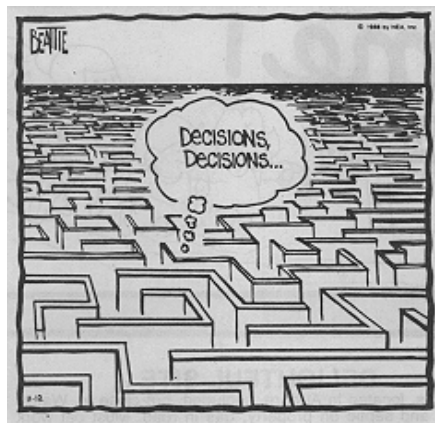


Chapter 9 Routing ctd

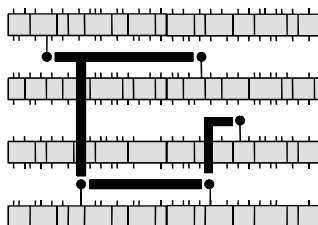
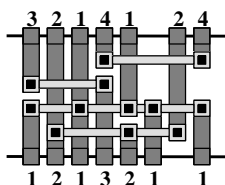
Global Routing



Local Routing

Local routing is the process of determining the exact patterns that interconnect sets of terminals in a given **routing area**.

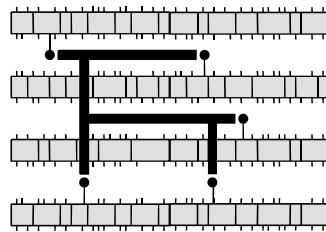
Local routing is opposed to **global routing**, the process of determining through which routing areas a connection will run without fixing the wiring patterns within the routing areas.



Introduction to Global Routing (1)

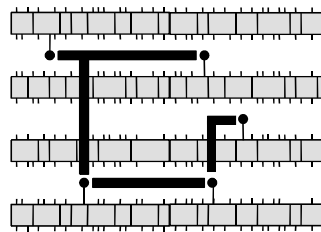
This is the process of roughly fixing the shapes of the connections for each net by distributing the wiring segments among the available channels. Each shape is a **rectilinear Steiner tree**.

Possible shape



(a)

Alternative shape



(b)

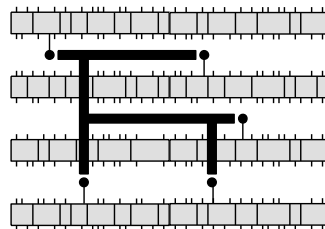
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

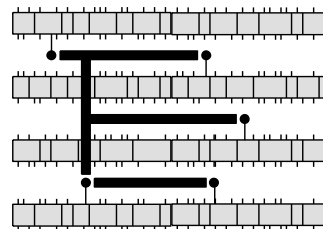
3

Introduction to Global Routing (2)

- The **channel density** is used to estimate the global routing quality.
- **Feedthroughs** (dedicated feedthrough cells, build-in feedthrough wires, unused metal layer) are used in standard-cell layout to cross cell rows; they may be scarce.



(a)



(b)

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

4

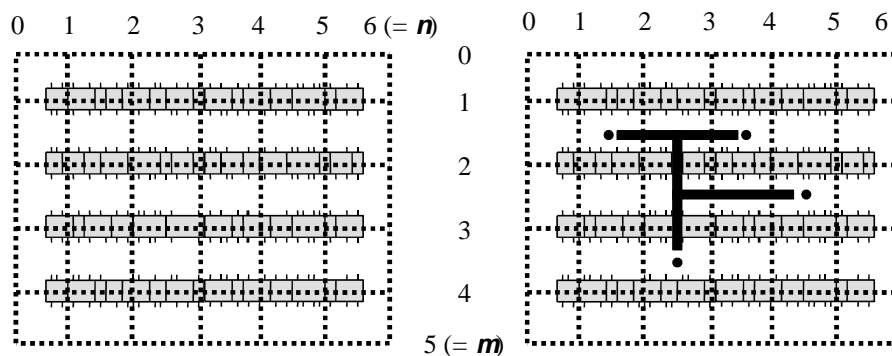
Timing-Driven Routing

With the decrease of feature sizes in CMOS, timing is becoming increasingly important in VLSI design. Timing models for global routing are:

- the **lumped** model: the entire wiring shape is considered a single resistor and capacitor; timing optimization amounts to reducing the total length of the Steiner tree.
- the **transmission-line** model: the distinct wiring shapes are separately modeled, making the signal arrive at different times at the terminals; timing optimization amounts to minimizing the wire length from the source terminal to the **critical sink**.

