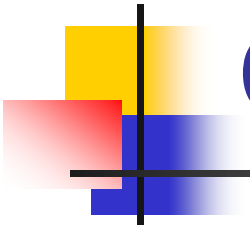
B-Escape: A Simultaneous Escape Routing Algorithm Based on Boundary Routing

Lijuan Luo[1], Tan Yan[1], Qiang Ma[1]
Martin Wong[1], Toshiyuki Shibuya[2]

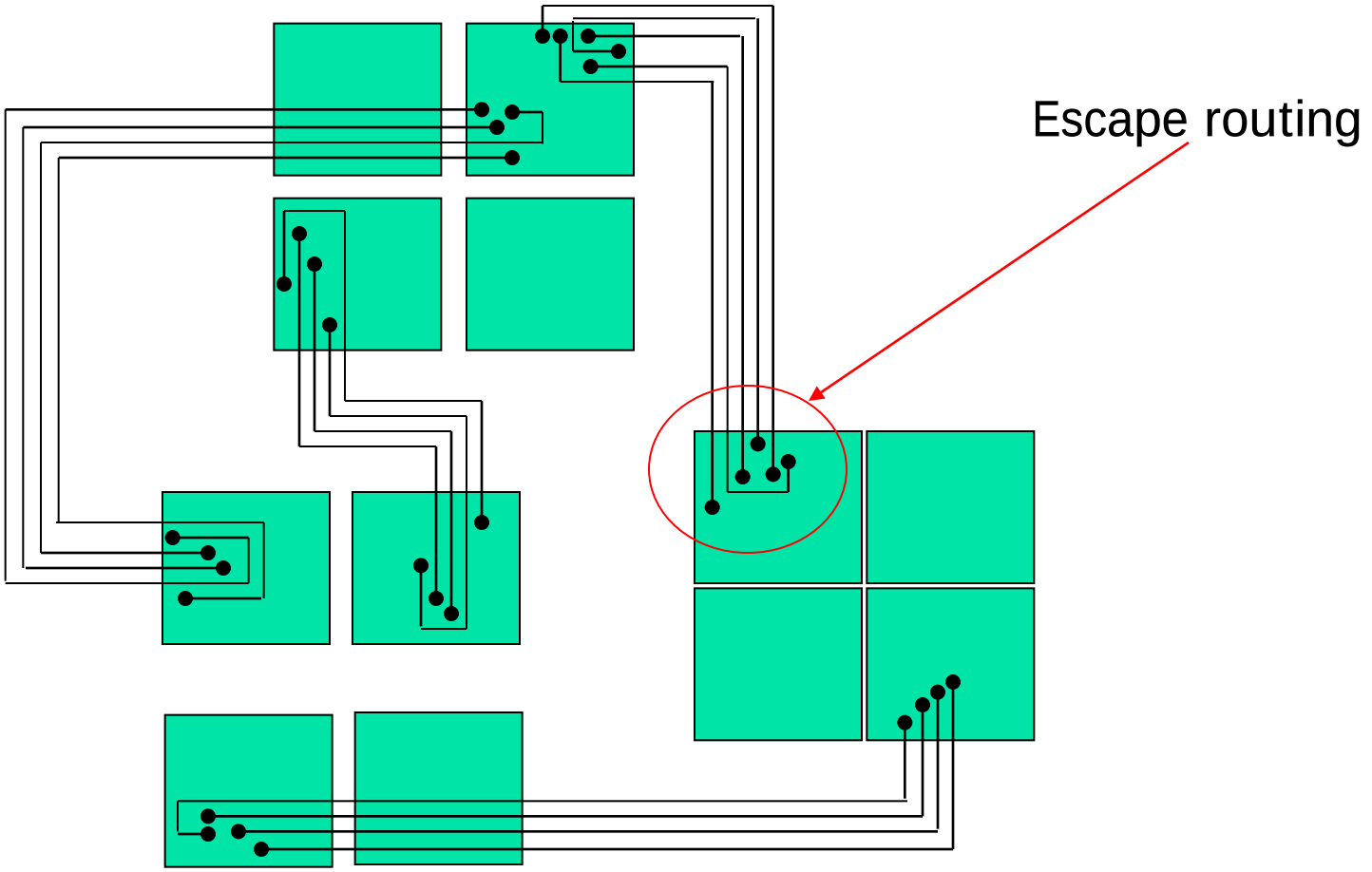
[1] ECE Department, University of Illinois, Urbana-Champaign
[2]Fujitsu Laboratories of America, Inc.



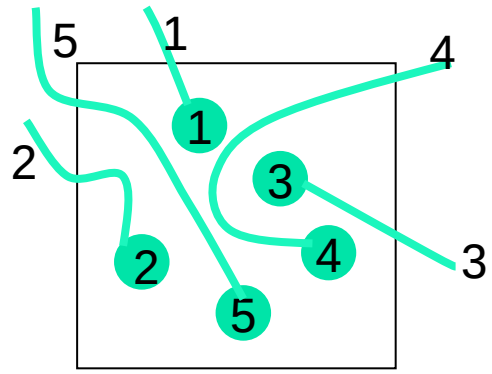
Outline

- Introduction
- Our Algorithm (B-Escape)
  - Boundary Routing
  - Dynamic Net Ordering
- Implementation Details
- Experimental Results
- Conclusion

