Efficient Multilayer Routing Based on Obstacle-Avoiding Preferred Direction Steiner Tree

Ching-Hung Liu, Yao-Hsin Chou, Shin-Yi Yuan,

and Sy-Yen Kuo

National Taiwan University

Outline

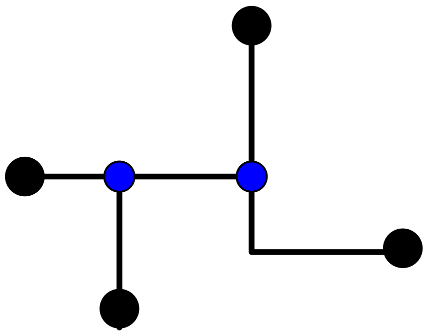
- Introduction
- Problem Formulation
- Routing Graph
- Approximation Algorithm
- Experimental Results
- Conclusion

Outline

- Introduction
- Problem Formulation
- Routing Graph
- Approximation Algorithm
- Experimental Results
- Conclusion

The RSMT Problem

- Given a set of pins, a rectilinear Steiner minimal tree (RSMT)
  - Connects all pins, possibly through some Steiner points.
  - vertical and horizontal edges.
  - minimum total wirelength
- In VLSI design, RSMTs are used to route signal nets.

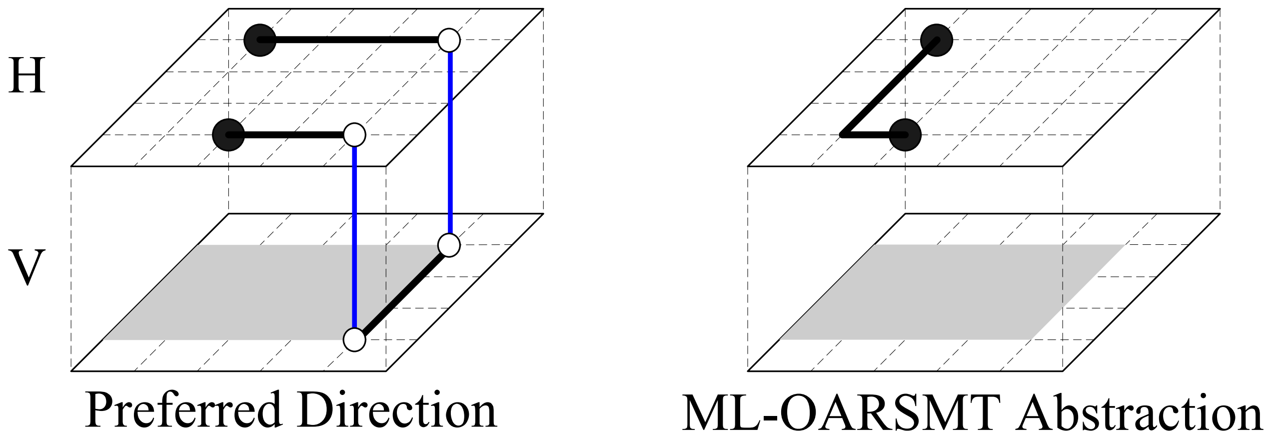


- Garey and Johnson, “The RSMT problem is NP-complete”
- Many approximation and heuristic methods.
- Two IC design constraints→ RSMT Abstraction impractical
  - Obstacles: IP blocks, Prerouted nets, Large-scale power networks.
  - Multiple Routing Layers: layered metal material

Preferred Directions

- Lin et al. (ICCAD 07)—the multi-layer obstacle-avoiding RSMT (**ML-OARSMT**) problem.
- The ML-OARSMT abstraction is still imperfect due to another practical condition.
- **Preferred directions:**

  Considering signal integrity and IC manufacturing, the orientation of routing in a single layer must be either vertical or horizontal.



Difference Routing Resources

- Routability is an important issue (ten thousand nets).
- Congestion should be minimized.
- Different routing resources could be a strategy for improving routability.
  - I.e., to weight the routing cost of a congested layer more higher such that the router would avoid routing nets through this congested layer.
- Yildiz and Madden (TCAD 21:11 2001) proposed preferred direction Steiner tree (**PDST**) model:
  - Different routing resources
  - Via cost
- But, the PDST model does not consider obstacles.