Problem Definition: Global Routing for Standard-Cell Layout

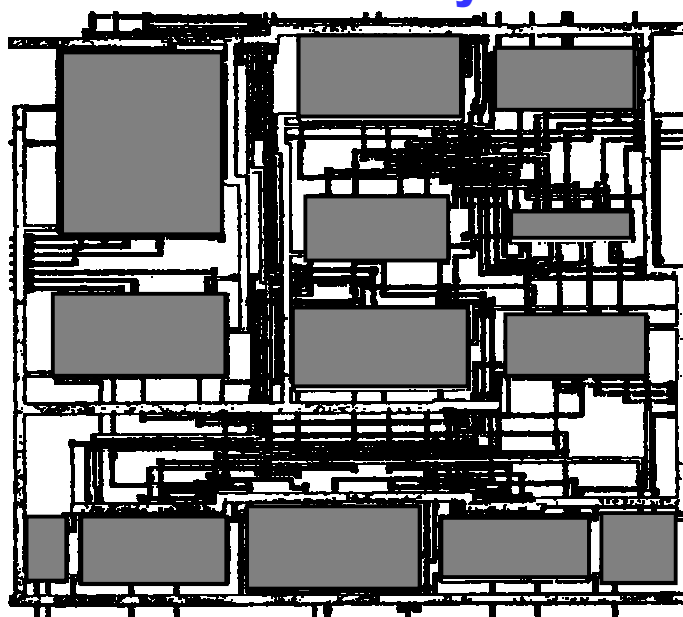
- Embed layout in a grid;
- Use grid centers to compute routing patterns (Steiner trees).

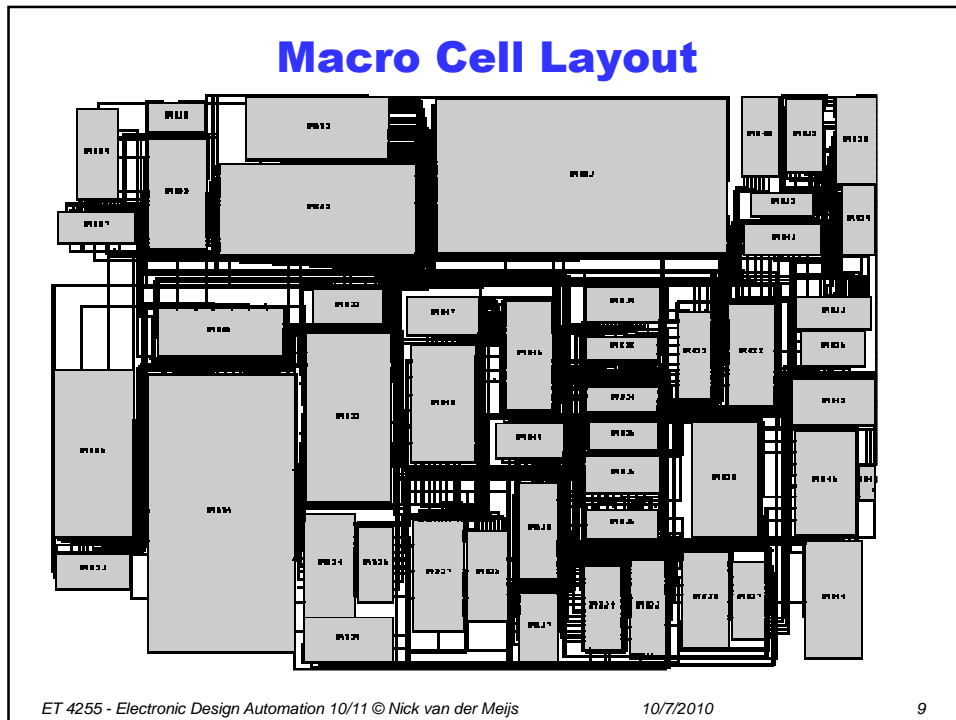


Problem Definition (Continued)