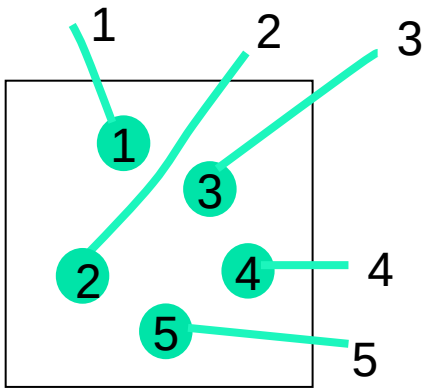
PCB Routing



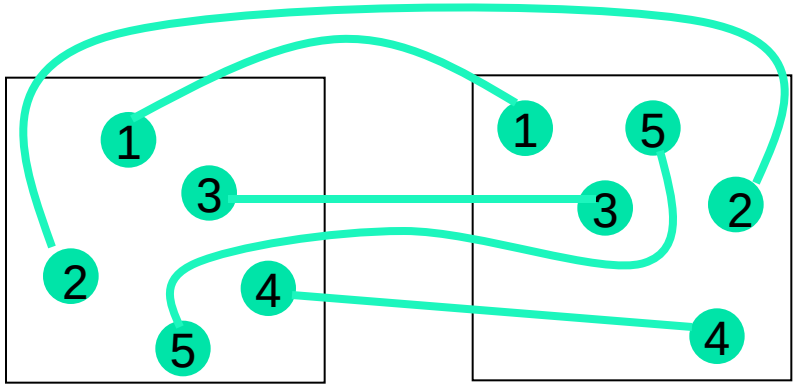
Escape Routing Problems



(a) Unordered Escape

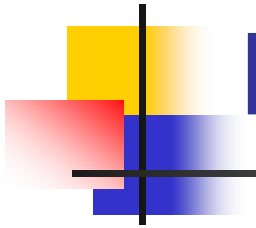


(b) Ordered Escape



(c) Simultaneous Escape

Previous Approaches

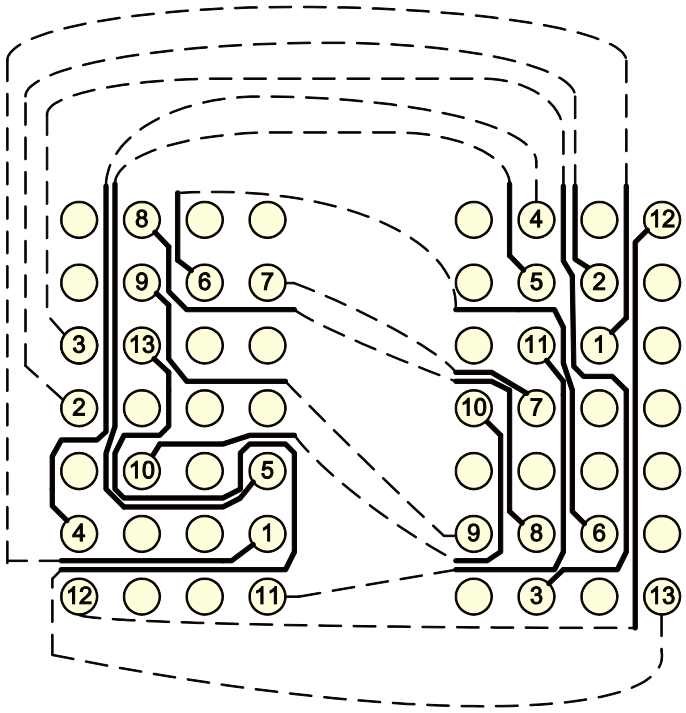


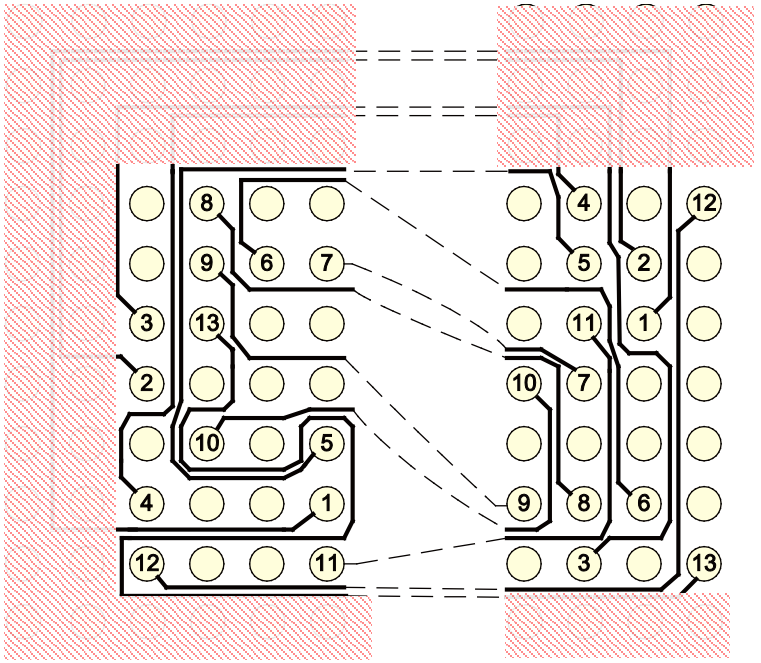
- Manual routing 
  - Time Consuming
- Pattern Routing
  - Limited ability to do complex escapes
- Negotiated Congestion Routing
  - Difficult to resolve crossings

Our Contributions

- Introduce a boundary routing approach
- Capable of handling complicated problems in very short time
- Solved 14 industrial benchmarks while Cadence Allegro only solved 7.