Obstacle-Avoiding Preferred Direction Steiner Tree

- To our best knowledge, none of existing works entirely catches all mentioned constraints at the same time
  - Processing conditions:
    obstacles, multiple layers and preferred directions.
  - Constraints for improving routability:
    via costs and different routing resources.
- Obstacle-Avoiding Preferred Direction Steiner Tree (OPDST) model
  - Obstacles
  - Multiple layers
  - Preferred directions
  - Different routing resources
  - Via costs

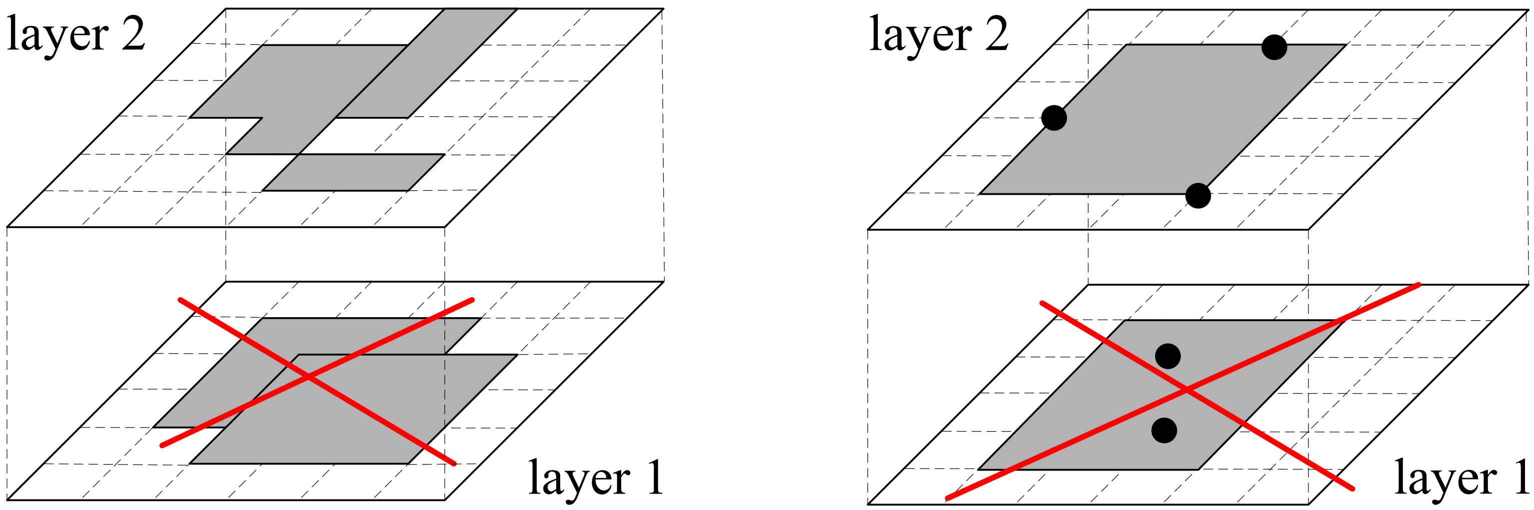
Summary of Our Contributions

- This is the first attempt to formulate the OAPDST problem which considers more practical conditions.
- We propose a routing graph, preferred direction evading graph (**PDEG**) and prove that at least one optimal solution exists in PDEG.
- We develop a factor 2 approximation algorithm to provide near-optimal solutions and discuss further contributions of the related proofs.
- In Brief, our work is very theoretical and consists of many theorems and proofs. We will detail the theoretical and practical benefits from those theorems.
- Besides, experimental results also show that our method outperforms an extension of traditional methods. (e.g, over 40 % total cost improvement and more stable running time)

Outline

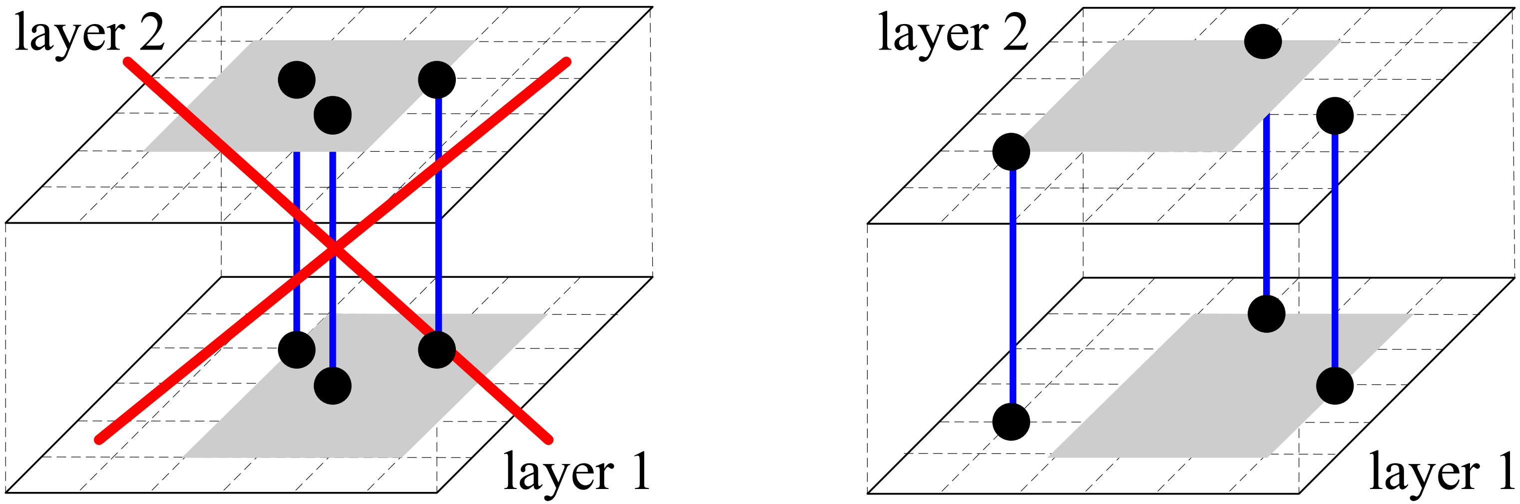
- Introduction
- **Problem Formulation**
- Routing Graph
- Approximation Algorithm
- Experimental Results
- Conclusion