- The **local vertical density** $d_v(i,j)$ gives the number of wire segments crossing a vertical grid segment.
- The **local horizontal density** $d_h(i,j)$ gives the number of wire segments crossing a horizontal grid segment.
- The **channel density**: $D_v(i) = \max_{j=1}^n d_v(i, j)$
- Goal is to **minimize** the **total channel density**: $\sum_{i=1}^m D_v(i)$
 subject to $d_h(i,j) \leq M_{ij}$
 (nowhere exceed the feedthrough capacity).

Macro Cell Layout



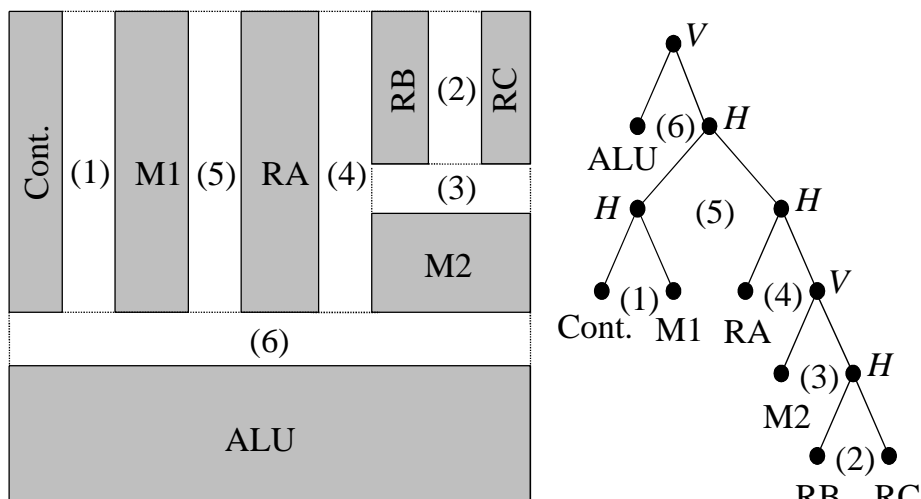


Global Routing for Building-Block Layout

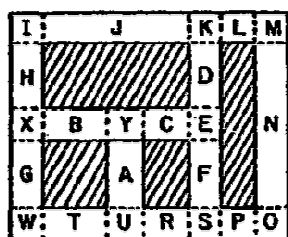
- Problem is more complex than in standard-cell layout.
- Identification of channels not always obvious: the **channel definition** problem.
- **Channel ordering** is important to ensure routability.
- Channel definition and ordering are simple if the placement has a **slicing floorplan**.
- Otherwise, **three-sided channels** and **switchboxes** may be required to finish local routing.

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 10

Channel Definition and Ordering

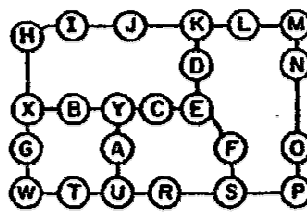


Region Adjacency Graphs

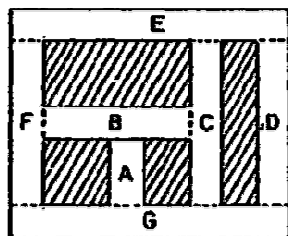


(a)

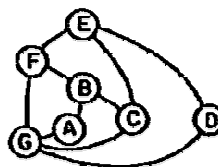
Macrocell layout



(b)



(c)



(d)

Bottleneck Graphs

Active edges

Active edges

no bottleneck

Active Region of bottleneck

Q: Why oh why
... were bottleneck graphs invented ?

A: To reduce the search space while sacrificing quality as little as possible!

(a)

(b)

(c)

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs
10/7/2010
13

Possible Approaches

Sequential routing:

- Compute minimum-weight Steiner trees, where the weight for crossing a grid segment is derived from wires already crossing the segment.
- Method suffers of same disadvantages as Lee routing (dependence on net ordering).