Focus on 1-side Escape

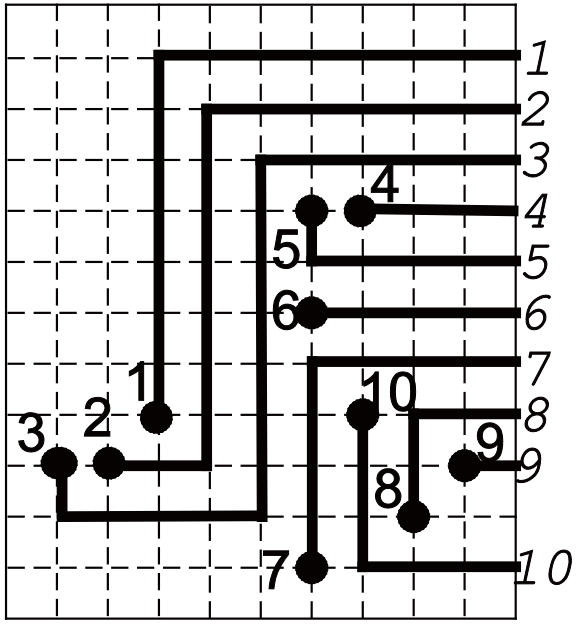




- Without loss of generality, we only consider 1-side Escape.

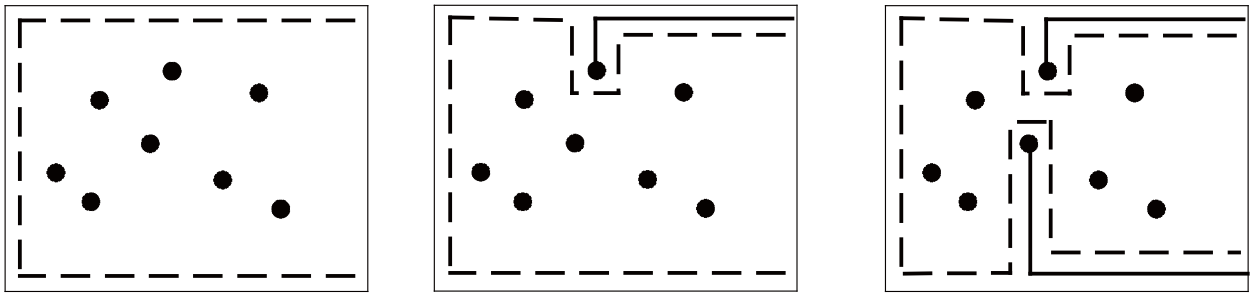
Ordered Escape

- Use ordered escape to present our boundary routing approach
- Foundation of our simultaneous escape

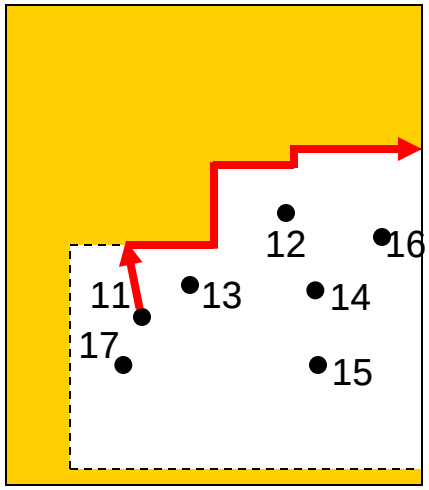


Routing Boundary

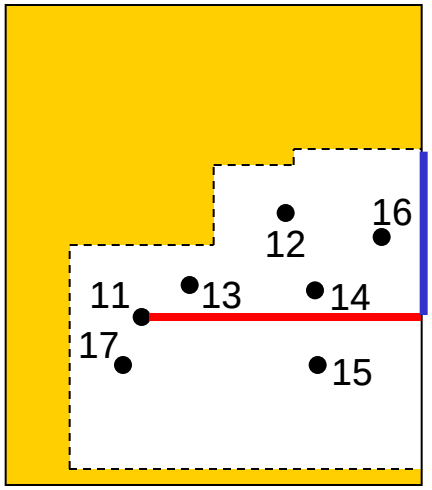
- The boundary of the maximum routable region for the unrouted pins
- Shrinks as we route more pins



