Simultaneous Timing Driven Tree Surgery in Routing with Machine Learning-based Acceleration

Peishan Tu, Chak-Wa Pui, Evangeline F. Y. Young

Department of Computer Science and Engineering,
The Chinese University of Hong Kong

May 25, 2018

Outline

Outline

Previous Work

- Global routers without timing consideration [NCTUgr] [NTHURoute] [FastRoute] [MaizeRouter]
- Timing aware global routers optimized delay independently [Jiang00] [Tong03] [Minsik07] [Youssef10]
- Timing driven single net routing algorithms with the lumped resistance driver model [Genjie17] [Charles18]

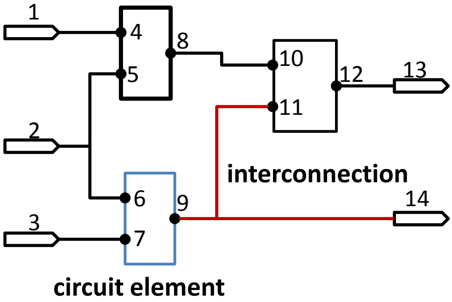
Motivation

- A simple and efficient technique to improve timing of timing-unaware global routers
- Optimize timing of the circuit using a better driver model
- Current driver model: Non Linear Delay Model(NLDM) and Composite Current Source (CCS) is sensitive to input slew and output load. The changing topology of net i will influence net i downstream gate delay

Outline

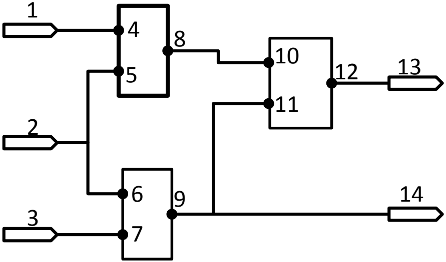
Preliminary-Example

An example



A simple Circuit

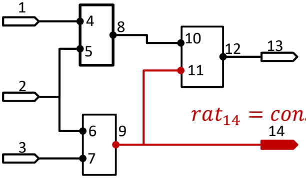
Preliminary-Slack



$$slack_{13} = rat_{13} - aat_{13}$$

$$slack_{14} = rat_{14} - aat_{14}$$

Preliminary-RAT & AAT

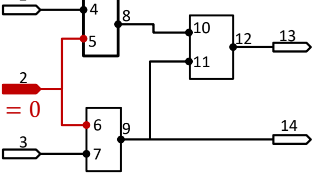


$rat_{14} = const$

$rat_9 = \min(rat_{11} - d_{9\to 11}, rat_{14} - d_{9\to 14})$

RAT-Wire

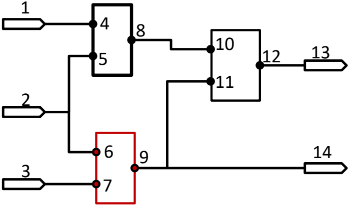
$aat_5 = aat_2 + d_{2\to 5}$



$aat_2 = 0$

$aat_6 = aat_2 + d_{2\to 6}$

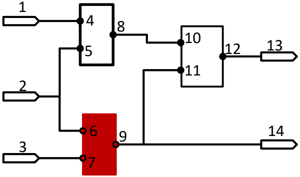
AAT-Wire



$rat_6 = rat_9 - gd_{6\to 9}$

$rat_7 = rat_9 - gd_{7\to 9}$

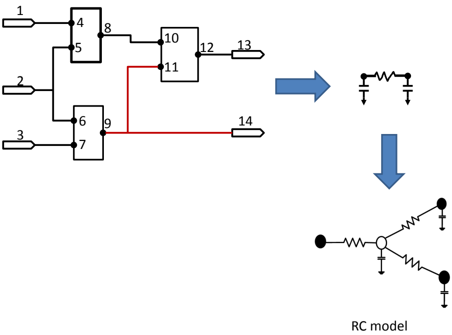
RAT-Gate



$aat_9 = \max(aat_6 + gd_{6\to 9}, aat_7 + gd_{7\to 9})$

AAT-Gate

Preliminary-Interconnect Delay



1. Interconnect delay calculation

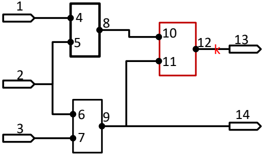
$$d_{S \to T_1} = \sum_{k \in N} R_{k \to T_1} C_k \quad (1)$$