Simultaneous routing:

- Compute (nearly) optimal Steiner trees disregarding the net interaction.
- Resolve over-congestion by **reshaping** nets later on.

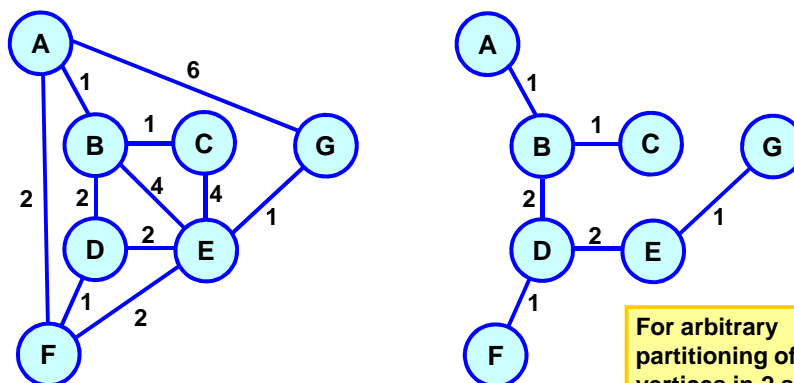
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs
10/7/2010
14

Rectilinear Steiner Tree Problem

- Given: a set of points $P = \{ p_1, p_2, \dots, p_n \}$ in the two-dimensional plane.
- The distance between two points $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$ is computed by $|x_i - x_j| + |y_i - y_j|$. (Manhattan distance)
- Problem: find the minimum-length tree interconnecting all points in P and additional points in the plane, if they contribute to a shorter tree length. These additional points, the **Steiner points**, are in the set S .

Property: a spanning tree has cost of at most $3/2$ that of the optimal Steiner tree
 → many heuristics start with spanning trees

Minimum Spanning Tree



- Book presents Prim's algorithm for MST
- Alternative: **Kruskal**
 - Sort edges in order of increasing weight
 - Add edges that don't form a loop
- Q: How to *efficiently* find if edges form a loop

For arbitrary partitioning of vertices in 2 sets, MST contains shortest edge between both sets

Steiner Tree Example

$G = \{a, b, c, d, e\}$
 $R = \{a, b, d\}$

Steiner tree in graph

Solution: cost = 9

Determine a Minimum Steiner Tree on R in G

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 17

Steiner Tree Heuristic (simple)

Input: graph $G = (V, E)$
 net $R = \subset V$

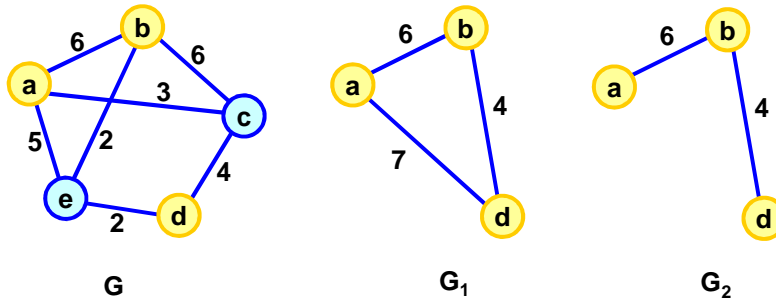
Output: approximation of MStT on R, in G

Algorithm:

1. Determine complete distance graph G_1 among vertices of R. Weight of edge (v, w) is shortest path between v and w in G.
2. Determine MST G_2 of G_1
3. Replace each edge in G_2 by corresponding path in G. This is G_3
4. Determine MST G_4 of G_3
5. Remove all leaves of G_4 which are not in R. This is the approximation of MStT.

