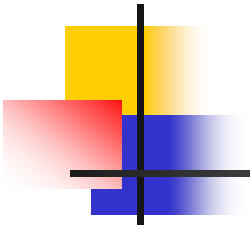
Logical and Physical Restructuring of Fan-in Trees

Hua Xiang Haoxing Ren Louise Trevillyan
Lakshmi Reddy[+] Ruchir Puri Minsik Cho

IBM T.J. Watson Research Center
[+]IBM EDA Lab. STG

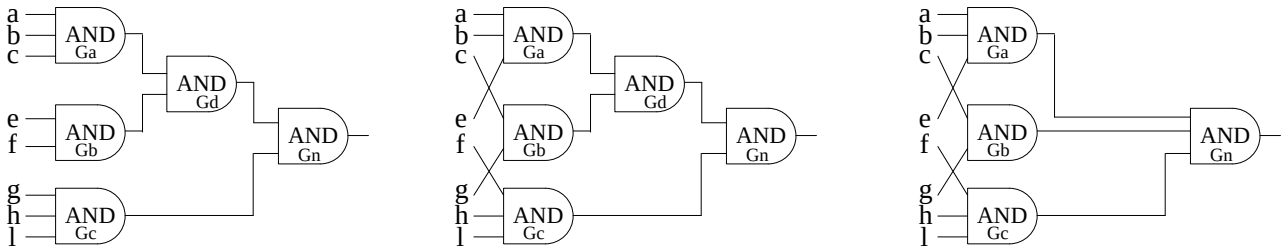


Introduction

- As feature sizes shrink and design complexity increases, the use of integrated logic synthesis and physical design is expanding.
- Tree restructure for SFFT (Symmetric-Function Fan-in Tree)
  - Trees are created during logic synthesis without placement information
  - Tree placement may be far from optimal since tree structure is fixed during physical design
  - Tree restructure can produce a more placeable and wire-efficient design.

Symmetric-Function Fan-in Tree (SFFT)

- A symmetric-function fan-in tree is a fanout-free logic cone
  - Compute a symmetric function
  - All of the leaf nets in its support set are commutative
  - The tree can be implemented with various structures of a uniform set of gates (i.e., AND, OR, XOR)



$$F = a \cdot b \cdot c \cdot e \cdot f \cdot g \cdot h \cdot l$$

- SFFT trees are frequently found in designs, especially when the design originated as two-level logic
- Rebuild SFFT trees based on the placement information to reduce wire length

Tree Restructure

- Algorithm outline

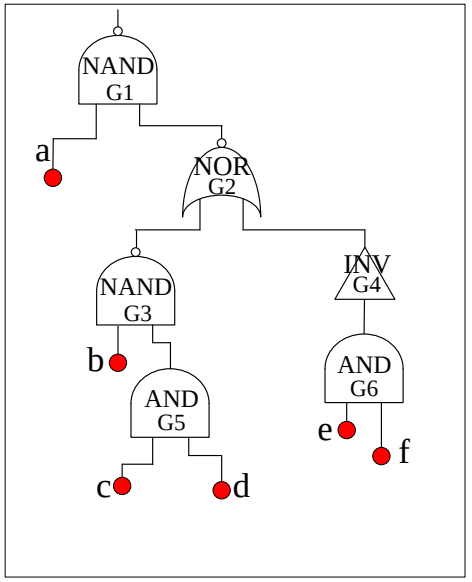
  SFFT_Restructure (***TreeRoot***, ***Threshold***)

  1. Identify SFFT tree with root ***TreeRoot***
  2. Get the locations of all leaf pins and root pin
  3. If #leaf pins < ***Threshold***
  4. Apply Dynamic programming based algorithm
  5. else
  6. Apply Steiner-Tree based restructure algorithm
  7. Output the new SFFT tree

AND SFFTs

- Possible gates in AND SFFTs
  - Any gates whose logic can be expressed by AND/NOT operations

