Dim Sum: Light Clock Tree by Small Diameter Sum

Gengjie Chen and Evangeline F. Y. Young

CSE Department, The Chinese University of Hong Kong

Mar 26, 2019

Introduction

Clock routing

- **Skew**: maximum difference in signal arrival time among sinks
- **Wire length**

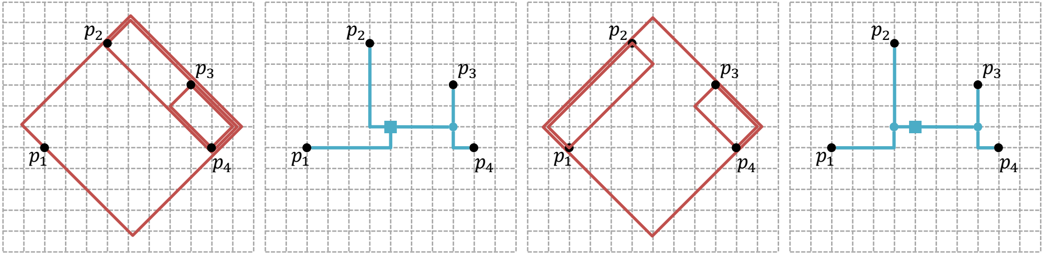
Zero-skew tree (ZST)

- Deferred-merge embedding (DME) gives optimal locations of Steiner points for a given (abstract) topology
- Expensive wire length $\implies$ path divergence (on-chip variations) & power
- Unnecessary due to tolerance & useful skew
- But can help building bounded-skew tree (BST)

Introduction

ZST T and hierarchical clustering (HC)

- Solution correspondence: treat $leaves(v)$ as a cluster, $v \in T$
- Cost equivalence: ZST **wire length** is a linear function of HC **diameter sum**
  - $length(T) = \frac{1}{2}(\sum_{v \in T} d(leaves(v)) + d(P))$



(a) HC1 (b) ZST1 (c) HC2 (d) ZST2

Introduction

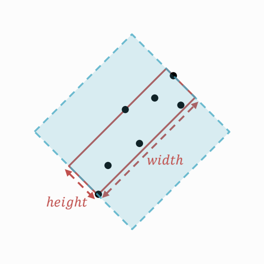
Our contributions

- **Equivalence** between ZST wire length and HC diameter sum
- ZST Construction
  - $O(n \log n)$-time $O(1)$-approximation algorithm
  - Optimal dynamic programming
- BST $=$ RSMT $+$ decomposition $+$ ZST
  - Linear-time optimal tree decomposition

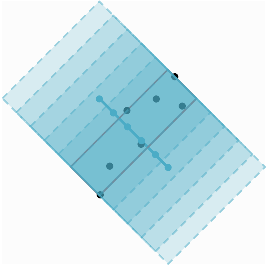
ZST Properties

The diameter $d(P)$ of points P is

- Diameter of Manhattan bounding circle (MBC) of P, or equivalently
- $\max_{p_1,p_2\in P} dist(p_1, p_2)$ in l_1



(a) An MBC for P

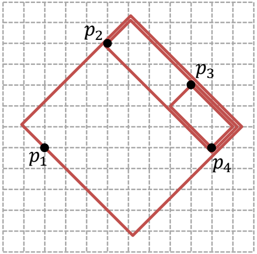


(b) Many MBCs for P

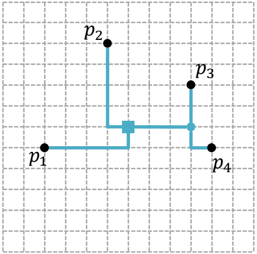
ZST Properties

ZST wire length is a linear function of HC diameter sum

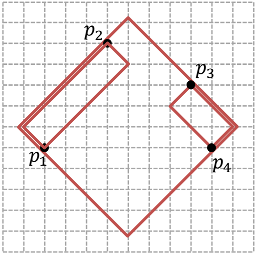
$$length(T) = \frac{1}{2}\left(\sum_{v \in T} d(leaves(v)) + d(P)\right)$$