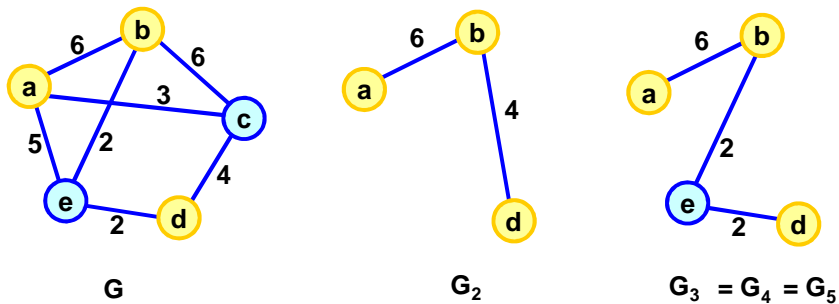
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 18

Steiner Tree Heuristic (simple)



1. Determine complete distance graph G_1 among vertices of R . Weight of edge (v, w) is shortest path between v and w in G .
2. Determine MST G_2 of G_1

Steiner Tree Heuristic (simple)



3. Replace each edge in G_2 by corresponding path in G . This is G_3
4. Determine MST G_4 of G_3
5. Remove all leaves of G_4 which are not in R . This is the approximation of MST, G_5 .

Sub-optimal!

Steiner Tree Heuristic (simple)

G

G₃: cost = 10

Optimal: cost = 9

Sub-optimal!

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 21

Steiner Tree: Flipping Heuristic

Steiner tree in plane

- Compute the spanning tree.
- Optimize the tree length by flipping L-shaped connections.
- Steiner points always have at least degree 3.

(a)

(b)

(c)

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 22

Iterated 1-Steiner Tree Heuristic

- **1-Steiner tree problem:** the minimal Steiner tree problem with the restriction that the tree contains only a single Steiner point.
- The optimal solution of the 1-Steiner tree problem can be found efficiently.
- The “repeated 1-Steiner tree heuristic” gives provably good results (but no optimal solution is guaranteed).
- **MRST problem:** given a set of points P in the plane, find the **Minimum Rectilinear Steiner Tree**

Iterated 1-Steiner Tree Heuristic

Input: P = set of points in the plane

Output: S = set of Steiner points

Algorithm:

$S = \emptyset$

cost = $c(\text{MST}(P))$

MST = minimum spanning tree

do

 old $S = S$; old cost = cost

$x = 1\text{-steiner point on } P \cup S$

$S = S \cup \{x\}$; cost = $c(\text{MST}(P \cup S))$

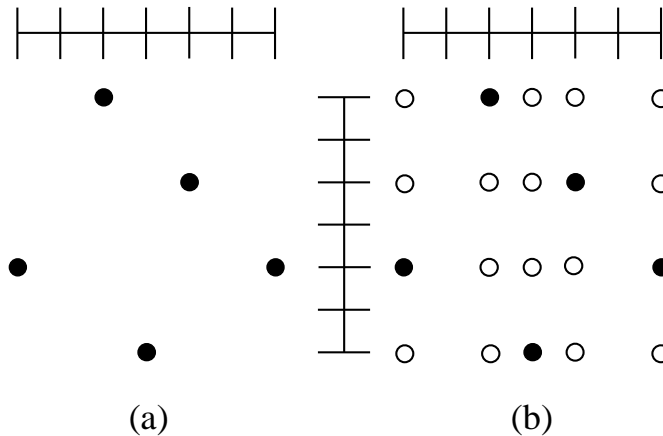
while (cost < old cost)

return S

Hanan Points

Hanan has proved that restricting all Steiner points to be on the grid lines already occupying the original points, does not prevent finding the optimal solution.

Used by 1-steiner



ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

25

1-Steiner: Try All Hanan Points

Input: P = set of points in the plane

Output: x = best Steiner point

Algorithm:

H = compute Hanan Points

best cost = ∞

foreach $h \in H$

 cost = $c(\text{MST}(P \cup \{h\}))$

if cost < best cost

 best cost = cost

$s = h$

end foreach

return s

■ Any comments?