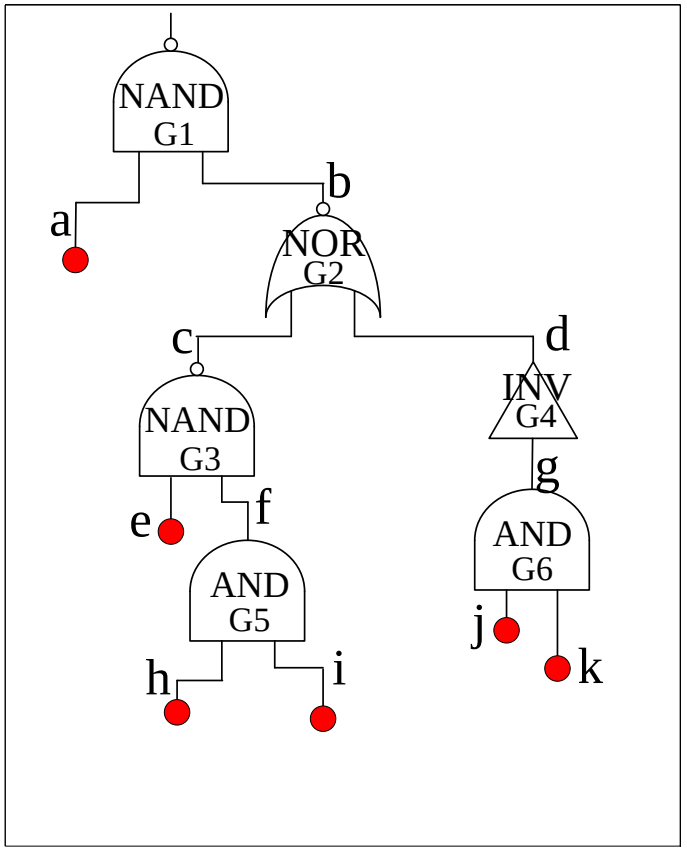
Gate Type	Function
AND	$F = a \cdot b$
NAND	$F = \overline{a \cdot b}$
OR	$F = \overline{\overline{a} \cdot \overline{b}}$
NOR	$F = \overline{a} \cdot \overline{b}$
NOT	$F = \overline{a}$
BUF	$F = a$

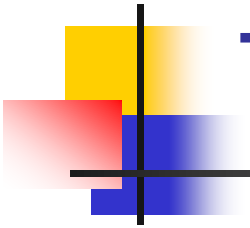


$$F = \overline{a \cdot b \cdot c \cdot d \cdot e \cdot f}$$

Stack-based Tree Identification

- Build look-up table to convert gates to AND/NOT gates
- Build the initial stack based on the tree root
- Traverse from top to bottom to identify tree nodes
- Traverse stops
  - Multiple-pin net
  - Infeasible gates
  - Invalid logic