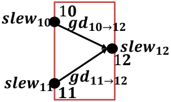
Preliminary-Circuit Element Delay

1.circuit element delay

slew y / capacitance x	y_1	y_2	y_3
x_1	z_{11}	z_{12}	z_{13}
x_2	z_{12}	z_{22}	z_{23}
x_3	z_{13}	z_{23}	z_{33}

$$L(x, y) = a_0 + a_1 x + a_2 y + a_3 xy \quad (2)$$



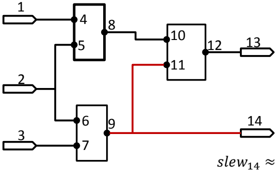


$gd_{10\to12}$=L($capacitance_k, slew_{10}$)

$gd_{11\to12}$=L($capacitance_k, slew_{11}$)

Preliminary-Slew

1.Interconnect Slew

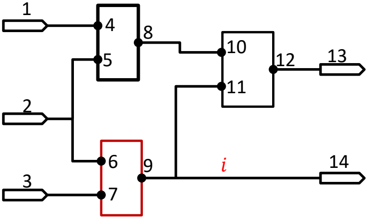


$$slew_{14} \approx \sqrt{slew_9^2 + imp_{14}^2}$$

$$imp_{14} \approx \sqrt{2\beta_{14} - d_{9\to 14}^2}$$

$$\beta_{14} = \sum_{k\in N} R_{k\to 14} C_k d_{9\to 14}$$

2.Circuit Element Slew



$$slew'_6 = L'(capacitance_i, slew_6)$$

$$slew'_7 = L'(capacitance_i, slew_7)$$

$$slew_9 = \max(slew'_6, slew'_7)$$

Outline

Problem Formulation

If the slack information of each timing end point is positive, there is no timing problem in the circuit. To achieve that, we maximize the summation of slacks by reconnecting the critical sink.

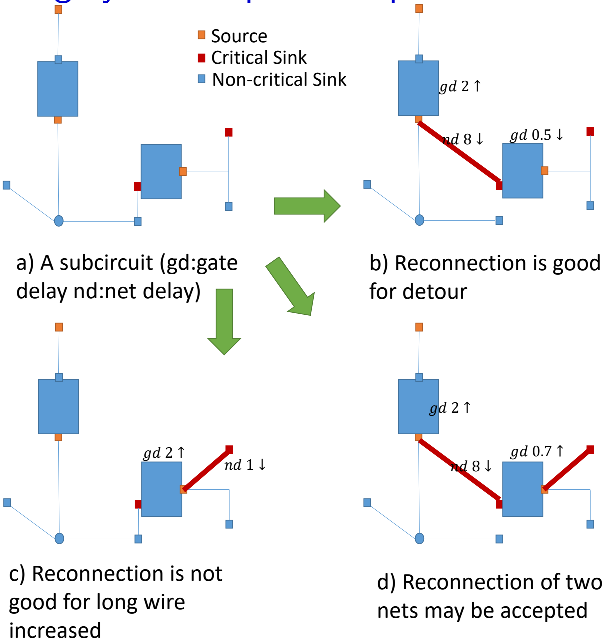
$$\begin{aligned} \max \quad & \mathrm{TNS}, \\ s.t. \quad & x_i \in \{0, 1\} \quad \forall i \in N_c, \end{aligned} \tag{3}$$

where x_i denotes whether the critical sink of each net $i \in N_c$ is reconnected
the pin will be reconnected of net $i \in N_c$:

- most negative slack
- parent is not source

Outline

Tree Surgery Technique-Example



Tree Surgery Technique - QP-based TST (QPTST)

Maximize source slack:

$$\begin{aligned} slack_S &= rat_S - aat_S \\ &= rat_{T_1} - d_{S\to T_1} - aat_S - gd_{l\to S} \\ &= rat_Y - aat_s - \boxed{(gd_{l\to S} + d_{S\to T_1})} \end{aligned} \quad (4)$$

minimize delay $\rightarrow$ maximize delay reduction

Tree Surgery Technique-QPTST

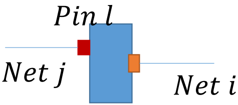
Objective function:

$$\max \sum_{i=1}^{n} (\Delta d_{s \to j_x^i} x_i + \beta \Delta L_i) \qquad (5)$$
$$s.t. \quad x_i = \{0, 1\} \quad \forall i \in N$$