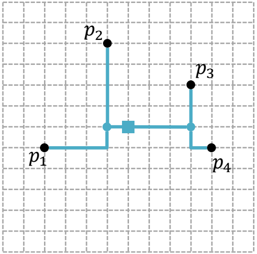
(a) HC with diameter sum $= d(\{p_3, p_4\}) + d(\{p_2, p_3, p_4\}) + d(P) = 4 + 10 + 10 = 24.$



(b) ZST corresponding to (a) with wire length $= \frac{1}{2}(4 + 10 + 10 + 10) = 17.$



(c) HC with diameter sum $= d(\{p_3, p_4\}) + d(\{p_1, p_2\}) + d(P) = 4 + 8 + 10 = 22.$



(d) ZST corresponding to (c) with wire length $= \frac{1}{2}(4 + 8 + 10 + 10) = 16.$

ZST Construction

Dim Sum: a simple iterative merging

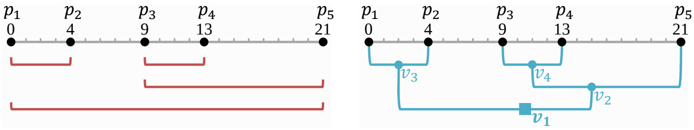
Input: Sinks P
Output: ZST T
1: Initialize clusters with each sink being a cluster by itself
2: **for** $i = 1, 2, ..., |P| - 1$ **do**
3: Merge the two clusters with the smallest diameter of their union among all pairs

ZST Construction

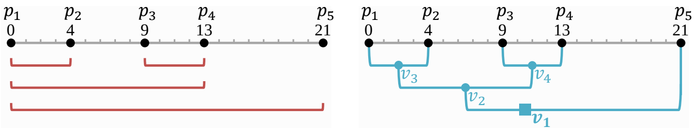
Dim Sum vs Greedy DME

- Both merges trees/clusters from bottom up
- Dim Sum prefers smaller diameter after merging
- Greedy DME prefers smaller distance between merging segments

Dim Sum
HC diameter sum
$= 4 + 4 + \underline{\mathbf{12}} + 21 = 41$
ZST wire length
$= 4 + 4 + \mathbf{10} + 13 = 31$



Greedy-DME
HC diameter sum
$= 4 + 4 + \mathbf{13} + 21 = 42$
ZST wire length
$= 4 + 4 + \underline{\mathbf{9}} + 14.5 = 31.5$



ZST Construction

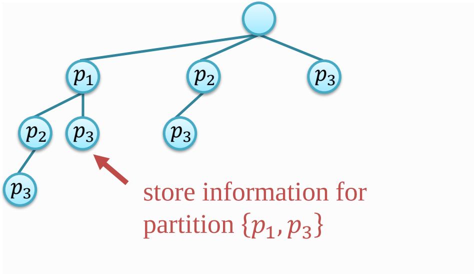
Dim Sum

- Actually the complete-linkage clustering method in l_1
- $O(1)$-approximation to ZST/HC
- $O(n\log(n))$ time

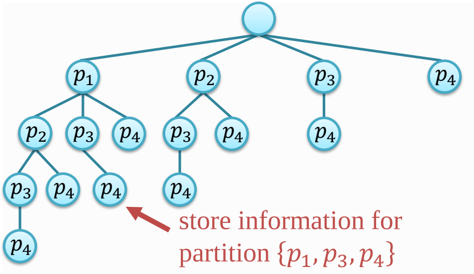
ZST Construction

Optimal Dim Sum: a trie-based dynamic programming

- Enumerate by adding sink after sink
- Store and retrieve optimal sub-ZST/HC cost by a trie (prefix tree)