Definitions for An Instance$_1$



- **Obstacles** are rectilinear polygons.
  - They cannot overlap with each other (see layer 1) but could be point-touched at the corners or line-touched at the boundaries (see layer 2).
- **Pin-vertices** cannot locate inside an obstacle but they could be at the corner or on the boundary of an obstacle

Definitions for An Instance$_2$



- A via is a rectilinear edge connecting two vertices in adjacent layers.
  - Neither of the two endpoints can locate inside an obstacle (see left), but they could be on the boundary or at the corner of an obstacle (see right).

Symbols and Assumptions

N_l	The **number** of routing layers
m	The **number** of pin-vertices
n	The **number** of pin-vertices and obstacle corners. (**Input Size**)
C_v	The **cost** of a via
UC_i	The **unit cost** of wires in **layer i**

- Without loss of generality, we assume the PD constraints as follows:
  - **Odd** layers only allow **vertical** edges
  - **Even** layers only allow **horizontal** edges

Problem Formulation

- The Obstacle-Avoiding Preferred Direction Steiner Tree (OAPDST) Problem:
  - Given a constant C_v, a set of pin-vertices, a set of obstacles, N_l routing layers with their specific unit costs of wires (UC_i, $1 \leqq i \leqq N_l$), and **PD** constraints.
  - Construct a tree connecting all pin-vertices possibly through some Steiner points
    - No tree edges intersect any obstacles
    - No tree edges violate the PD constraints
    - The total cost is minimized
  - An **OAPDST**: a solution; An **OAPDSMT**: an optimal solution

ML-OARSMT	obstacles, multiple layers
PDST	multiple layers, different routing resources
OAPDST	obstacles, multiple layers, different routing resources

Outline

- Introduction
- Problem Formulation
- **Routing Graph**
- Approximation Algorithm
- Experimental Results
- Conclusion

Preferred Direction Evading Graph (PDEG)

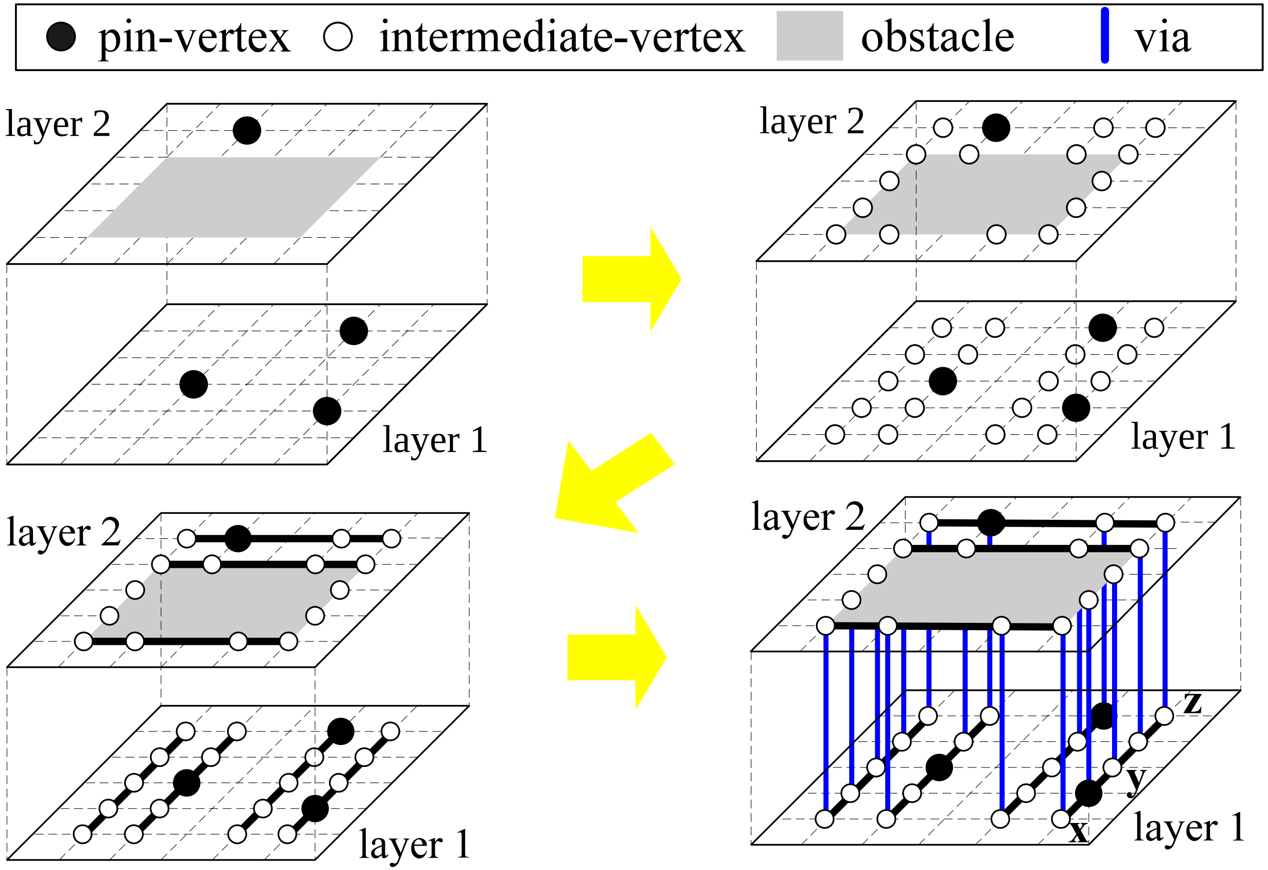
- PDEG is a routing graph constructed by our algorithm for the OAPDST problem, and PDEG guarantees to hold at least one optimal solution (we denote that as the optimality of PDEG or PDEG optimality).