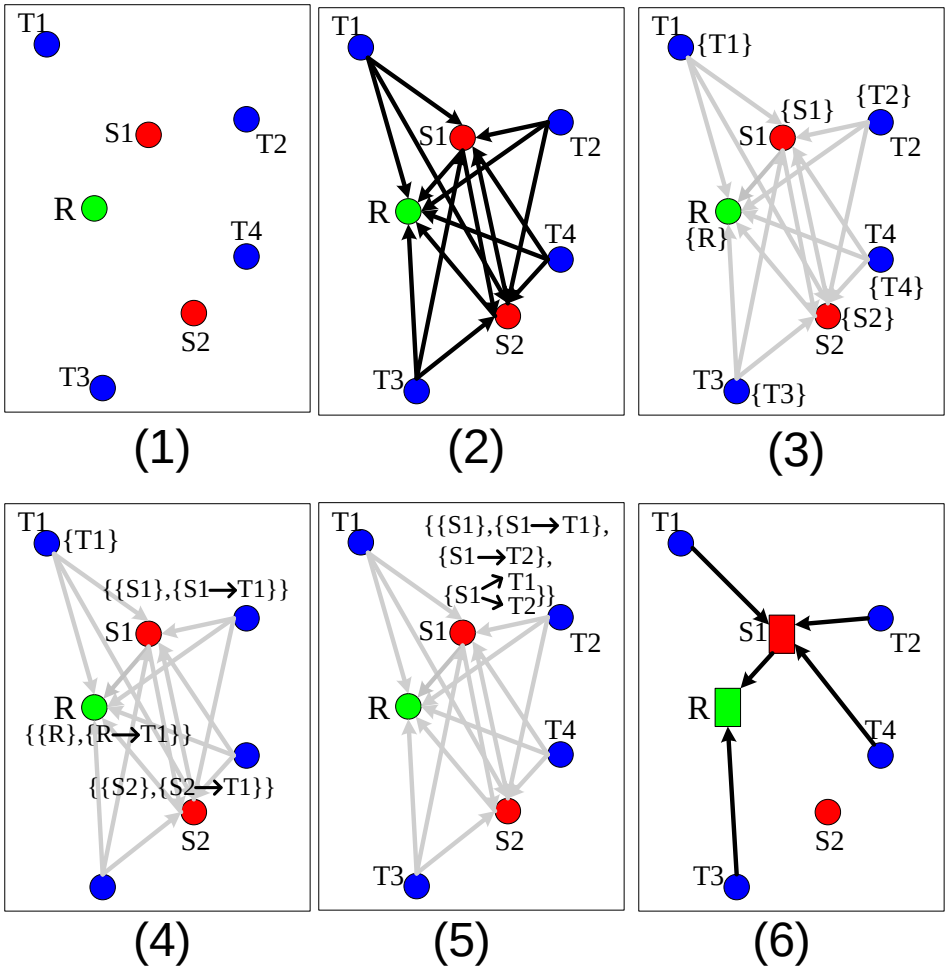




Tree Restructure

- Rebuild SFFT trees for shorter wire length
  - Keep the leaf input nets and root output net unchanged
  - Remove all internal gates/nets
  - Use AND/NOT gates to rebuild the tree
- Restructure algorithms
  - Dynamic programming based restructure
    - Optimal solution for a given tree
    - Long runtime
  - Steiner-Tree based restructure
    - The new tree follows the shape of a Steiner Tree for shorter wire length
    - Build sub-trees with DP based tree restructure
    - Balance quality and runtime

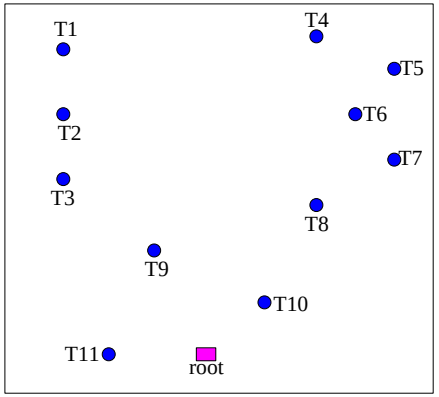
DP based Tree Restructure



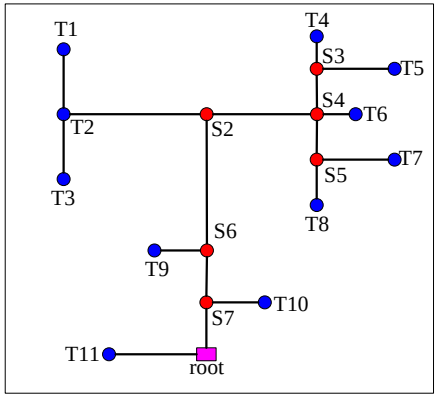
SFFT_DP (***TreeRoot***)