(a) Trie for three sinks $\{p_1, p_2, p_3\}$



(b) Trie for four sinks $\{p_1, p_2, p_3, p_4\}$

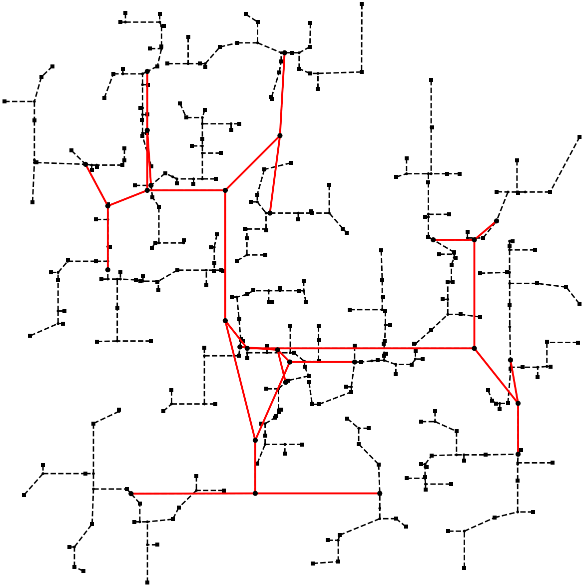
ZST Construction

Optimal Dim Sum

- Can be pruned by a cost upper bound (e.g., Dim Sum)
- Can be used to refine a sub-optimal solution (in linear time)

BST Construction

- BST = RSMT + decomp. + ZST
- **Tree decomposition**
  - Constrain maximum distance on each subtree
  - Optimal (minimum # subtrees)
  - Linear-time



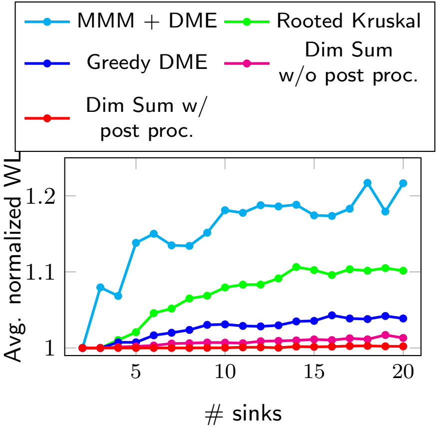
Experimental Results

Table: Wire Length Comparison of ZST Methods on Realistic Benchmarks (unit: μm)

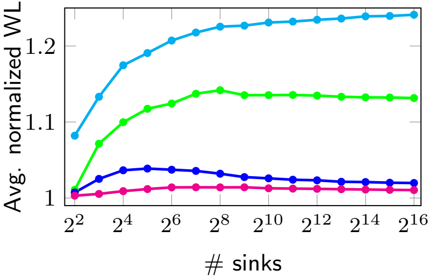
Benchmarks	DAC'90		TCAD'93					Avg. ratio
	p1	p2	r1	r2	r3	r4	r5	
# sinks	269	594	267	598	862	1903	3101	
MMM + DME	149	373	1601	3240	4165	8218	12274	1.248
Rooted Kruskal + DME	142	341	1461	2837	3711	7656	11052	1.136
GMA + DME	140	350	1497	3013	3902	7782	11665	1.173
BB + DME	141	361	1500	3010	3908	8000	11757	1.185
Greedy DME	133	314	1313	2566	3339	6707	9943	1.028
Dim Sum w/o refinement	131	309	1297	2568	3285	6631	9801	1.015
Dim Sum w/ refinement	**128**	**306**	**1272**	**2506**	**3248**	**6545**	**9711**	**1.000**

Benchmarks	ISPD'09											ISPD'10								Avg. ratio
	f11	f12	f21	f22	f31	f32	f33	f34	f35	fnb1	fnb2	1	2	3	4	5	6	7	8	
# sinks	121	117	117	91	273	190	209	157	193	330	440	1107	2249	1200	1845	1016	981	1915	1134	
MMM + DME	186	171	205	113	439	321	315	270	324	45.2	120	355	636	57.8	126	66.3	49.6	93.3	61.9	1.333
Rooted Kruskal + DME	190	171	202	116	420	314	317	257	290	41.5	110	306	564	31.2	99.8	45.6	38.5	71.1	44.2	1.138
Greedy DME	174	156	192	105	380	291	292	241	273	36.6	96.6	277	510	28.2	87.4	40.4	34.0	62.6	40.2	1.032
Dim Sum w/o refinement	176	157	184	106	375	283	287	238	268	35.0	95.4	273	504	27.8	86.6	40.0	33.3	61.5	39.2	1.016
Dim Sum w/ refinement	**171**	**153**	**182**	**103**	**369**	**281**	**282**	**236**	**264**	**34.7**	**93.6**	**269**	**496**	**27.6**	**85.5**	**39.0**	**32.7**	**60.6**	**38.8**	**1.000**

Experimental Results



(a) On small nets. WL normalized by **Optimal Dim Sum**.