- **Observation:**

  Vertices with the same coordinates as pin-vertices and obstacle corners are possible candidates of an optimal solution.

- For holding an optimal solution, PDEG should contain all possible candidates of an optimal solution.

PDEG Construction



- The time complexity of PDEG Construction is **O(n²)**
- The vertex size and edge size of PDEG are both **O(n²)**

Evading Line Segment and Via Connection

- A line segment is a rectilinear edge connecting two vertices in the same layer.
- A via connection is a rectilinear edge connecting two vertices in different layers.
- A line segment (via connection) can span more than one vertex.
- An evading line segment is a maximal line segment of PDEG.
- An evading via connection is a maximal via connection of PDEG.
- PDEG is represented as a union of evading line segments and evading via connections

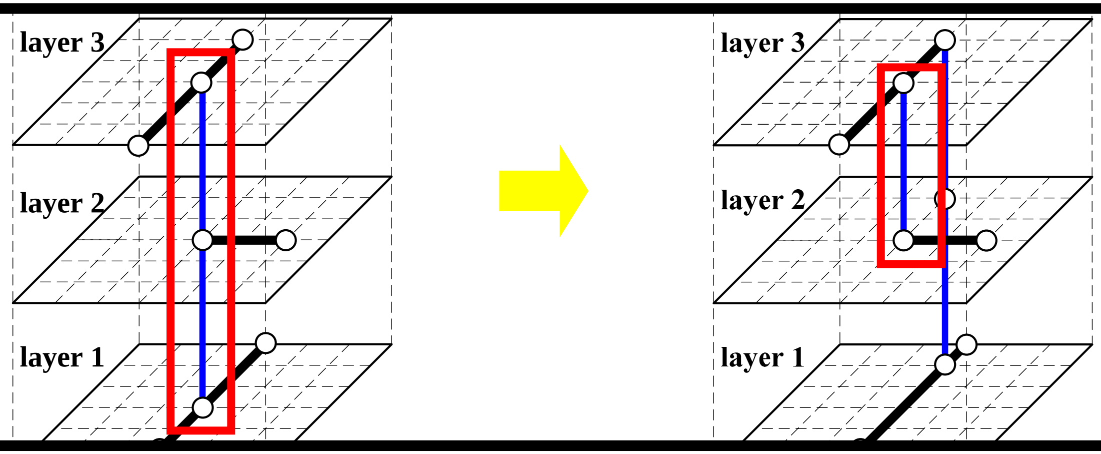
Preliminary of PDEG Optimality

- PDEG Optimality:
  At least one optimal solution exists in PDEG.
- The key idea of our proof-
  we can move an optimal solution to PDEG ***without increasing cost***.
  - In other words, we can move all line segments and via connections of an optimal solution to evading line segments and evading via connections without increasing cost.
- We only discuss basic concepts of the movement and give a rough sketch of our proofs.
- The key points are
  1. Two-stage movement – decide the order of movement
  2. Consecutive line segment – deals with line segments in different layers.