Boundary Routing Methodology



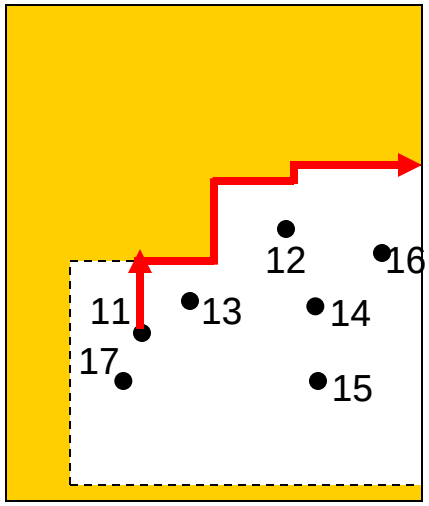
(a) Boundary routing



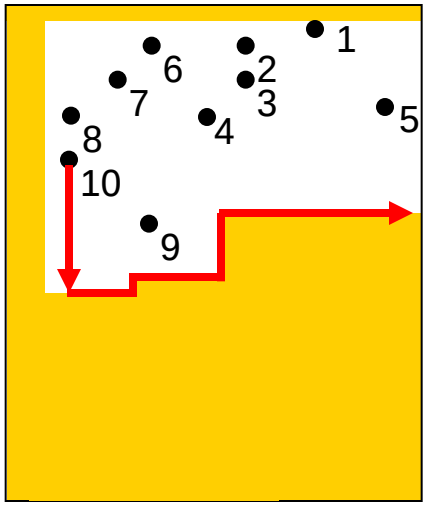
(b) Routing based on shortest path

- Tend to leave more space for the unrouted pins

Routing Modes



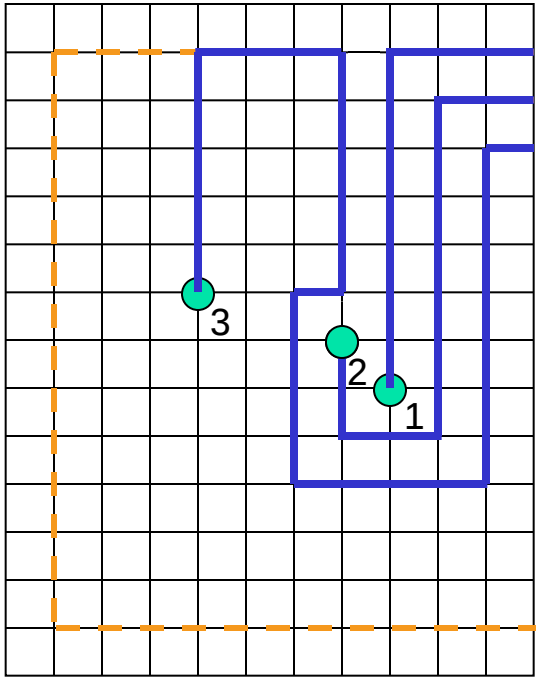
Upward Mode



Downward Mode

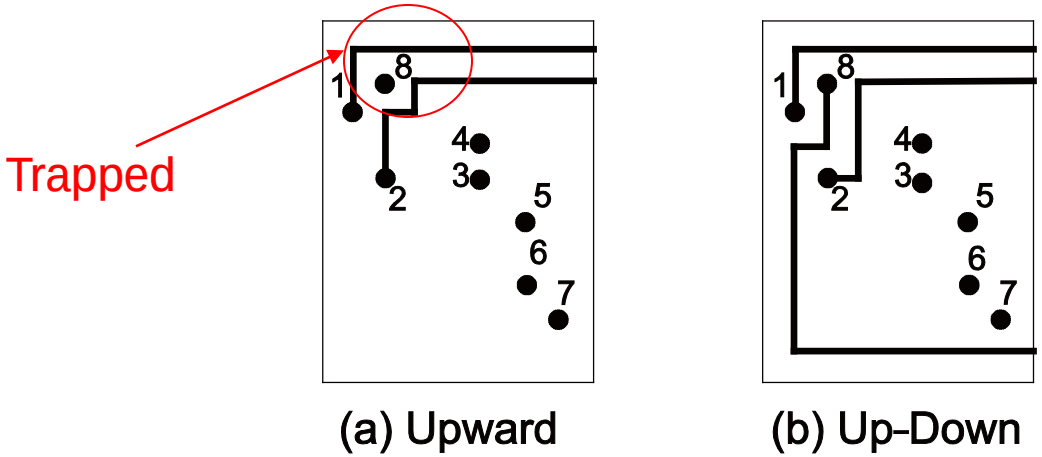
Example

- Upward Mode



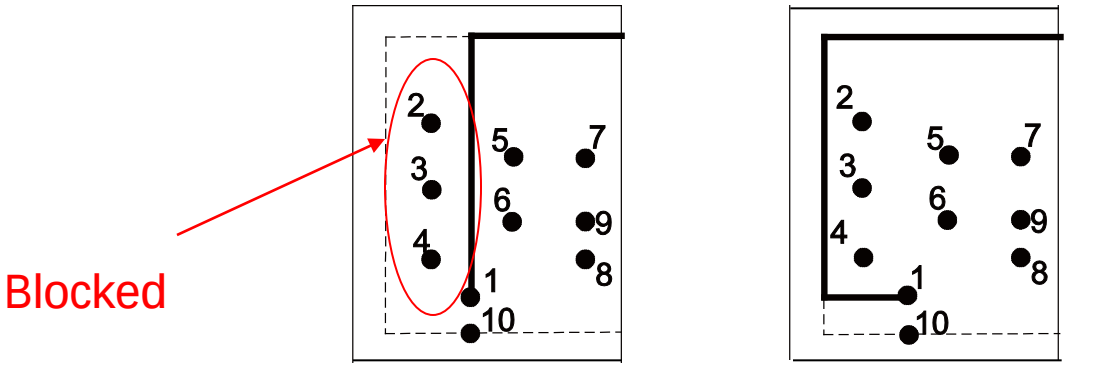
Routing Modes

- Up-down Mode
  - Alternate between the up and down modes



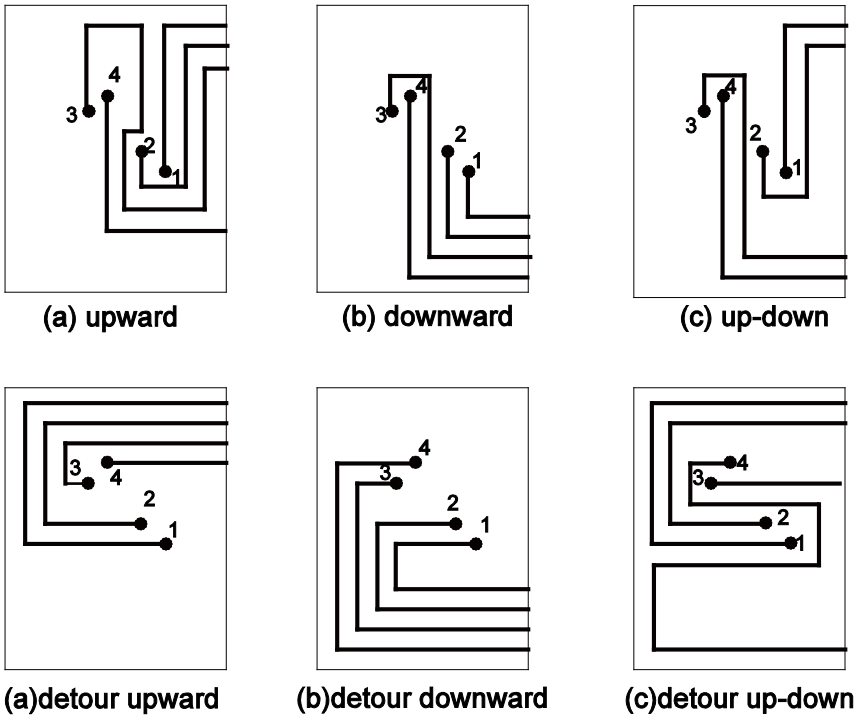
Routing Modes

- Detour Modes