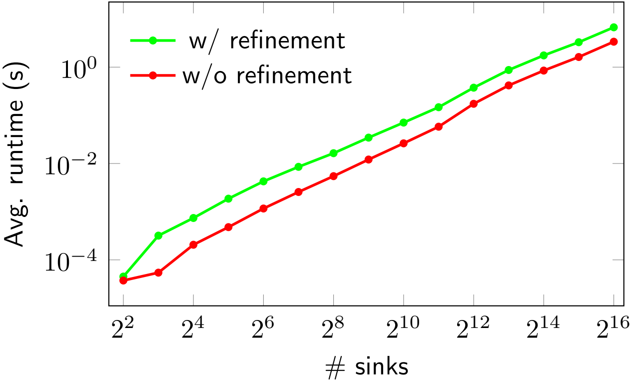
(b) On large nets. WL normalized by **Dim Sum** with post processing.

Wire length (WL) comparison of ZST methods on random nets

Experimental Results

Runtime of Dim Sum:

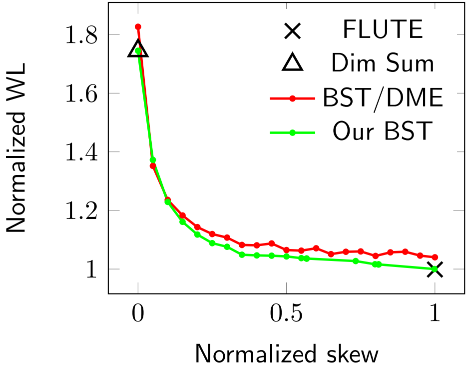
- $O(n \log n)$
- 3.4 seconds for 65536 sinks



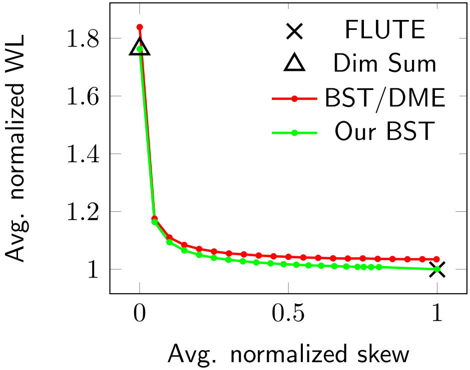
Experimental Results

Wire length of our BST method:

- Better Pareto frontiers compared with BST/DME
- Smooth trade-off between RSMT (FLUTE) and ZST (Dim Sum)



(a) WL on case r5 with 3101 sinks.

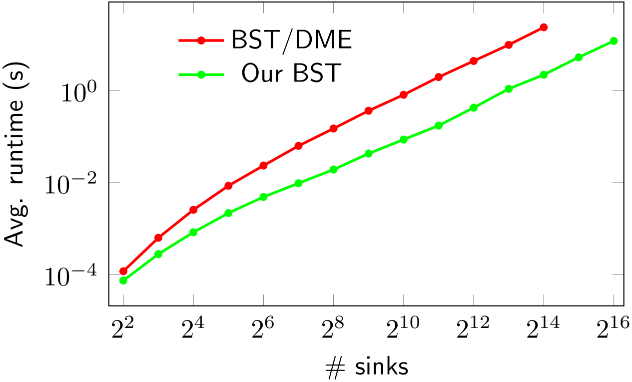


(b) WL on 100 random nets with 16384 sinks.

Experimental Results

Runtime of our BST method:

- Almost 10× speed-up compared with BST/DME
- 12 seconds for 65536 sinks



Conclusion

Equivalence between ZST wire length and HC diameter sum

- ZST: Dim Sum, Optimal Dim Sum
- BST $=$ RSMT $+$ optimal tree decomposition $+$ ZST

Future works

- Tight(er) bound for Dim Sum, e.g., 3 or 2?
- Better clustering methods instead of tree decomposition for BST?
- More practical factors, e.g., layer assignment, Elmore delay, etc?