■ Calling MST for each Hanan point is EXPENSIVE

■ It turns out that you can modify spanning tree for $P \cup \{h\}$ when you have the spanning tree for P

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

26

Get Spanning Tree Incrementally

```

(set of struct vertex, set of struct edge, int)
spanning_update (set of struct vertex V, set of struct edge E,
                 struct vertex s) {
    delta ← 0;
    V ← V ∪ {s};
    for each d ∈ {north, east, south, west} {
        u ← closest_point (V, s, d);
        delta ← delta - distance (s, u);
        E ← E ∪ {(s, u)};
        if ( cycle (V, E) ) {
            (v, w) ← largest_cycle_segment (V, E);
            E ← E \ {(v, w)};
            delta ← delta + distance (v, w);
        }
    }
    return (V, E, delta);
}

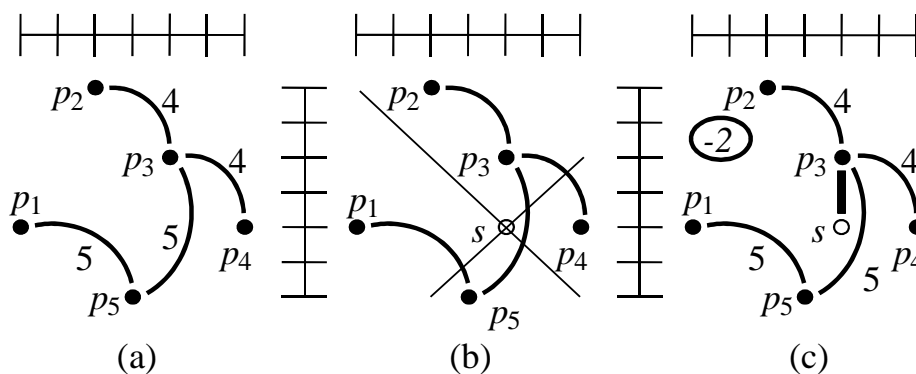
```

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

27

Example (Part 1)

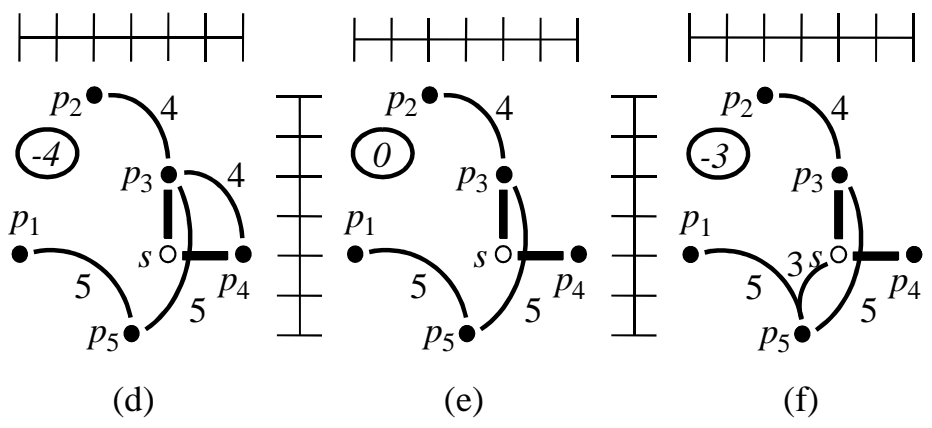


ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

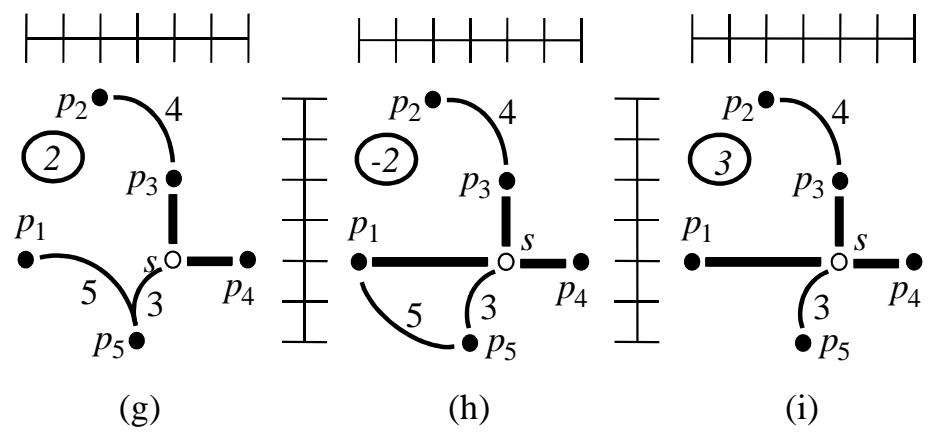
10/7/2010

28

Example (Part 2)