Difference of interconnect delay:

$$\Delta d_{s \to j_x^i} = d^o_{s \to j_x^i} - d_{s \to j_x^i} \qquad (6)$$

Difference of gate delay:



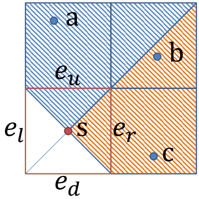
$$\begin{aligned} \Delta L_i =& L(cap_i^o, slew_l^o) - L(cap_i, slew_l) \\ =& a_1(cap_i^o - cap_i) + a_2(slew_l^o - slew_l) \\ &+ a_3(cap_i^o slew_l^o - cap_i slew_l) \end{aligned} \qquad (7)$$

$$\begin{aligned} \Delta L_i =& (a_1 + a_3 slew_l^o) \Delta cap_i x_i + (a_2 + a_3 cap_i^o) \Delta slew_l x_j \\ &- a_3 \Delta cap_i \Delta slew_l x_i x_j \end{aligned} \qquad (8)$$

Congestion Aware QPTST

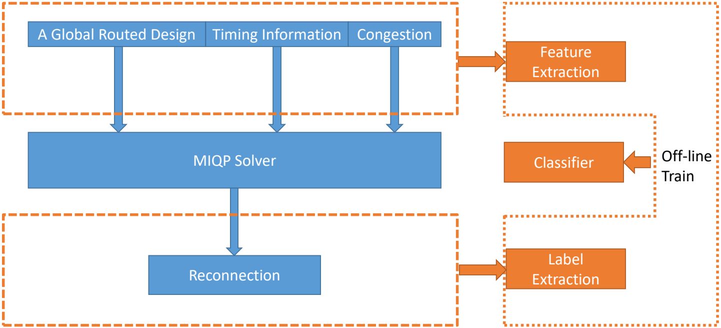
By adding **overflow penalty** po_i into the objective function, we can optimize timing and congestion simultaneously.

$$\max \sum_{i=1}^{n} (\beta \cdot \Delta L_i + \Delta d_{s \rightarrow j_x^i} x_i) + \boldsymbol{\alpha \cdot po_i \cdot x_i} \quad (9)$$
$$s.t. \quad x_i = \{0, 1\} \quad \forall i \in N_c$$



An example of how to calculate potential routing overflow.

Machine Learning-based Acceleration



-Solving MIQP($X^TAX + b^TX$)($|X| = N \approx 1million$ and A is very sparse) takes long time but prediction by ML is very quick($f^B(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x)$)($|x| = d$ and it may calculate $|dBN|$.)

-Our problem can be formulated as binary classification problem

Outline

Benchmark Information

Designs	#nodes	#nets	clock periods (ns)
superblue10	1876103	1898119	10
superblue1	1209716	1215710	9
superblue16	981559	999902	5.5
superblue18	768068	771542	7
superblue3	1213253	1224979	10
superblue4	795645	802513	6
superblue5	1086888	1100825	9
superblue7	1931639	1933945	5.5

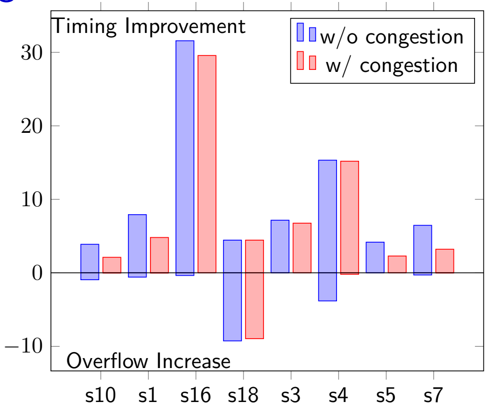
Experimental Results of Tree Surgery Technique.

