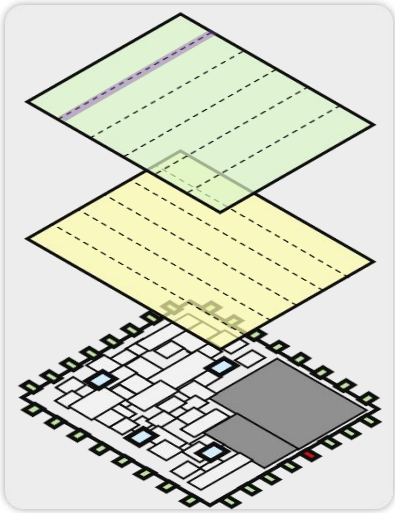
Clock Tree Synthesis Tutorial and iCTS Software

Weiguo Li
Minnan Normal University
dawnli619215645@gmail.com

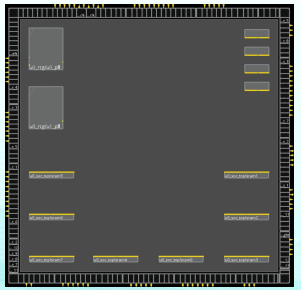
Overview

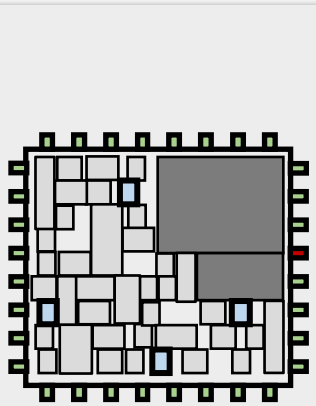
Where's CTS

Digital EDA	Tools
RTL design	Logic compiling
Simulation /Verification	Logic optimization
Logic synthesis	Tech mapping
Formal	Floorplan
Physical design	Placement
Sign-off	**CTS**
Physical verification	Routing
Layout synthesis	Design optimization
.....	STA
	Power analysis
	Parasitic extraction
	EM / IR
	DRC / ERC
	LVS
	.....

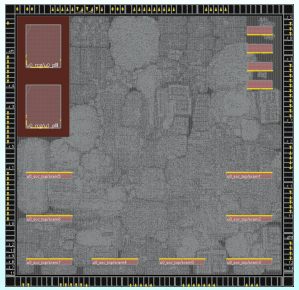


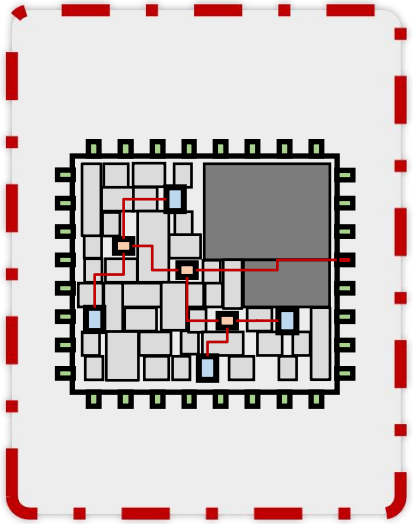
Floorplan determining the area of the Die and Core, and placing the macro cell.



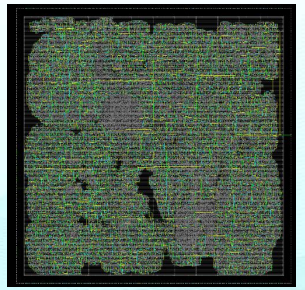


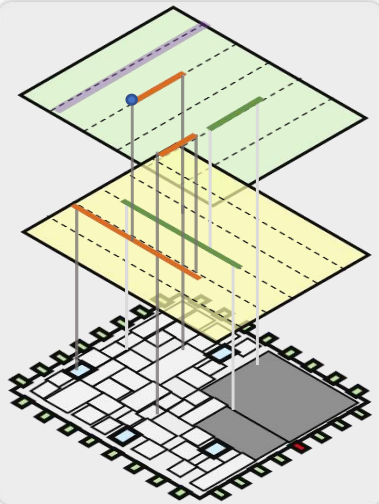
Placement placing standard cells into legal positions to achieve the optimal design object.



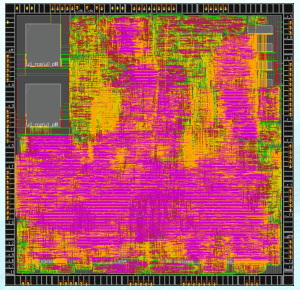


CTS designs a clock network, ensuring clock synchronization, achieving low-power and high-performance.