Two-Stage Movement

- Line segments and via connections: which one is first?
- Could we first move via connections? No

Line segments connected to this via connection have different directions



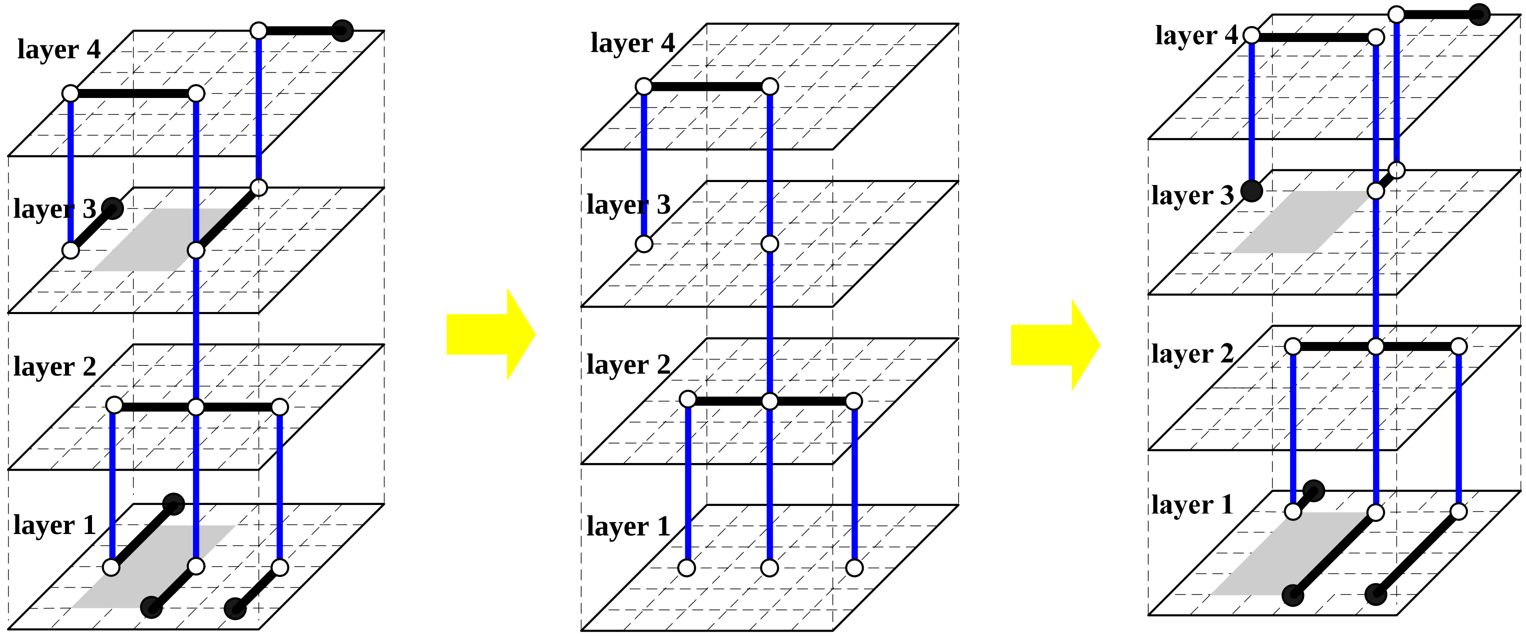
We can first move line segments and then move via connections.

- If we have moved all line segments to evading line segments, a via connection which is not contained by an evading via connection has all line segments with the same direction, i.e, we can move such a via connection

Consecutive Line Segment (CLS)