Benchmarks	FLUTE Baseline**						Direct Connection*		
	WNS	r_wns	TNS	r_tns	stWL	r_stwl	r_wns	r_tns	r_stwl
superblue10	-1.65	0.00	-33.10	0.00	2.05	0.00	0.47%	2.47%	-13.92%
superblue1	-0.50	0.00	-0.46	0.00	0.96	0.00	-0.26%	3.20%	-19.76%
superblue16	-0.46	0.00	-0.76	0.00	0.93	0.00	3.58%	25.18%	-14.74%
superblue18	-0.46	0.00	-1.03	0.00	0.58	0.00	-0.75%	2.10%	-23.30%
superblue3	-1.01	0.00	-1.50	0.00	1.14	0.00	4.82%	5.79%	-18.87%
superblue4	-0.62	0.00	-3.47	0.00	0.71	0.00	0.90%	10.81%	-18.51%
superblue5	-2.57	0.00	-6.95	0.00	1.08	0.00	0.07%	1.42%	-17.24%
superblue7	-1.51	0.00	-1.84	0.00	1.40	0.00	0.00%	3.54%	-23.56%
Average	-1.10	0.00	-6.14	0.00	1.11	0.00	1.10%	6.81%	-18.74%

Benchmarks	QPTST					Congestion Aware QPTST				
	r_wns	r_tns	r_stwl	r_d	CPU(s)	r_wns	r_tns	r_stwl	r_d	CPU(s)
superblue10	0.92%	3.88%	-0.79%	11.52%	69.73	0.00%	2.11%	-0.48%	11.80%	77.03
superblue1	1.76%	7.92%	-0.38%	16.18%	15.60	1.78%	4.81%	-0.37%	16.19%	25.05
superblue16	3.94%	31.58%	-0.38%	12.52%	6.42	3.94%	29.57%	-0.36%	12.54%	15.21
superblue18	2.27%	4.45%	-0.18%	18.75%	17.93	2.27%	4.45%	-0.18%	18.75%	13.01
superblue3	5.61%	7.16%	-0.11%	15.78%	6.12	5.37%	6.76%	-0.09%	15.80%	15.10
superblue4	1.60%	15.33%	-1.79%	14.10%	48.95	0.47%	15.19%	-1.58%	14.29%	57.36
superblue5	0.32%	4.17%	-0.62%	14.18%	11.93	0.12%	2.29%	-0.27%	14.48%	22.02
superblue7	0.00%	6.46%	-0.13%	18.96%	44.13	0.00%	3.21%	-0.08%	19.00%	20.91
Average	2.05%	10.12%	-0.55%	15.25%	27.60	1.74%	8.55%	-0.43%	15.35%	30.71

- *Direct Connection: directly connect the critical sinks to the source for all nets.
- **WNS is in $10^4 ps$. TNS is in $10^6 ps$. stWL is in $10^8 um$.

Results-Performance Analysis on Timing and Routing Congestion



Timing Improvement: TNS improvement compared with the initial routing result

Overflow Increase: Routing overflow increased compared with the initial routing result

Blue is QP result and red is congestion aware result

Experimental Results of Machine Learning Acceleration (MLA).

Benchmarks	ML			ML Over Base			ML Over QP		
	ACC	CPU(s)	QP-CPU(s)	r_wns	r_tns	r_wl	r_wns	r_tns	r_wl
superblue18	97.13%	1.21	18.65	0.09%	3.89%	-0.18%	-0.01%	0.01%	0.00%
superblue16	95.53%	1.54	6.31	5.24%	29.54%	-0.32%	0.10%	0.14%	0.00%
superblue7	94.91%	1.59	45.63	0.00%	5.49%	-0.13%	0.00%	0.41%	0.00%
superblue4	99.14%	6.39	49.13	1.55%	13.87%	-1.78%	-0.02%	0.11%	0.00%
superblue1	82.74%	2.98	14.87	1.71%	6.10%	-0.38%	0.42%	1.28%	-0.06%
superblue3	82.38%	1.13	6.16	3.67%	5.32%	-0.10%	0.37%	1.26%	-0.05%
superblue5	83.19%	2.97	11.87	0.25%	3.40%	-0.57%	0.05%	0.06%	-0.01%
superblue10	87.99%	7.73	61.68	0.73%	3.71%	-0.76%	0.07%	0.55%	-0.08%

ACC: classfication accuracy **CPU**:ML runtime **QP-CPU**:our solver runtime

ML Over Base:timing result of classifier compared with the initial routing solution

ML Over QP:timing result of classifier compared with our solver solution

Outline

Contribution

- To optimize the tree topologies globally, a QP is formulated to determine how to adjust the most critical sink connection to optimize timing and congestion.
- We study various circuit properties and identified those that contribute to timing. Later, these features will be used to accelerate the QP-based tree surgery technique by a machine learning-based technique.
- Experimental results show that we can improve timing of the design significantly with small increase in routing congestion.

Thanks!