  - Go leftward to reach the boundary
    - Detour upward
    - Detour downward
    - Detour up-down



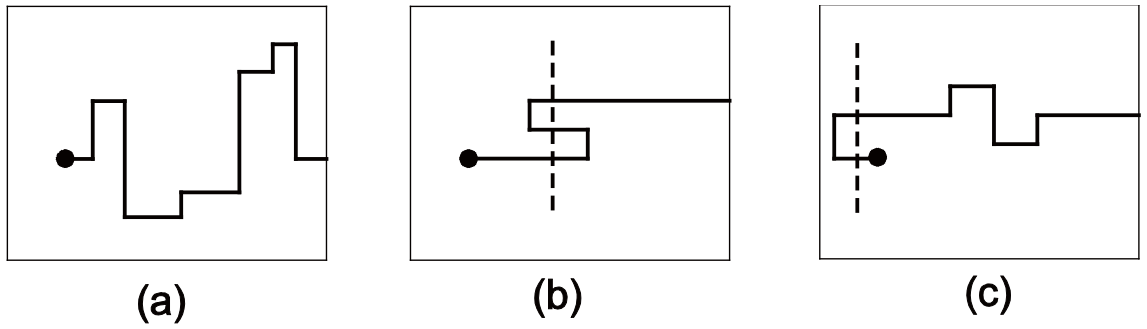
(a) Upward (b) Detour upward

Six Routing Modes



Optimality for Monotonic Routing

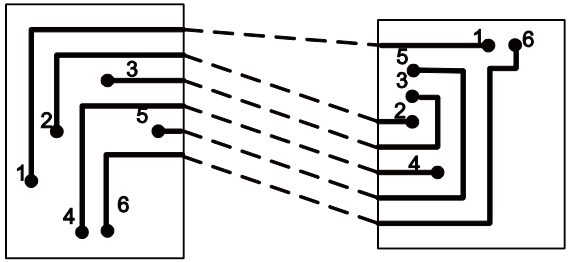
- Guarantee an escape solution for monotonically escapable problems



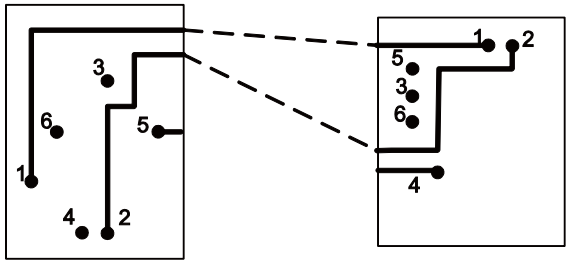
Monotonic routing (a); Non-monotonic routing (b) (c)

Dynamic Net Ordering

- Huge routability differences caused by slightly different net orderings
- Difficult to decide the correct ordering beforehand.