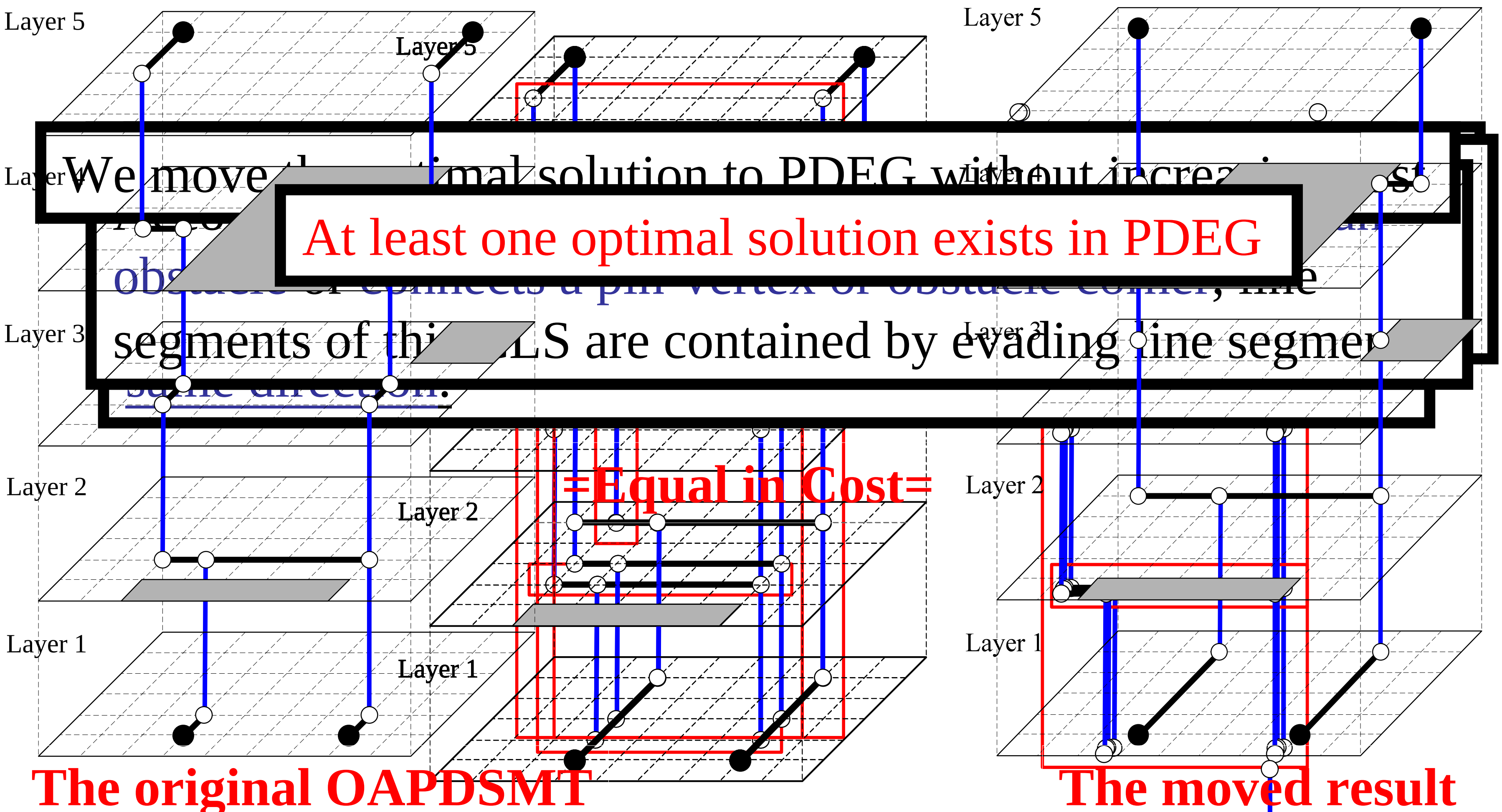
- A consecutive line segment (CLS) is a maximal connected component whose elements have the same either x-coordinate or y-coordinate.
- For a CLS, if line segments of a CLS are not contained by … d without increasing total cost.

This finding is the backbone of our movement.



Optimality of PDEG

- **If there exists an optimal solution for the OAPDST problem, at least one exists on PDEG.**



Benefits of Optimality Proof

- By the optimality of PDEG , we prove that even the OAPDST problem has many constraints, the optimal solution can be restricted in a simple graph.
- In other words, PDEG reduces the infinite geometry solution space to $O(n^2)$ graph. Hence, using PDEG as solution space, more efficient methods could be generated.
- However, PDEG is simple such that PDEG could be used intuitively. Our proof has further critical contributions.
  1. Makes the solutions generated from PDEG more convincing. (like Hanan Graph and Escape Graph)
  2. Provides a way to analyze solution quality, which helps to develop approximation algorithms or strong heuristics. (e.g, our approximation algorithm)

Outline

- Introduction
- Problem Formulation
- Routing Graph
- Approximation Algorithm
- Experimental Results
- Conclusion

Key Concepts of Approximation Algorithms

- Approximation algorithms are used to find near-optimal solutions for optimization problem.
- Unlike heuristics, an approximation solution has a theoretical guarantee from the optimal solution. Eg, Cost(MST) $\leqq$ 2*Cost(SMT).
- Hence, the bottleneck of designing an approximation algorithm is how to **compare the approximation solution with the optimal solution**, especially for geometry problems which have infinite solution space.
- In our observation, PDEG provides such a way to analyze solution quality. Hence, we can find that an obstacle-avoiding preferred direction minimum spanning tree (OAPDMST) is an approximation solution of the OAPDST problem.

MST and OAPDMST

- Given a graph, a minimum spanning tree (MST) connects all vertices with minimum cost.
- However, in preferred direction model, there are no edges between pin-vertices.
- In general , an edge connecting two pin-vertices is regards as an obstacle-avoiding preferred direction shortest path (OAPDSP) between them.
- In preferred direction model, an obstacle-avoiding preferred direction minimum spanning tree (OAPDMST) connects all pin-vertices by a set of OAPDSPs among pin-vertices such that the sum of costs of those OAPDSPs is minimum.