1. Build connectivity graph
2. Let each leaf/Steiner node be a tree, and push them to ***Queue***
3. While (!***Queue***.empty) {
4. *subtree* = ***Queue***.pop_front
5. for parent node ***P*** of *subtree*.*TreeRoot*
6. merge trees related to ***P*** with *subtree*
7. prune redundant trees
8. push the new trees in ***Queue***
9. }
10. Return a tree at ***TreeRoot*** with max leaf set and min wire length

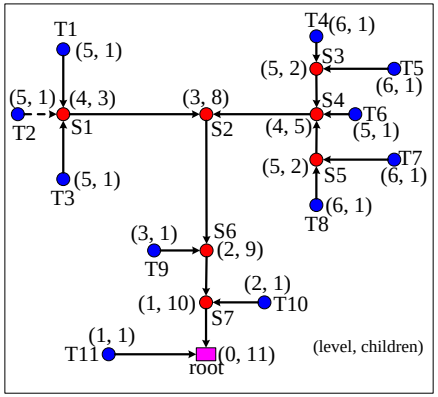
Steiner Tree based Restructure



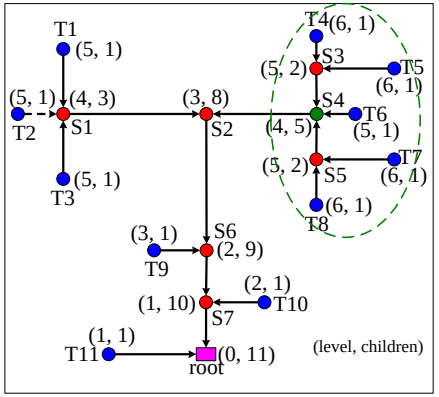
(1)



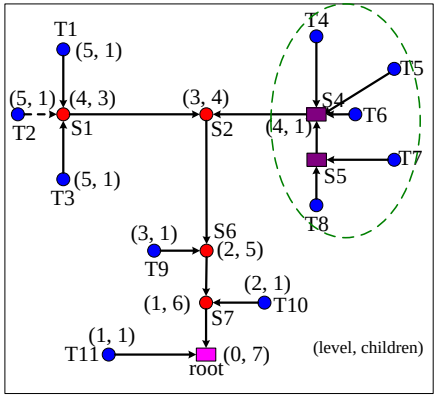
(2)



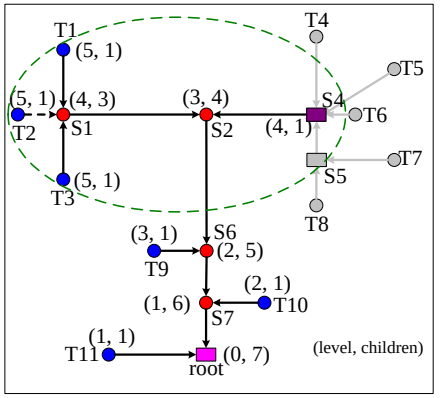
(3)



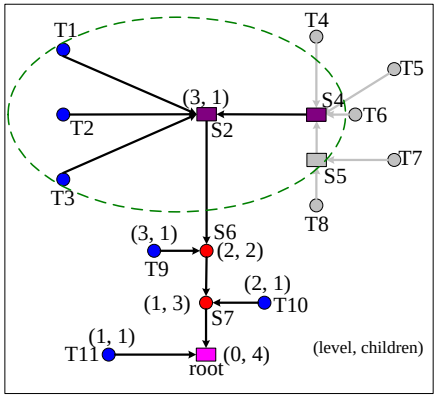
(4)



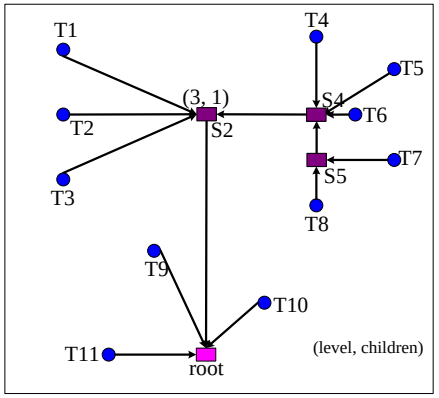
(5)