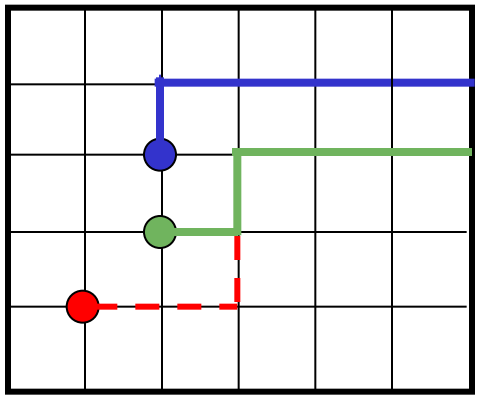
(a) Correct Ordering

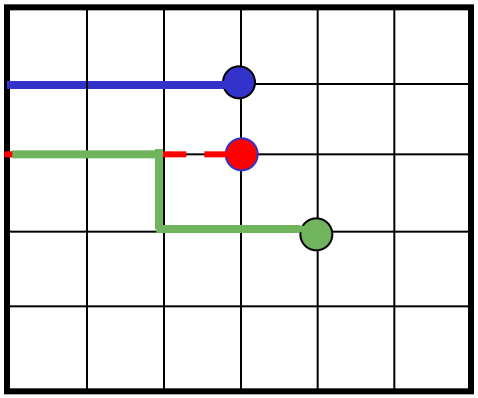


(b) Wrong Ordering

Dynamic Net Ordering

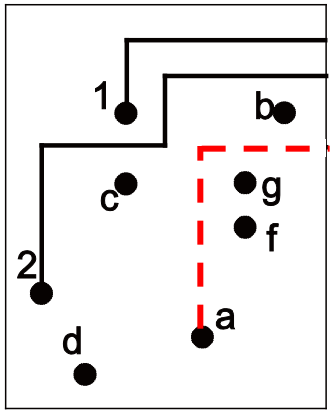
- Tentatively route each remaining net (using boundary routing)
- Choose the “best” one as the next net





Cost Function

- Trap cost: #Pin unroutable
- Block Cost: #Pin blocked (still routable)
- The overall cost takes both components into account.



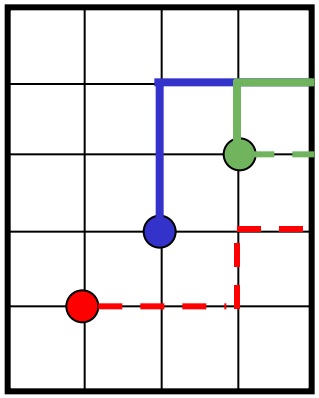
Cost of Net a: (α, β)

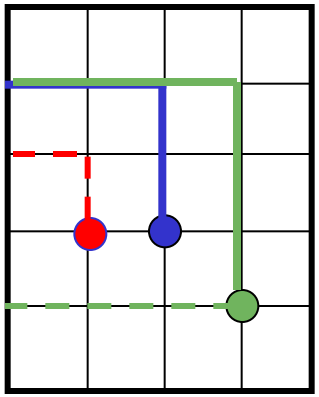
Trap b : $\alpha = 1$

Block c : $\beta = 1$

Reordering

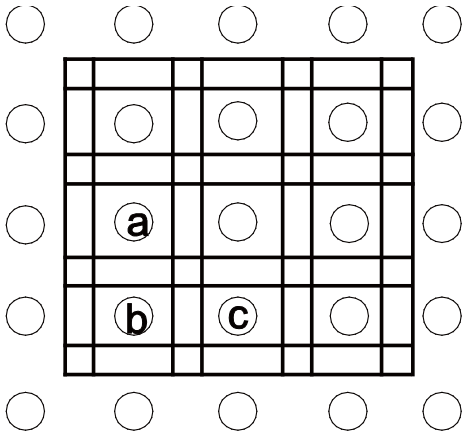
- Sometimes pins may get trapped
- Backtrack and get a different ordering