Existence Of OAPDMST (Theorem 2)

- **At least one OAPDMST exists in PDEG.**

Proofs:

- Since an OAPDMST consists of OAPDSPs, we only

We can compare Cost(OAPDSMT) with Cost(OAPDMST)

e

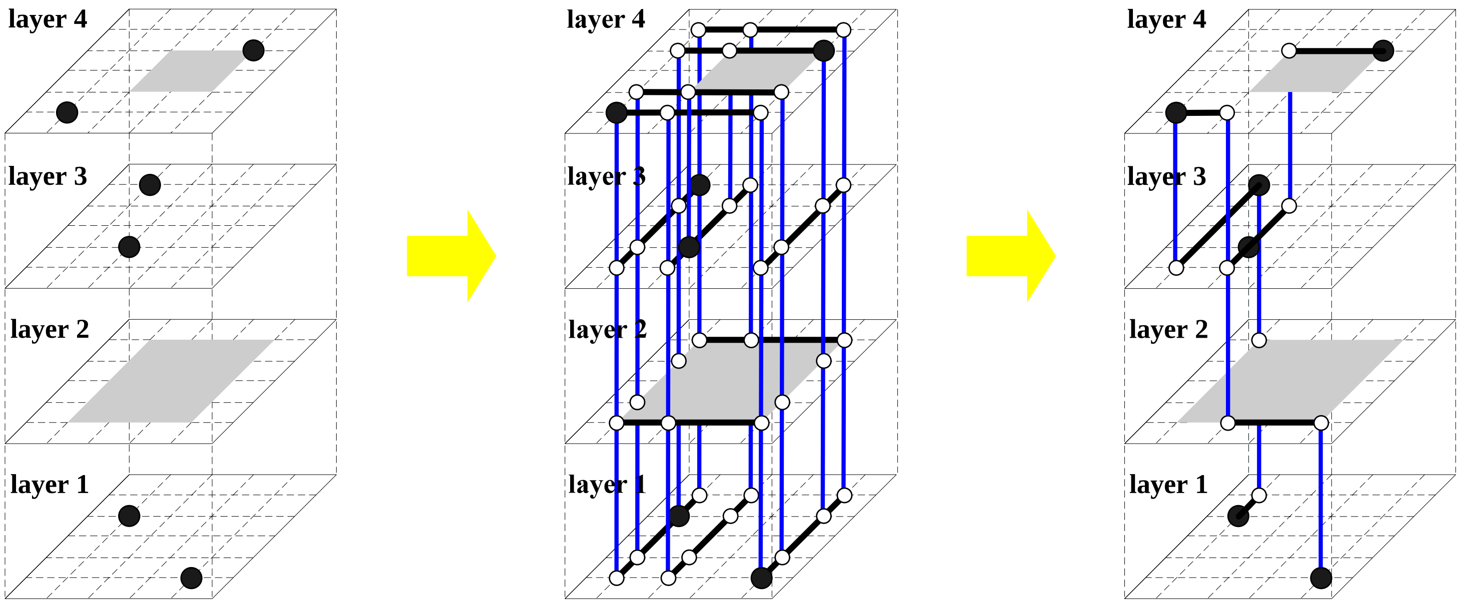
OAPDSP between them exists in PDEG.

- Assume s to be one OAPDSP between two pin-vertices.
- Clearly, we can move s to PDEG without increasing cost by the method for proving the optimality of PDEG.
- Hence, for any two pin-vertices, at least one OAPDSP exists in PDEG.
- To conclude, at least one OAPDMST exists in PDEG.

Approximation Algorithm

1. PDEG Construction: weight edges as follows:
    1. An edge e within a layer i: Cost(e) = length(e)*UC_i
    2. An edge e between layers: Cost(e) = C_v
2. OAPDMST Construction:
    - Apply an MST algorithm (Mehlhorn 1988) on PDEG.

- The total running time complexity is $O(n^2 \log n)$



Approximation Guarantee

- **Cost(OAPDMST) ≦ 2*Cost(OAPDSMT)**
  (i.e., approximation ratio of OAPDMST is 2)

Proof

- By the weight assignments, we directly have
  1. Cost(OAPDMST)=Cost(MST of PDEG)
  2. Cost(OAPDSMT)=Cost(SMT of PDEG)
- For a graph consisting of terminals and nonterminals
  - Cost(MST) ≦2*Cost(SMT)

(Takahashi and Matsuyama, *Math Japonica*, 1980)

Cost(OAPDMST)

≦

2*Cost(OAPDSMT)

