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.
CHARACTERIZATION

Local routing is characterized by a number of parameters (each parameter setting defines a distinct problem type):

* the number of wiring *layers*,

* the *orientation* of wire segments in a layer: horizontal, vertical, diagonal or some combination of these,

* *gridded* or *gridless* routing,

* presence or absence of *obstacles* in routing area,

* stretchable or fixed routing area,

* the constraints on the *positions* of the terminals: two parallel lines, along a rectangle, arbitrarily in an area, etc.

* terminals with a *fixed* or *floating* position.
THE LEE ALGORITHM FOR MAZE ROUTING

The *Lee algorithm* is a classical routing technique; it is the basis of many routing programs. The one-layer version is presented.

Main points:
* the routing area is a *grid of squares*.
* a square available for routing is *white*, one that is an obstacle is *black*.
* the goal is to connect all nets.
THE LEE ALGORITHM
(Continued)

* a two-point connection is realized by propagating a wave front from the source terminal outwards until the target terminal is reached.

* the shortest-connection is found by backtracking from the target to the source.

* in the case of multi-terminal nets: first two terminals are connected, this connection is the target for the wave propagation from the third terminal, etc.

* a routed net is an obstacle for the next nets.
THE LEE ALGORITHM
(Continued)

Evaluation:
* the algorithm always finds a connection if a connection exists.
* for two-terminal nets, this connection is the shortest possible; for multi-terminal nets the connection need not be the shortest possible (remember: the “minimal rectilinear Steiner tree” problem is NP-complete).
* it can be generalized for multiple layers: wave front expansion in three dimensions.
* its time complexity and space complexity is $O(n^2)$.
* the quality of its result strongly depends on the ordering of the nets.
Channel routing is characterized by:

* a rectangular routing area;
* the top and bottom rectangle boundaries contain terminals with fixed positions;
* the left and right boundaries of the channel have terminals with floating positions;
* the goal is to minimize the height of the routing area.
EXAMPLE

Solution in a restricted-layer model

Solution in a non-restricted-layer model
VERTICAL CONSTRAINTS

Two terminals located above each other give rise to a *vertical constraint*: the vertical segment connected to the top terminal cannot overlap with the vertical segment connected to the bottom terminal.
VERTICAL CONSTRAINTS (Continued)

* Vertical constraints can be combined into a *vertical constraint graph* under the assumption that each net will use one horizontal segment.

```
  a  b  c  d  e  f  g  h
  3  2  1  4  1  2  4
```

```
    2b  1c  1f  2g
    3a  4d  1e  4h
  1a  3d  2e  1h
  1  2
```

```
    3  2  1  4  1  2  4
    1  2  1  3  2  1  1
```
VERTICAL CONSTRAINTS (Continued)

* Cyclic constraints must be resolved by splitting horizontal segments.

* Doglegging can reduce the channel height.
HORIZONTAL CONSTRAINTS

The horizontal segments of different nets cannot be located on the same track. This is called a horizontal constraint.

The combination of horizontal and vertical constraints makes the channel routing problem difficult.
THE LEFT-EDGE ALGORITHM

Channel routing problems *without* vertical constraints can be solved efficiently by the *left-edge algorithm*.

**Problem:** given a set of segments (intervals) $[x_{i_{\text{min}}}, x_{i_{\text{max}}}]$, put non-overlapping segments on the same track such that the number of tracks is minimal.

THE LEFT-EDGE ALGORITHM

$l := $ “list of intervals sorted by first coordinate”; 
$t := 0$; 
**while** “$l$ is not empty” 
do $f := $ “first element in $l$”; 
$t := t + 1$
repeat “put $f$ on track $t$”; 
$f := $ “first element in $l$ non-overlapping with $f$” 
until “no such $f$ can be found”;

od;
THE LEFT-EDGE ALGORITHM (Ctd.)

An example problem:

```
1 4 5 3 5 7 2 3
```

```
6 4 1 6 4 7 7 2
```

```
0 5 10 15
```


Problem solution:
INTERVAL GRAPHS

A set of intervals defines an interval graph:
* there is a node for each interval,
* nodes corresponding to overlapping intervals are connected by an edge.

Example:

Intervals: \( i_1 = [1, 4], \quad i_2 = [12, 15], \)
\( i_3 = [7, 13], \quad i_4 = [3, 8], \quad i_5 = [5, 10], \)
\( i_6 = [2, 6], \quad i_7 = [9, 14]. \)

Solving the track assignment problem is equivalent to finding a minimal coloring of the graph.
**CHANNEL ROUTING ALGORITHMS**

* The channel routing problem is NP-hard.

* One heuristic is to use the left-edge algorithm while trying to satisfy the vertical constraints.

* The Lee algorithm will not perform very well if used directly (there is too much freedom initially; wrong decisions are taken, blocking future connections). However, it can be used within an iterative algorithm or as a postprocessing step for other algorithms.

* There are many algorithms based on other heuristics.