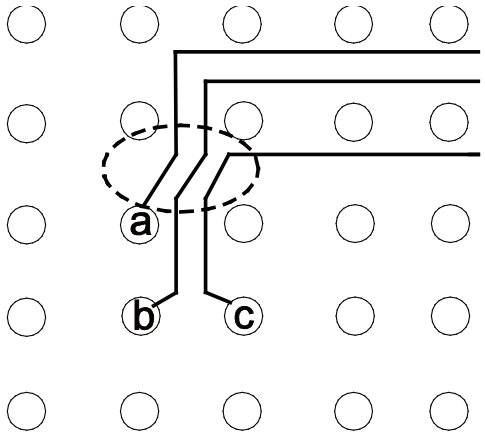


Detailed Modeling of Routing Grid

- Need to capture diagonal routing



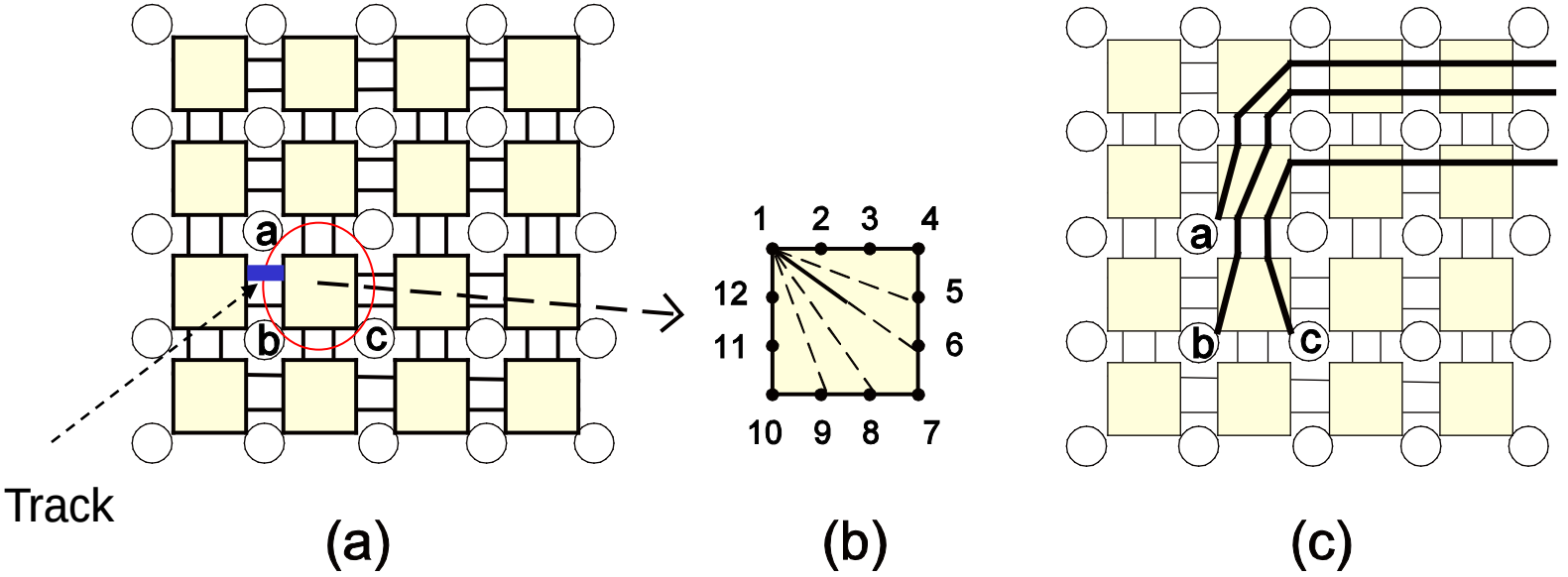
(a)



(b)

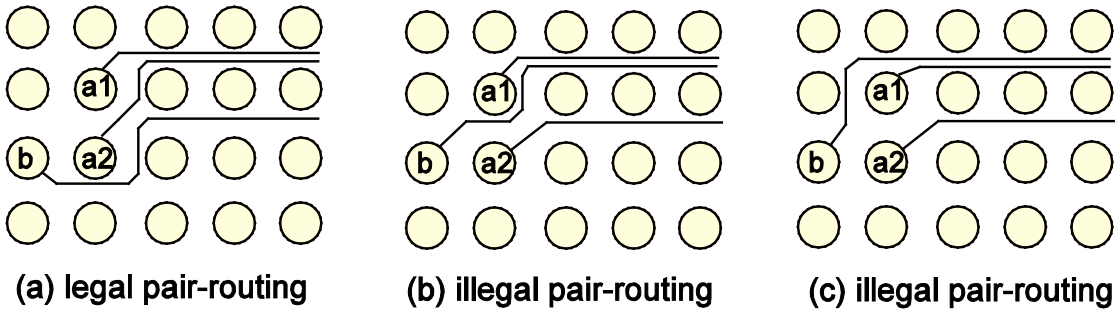
Detailed Modeling of Routing Grid

- Introduce “Switch-box” into the grid structure
- The switching condition:
  - No crossing
  - Satisfy capacity constraint



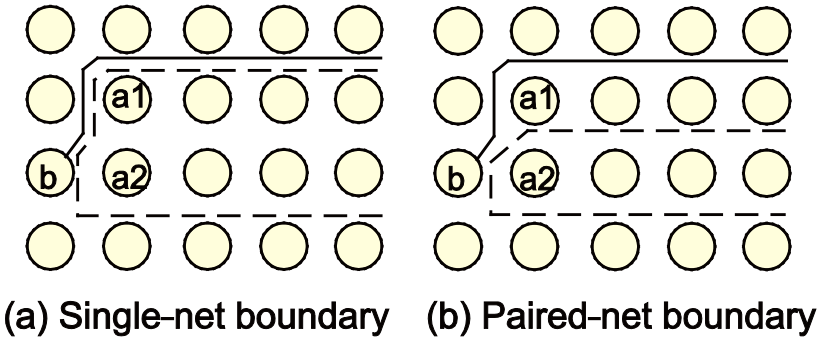
Routing Differential Pairs

- Pair constraint: If two nets belong to a differential pair, they are required to route together.
- Solution
  - Order the paired nets successively. (avoid (b))
  - Define a new boundary for paired nets.