Implication of our proof

- The proof for approximation guarantee shows that the cost of our approximation solution is not more than twice that of the optimal solution.
- Besides, our proof also has the following contributions.
  1. Give a strong motivation to develop more efficient OAPDMST algorithms. For instance, an O(nlogn) OAPDMST algorithm will be an O(nlogn) approximation algorithm for the OAPDST problem.
  2. Give some features to support strong heuristic methods just like some MST-based methods. MST-based methods are wildly used for the OARSMT problem.
- Without PDEG and proof of optimality, the approximation guarantee of OAPDMST may not be proved

Outline

- Introduction
- Problem Formulation
- Routing Graph
- Approximation Algorithm
- **Experimental Results**
- Conclusion

Construction-by-Correction Method

- Since there is no existing work for the OAPDST problem, we compare our algorithm with an extension of construction-by-correction methods.
- Construction-by-correction methods are used for the OARSMT problem and have the following two steps:
  - **Construction**: Construct a minimum spanning tree as an initial Steiner tree without considering obstacles.
  - **Correction:** Replace edges which intersect obstacles with edges around the obstacles.
- We denote the extension as **CC**.

Experiment for The Same Routing Resources

- Each kind of testcases has 100 sample.
- N_l=6, C_v=3, UC_i=1 for 1 $\leqq$ i $\leqq$ N_l

# of Pins	# of Obstacles	Total Cost			Time (sec.)	
		CC	ours	Imp(%)	CC	ours
20	4	4540672	3905891	13.98	0.005	0.005
40	8	6380169	5489595	13.96	0.025	0.023
60	12	7982803	6741823	15.55	0.063	0.059
80	16	9281025	7768205	16.30	0.122	0.106
100	20	10157322	8617818	15.16	0.203	0.164
200	40	14259141	12083500	15.26	1.009	0.682
400	80	19939542	16927531	15.11	4.978	2.869
600	120	24595604	20683857	15.90	12.772	7.117
800	160	29017940	23726589	18.23	18.253	13.236
1000	200	31723317	26301914	17.09	30.977	20.923
Average				15.65		

Experiment for Different Routing Resources$_1$

- We use a multi-factor (mf) to control different routing resources, i.e, $UC_i=mf^{N_l-i}$, e.g., mf=2, N_l=6, $UC_2=2^4=16$
- We set mf to be 1.1 and 2 as same as experiments in PDST model (Yildiz and Madden, TCAD, 2002)

# of Pins	# of Obstacles	Total cost (mf=1.1)			Total cost (mf=2)		
		CC	Ours	Imp(%)	CC	Ours	Imp(%)
20	4	5890803	4254719	27.77	14865082	9221914	37.96
40	8	8629405	5995774	30.52	22486709	13301225	40.85
60	12	10903119	7406311	32.07	31774473	18022028	43.28
80	16	12829877	8531095	33.51	37772561	21146323	44.02
100	20	14205003	9491634	33.18	41633938	23597230	43.32
200	40	20247211	13283910	34.39	58661268	32896792	43.92
400	80	28403784	18613525	34.47	82734886	46324734	44.01
600	120	34825326	22755618	34.66	102[illegible]	[illegible]5966	43.86
800	160	39423871	25626193	35.00	124022621	67353596	45.69
1000	200	47946950	29836773	37.77	150358576	77566875	48.41
	Average			33.33	Average		43.54

43.54

Experiment for Different Routing Resources$_2$

- The time results also show that our algorithm performs stably for the OAPDST problem, but CC does not.
- That is because the correction step of CC could take much more time. It also indicates that our algorithm is more suitable.

# of Pins	# of Obstacles	Total cost (mf=1.1)		Total cost (mf=2)	
		CC	Our	CC	Our
20	4	0.007	0.005	0.009	0.005
40	8	0.032	0.027	0.045	0.028
60	12	0.083	0.069	0.128	0.070
80	16	0.155	0.134	0.266	0.134
100	20	0.239	0.223	0.424	0.223
200	40	1.066	1.100	2.261	1.098
400	80	5.117	5.436	12.78	5.416
600	120	13.034	14.048	35.264	13.816
800	160	17.692	20.401	44.263	19.839
1000	200	31.991	33.993	75.823	32.24

Outline

- Introduction
- Problem Formulation
- Routing Graph
- Approximation Algorithm
- Experimental Results
- **Conclusion**

Conclusion and Future Work

- In this paper, we
  1. First formulate the OAPDST problem which considers more processing conditions and routability.
  2. Propose PDEG as a routing graph for the OAPDST problem, and prove at least one optimal solution on PDEG. Hence, PDEG not only reduce the solution space but also helps to analyze the solution quality.
  3. Develop a factor 2 approximation algorithm by our proof for the approximation guarantee of OAPDMST. This proof could be a backbone of future algorithms.
- In the future, we will
  1. Try to develop more efficient approximation algorithms or strong heuristic methods
  2. Apply some refinement schemes to improve solution quality

Thank You