(6)



(7)



(8)

Experimental Results

- Algorithms were implemented in C, and have been integrated into a physical-synthesis system
- Tested on linux workstations (2.6GHz)
- Test cases were derived from industrial designs

Items	Test1	Test2	Test3
GateThreshold	4	4	4
SizeThreshold	8	8	11
Runtime	90s	131s	414s
ChangedNets	20129 (3.58%)	28954(3.84%)	90752 (8.15%)
TotalTrees	577	761	3617
MaxTree	709	471	243
ANDTrees	208	478	3535
ORTrees	369	283	82
XORTrees	0	0	0

Experimental Results (cont)

Restructuring Results with Re-placement

Items	Test1			Test2			Test3			Average
	Org	Restruct	Change	Org	Restruct	Change	Org	Restruct	Change	Change
STWL	4767328	3928756	-17.59%	8387826	4954639	-40.93%	55900403	34348299	-38.55%	-32.36%
TWL	61964038	60440335	-2.46%	73155265	71641961	-2.07%	125058151	112698170	-9.88%	-4.80%
AEC	47.43	46.17	-2.66%	54.35	53.34	-1.86%	60.36	55.75	-7.63%	-4.05%
ANC	49.31	49.03	-0.56%	45.31	44.79	-1.15%	64.57	61.63	-4.56%	-2.09%
AEC20	83.00	82.25	-0.91%	93.49	92.12	-1.47%	108.32	101.06	-6.70%	-3.03%
ANC20	88.43	87.09	-1.51%	96.70	95.43	-1.31%	133.60	127.61	-4.49%	-2.44%
Net90	53463	44543	-16.68%	123313	118995	-3.50%	316319	266259	-15.83%	-12.00%
Net100	12211	6339	-48.09%	68033	57473	-15.52%	264194	214013	-18.99%	-27.53%

STWL: Total wire length of SFFT trees
TWL: Total wire length
AEC: Average edge congestions
ANC: Average net congestions
AEC20: AEC20: Average Edge Congestions of top 20% congested edges
ANC20: AEC20: Average Net Congestions of top 20% congested nets
Net90: Number of nets whose congestion ≥ 90%
Net100: Number of nets whose congestion ≥ 100%

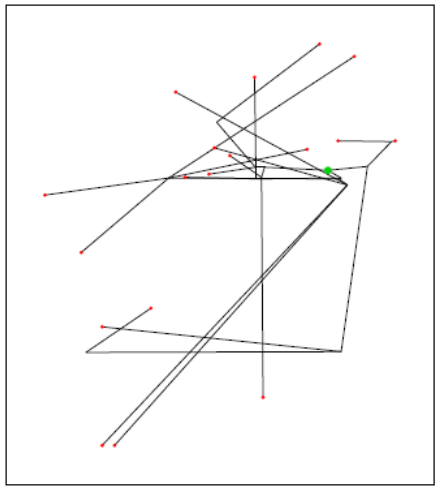
Experimental Results (cont)

Restructuring Results with Legalization

Items	Test1			Test2			Test3			Average
	Org	Restruct	Change	Org	Restruct	Change	Org	Restruct	Change	Change
STWL	4767328	2763617	-42.03%	8387826	4276455	-49.02%	55900403	22363738	-59.99%	-50.35%
TWL	61964038	61276845	-1.11%	73155265	71909245.84	-1.70%	125058151	118369793	-5.35%	-2.72%
AEC	47.43	47.06	-0.77%	54.35	53.62	-1.35%	60.36	58.09	-3.75%	-1.96%
ANC	49.31	48.89	-0.84%	45.31	44.57	-1.62%	64.57	62.94	-2.53%	-1.66%
AEC20	83.00	82.77	-0.28%	93.49	92.75	-0.79%	108.32	104.12	-3.88%	-1.65%
ANC20	88.43	88.08	-0.39%	96.70	96.02	-0.70%	133.60	129.15	-3.33%	-1.47%
Net90	53463	51665	-3.36%	123313	118856	-3.61%	316319	297009	-6.10%	-4.36%
Net100	12211	10290	-15.73%	68033	63601	-6.51%	264194	240637	-8.92%	-10.39%