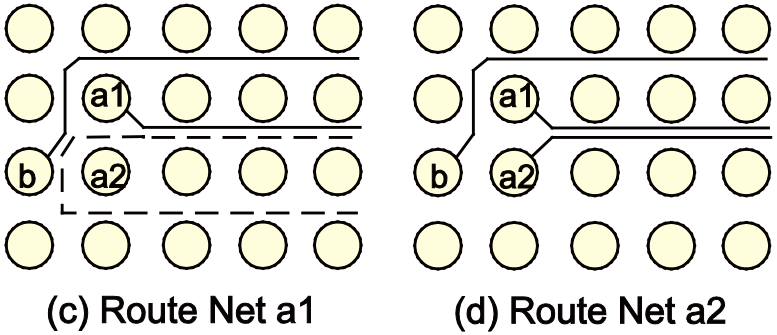


(a) legal pair-routing (b) illegal pair-routing (c) illegal pair-routing

Routing Differential Pairs



(a) Single-net boundary (b) Paired-net boundary



(c) Route Net a1 (d) Route Net a2

Experimental Setup

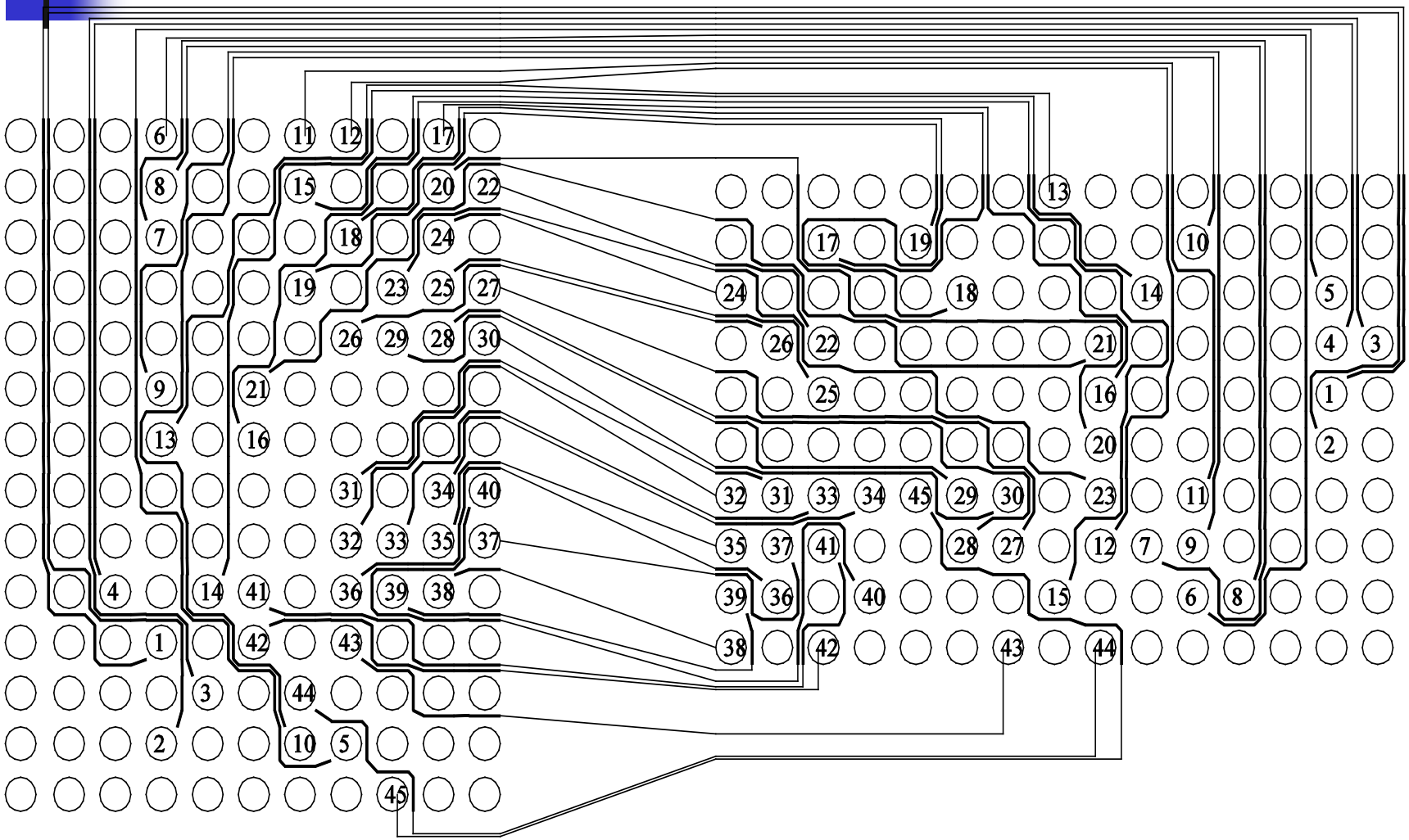
- Implemented in C++
- Pentium 4 2.8 GHz system with 4GB memory
- Benchmarks:
  - 14 Industrial benchmarks
  - Manually solved by designers for 8 hours/benchmark
  - #Net (18~64)
  - Grid size (16×14 ~ 38×36)

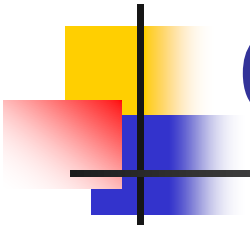
Experimental Results

Benchmark	Allegro	B-Escape
Ex1	100%	100%
Ex2	100%	100%
Ex3	100%	100%
Ex4	95%	100%
Ex5	80%	100%
Ex6	100%	100%
Ex7	90%	100%
Ex8	100%	100%
Ex9	100%	100%
Ex10	95%	100%
Ex11	96%	100%
Ex12	100%	100%
Ex13	70%	100%
Ex14	80%	100%
# routed problem	7/14	14/14

- Routability comparison with Cadence Allegro PCB Router:
  - 14/14 v.s. 7/14
- Run time:
  - B-Escape: 0.2s~691.3s
  - Negotiated Congestion Router: 55.8s~6189.0s

Experimental Results





Conclusion

- Currently escape problems are mostly solved manually
- We have presented a boundary routing method to solve complicated escape problems
- B-Escape outperforms the commercial PCB router.