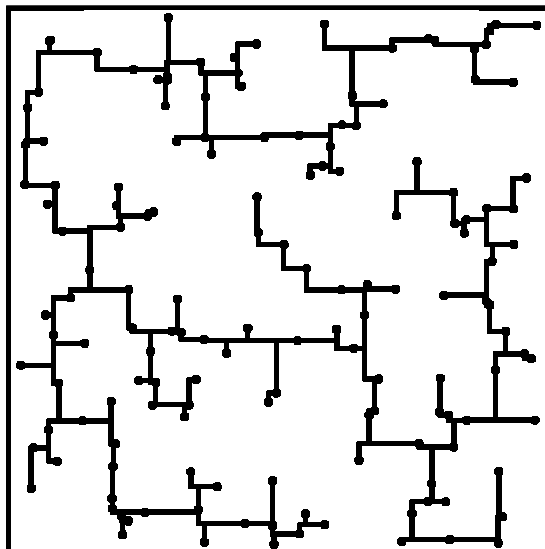
Example (Part 3)



Complexity

- n is number of points (vertices)
- UpdateSpanningTree is $O(n)$
- There are $O(n^2)$ Hanan points
- → 1-Steiner is $O(n^3)$
- 1-Steiner will be called $O(n^2)$ times
- → MRST is $O(n^5)$
- Very pessimistic upper bound: there are at most $n-2$ Steiner points
- Some improvements lead to $O(n^3)$ overall average time complexity

GeoSteiner Result

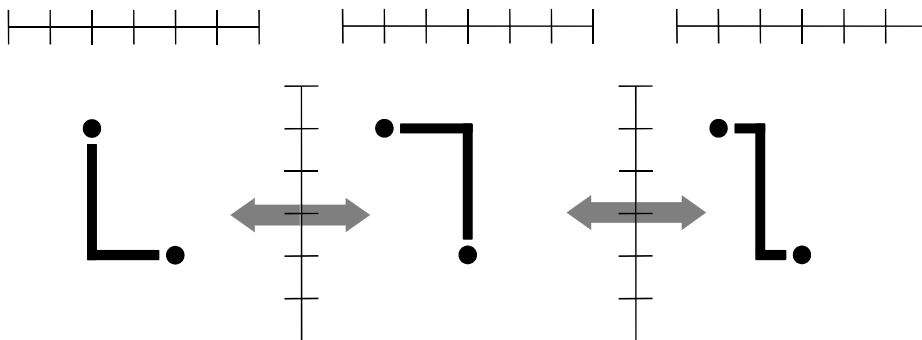


GeoSteiner package
Open source

Example:
150 points, 0.27 sec.

Local Transformations

Overcongestion after net-independent Steiner-tree construction can be eliminated by local transformations (e.g. guided by simulated annealing) or by ripping up a net and rerouting it with maze routing.

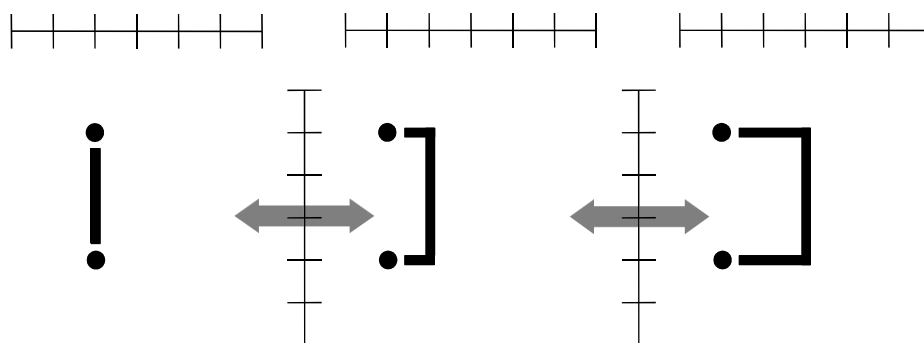


ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

33

Local Transformations (Ctd.)



ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs

10/7/2010

34