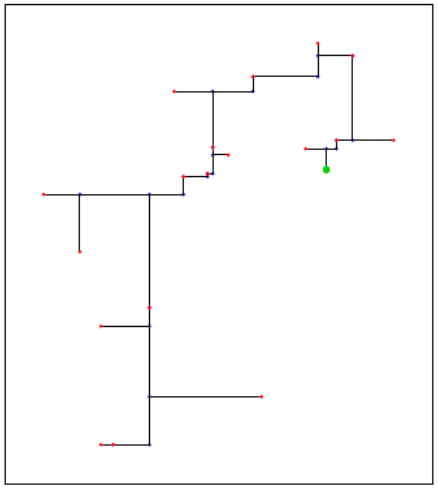
STWL: Total wire length of SFFT trees
TWL: Total wire length
AEC: Average edge congestions
ANC: Average net congestions
AEC20: AEC20: Average Edge Congestions of top 20% congested edges
ANC20: AEC20: Average Net Congestions of top 20% congested nets
Net90: Number of nets whose congestion ≥ 90%
Net100: Number of nets whose congestion ≥ 100%

Experimental Results (cont)



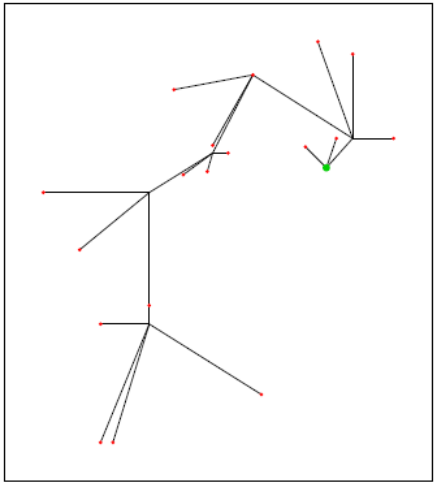
(a) Original Tree

Wire Length: 5627



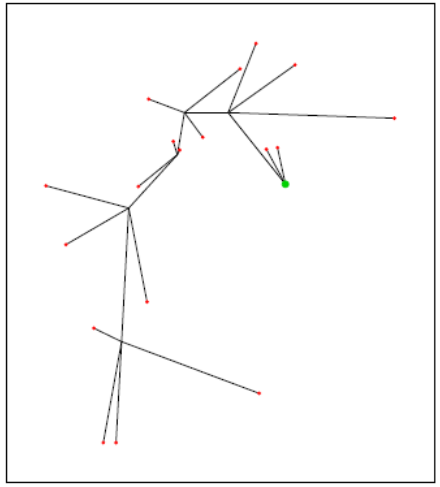
(b) Steiner Tree

Wire length: 1594



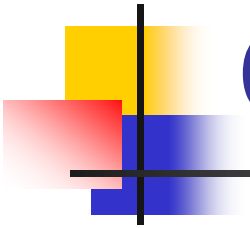
(c) Restructured Tree

Wire Length: 2719



(d) Legalized Tree

Wire Length: 2523



Conclusion

- SFFT tree is a fanout-free cone of logic that computes a symmetric function, so that all of the leaf nets in its support set are commutative.
- Propose efficient algorithms to identify SFFT trees and restructure them.
- SFFT tree restructure helps to get better tree shapes, reduce wire length and congestion on some designs.