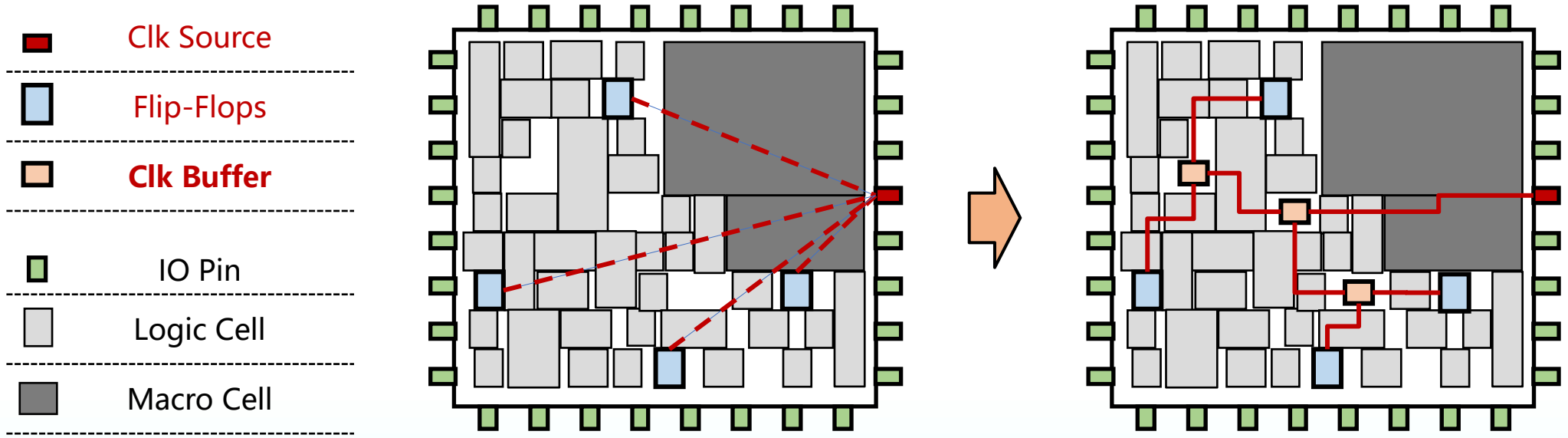




Routing materializes all nets to keep the wirelength as short as possible and adhere to the design rules.

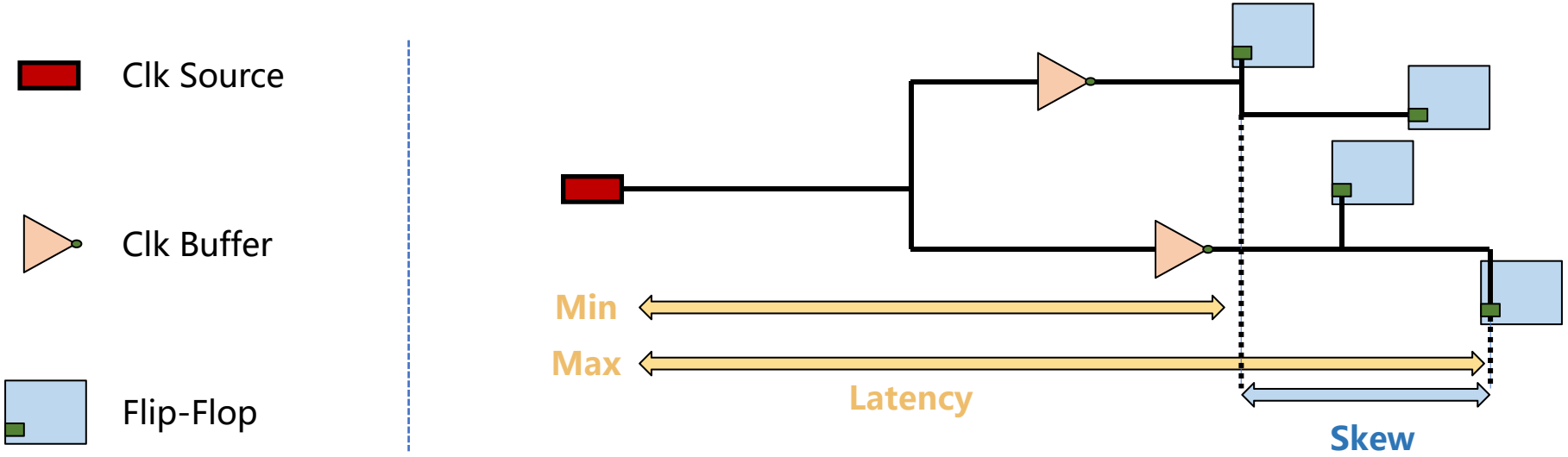


What's CTS?



- CTS, the bridge between Placement and Routing
- Achieving **skew balance** and **minimize design resource** (buffers, wirelength)

What's our concern



- Timing of Clock Tree
  - skew
  - latency
  - worst negative slack (WNS)
  - total negative slack (TNS)
  - …
- Design Resource
  - num of buffer insertions
  - wirelength
  - capacitance
  - power/area
  - …

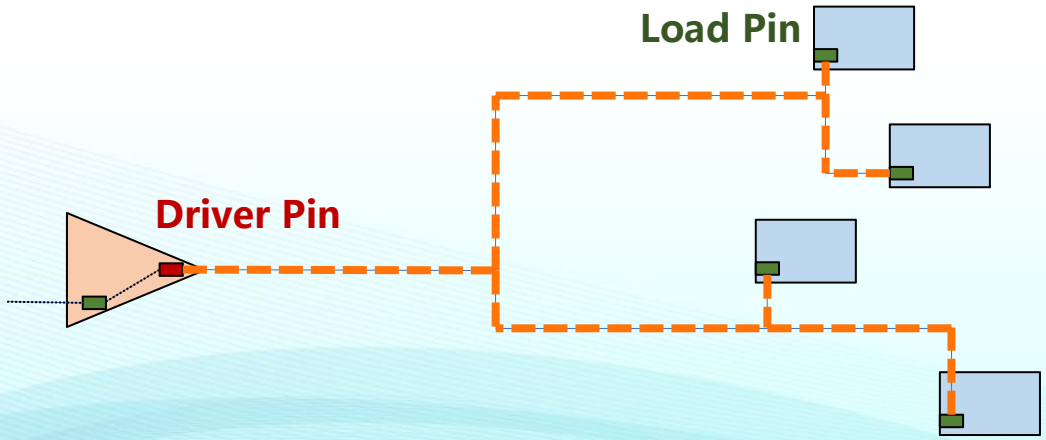
What's the operation in CTS

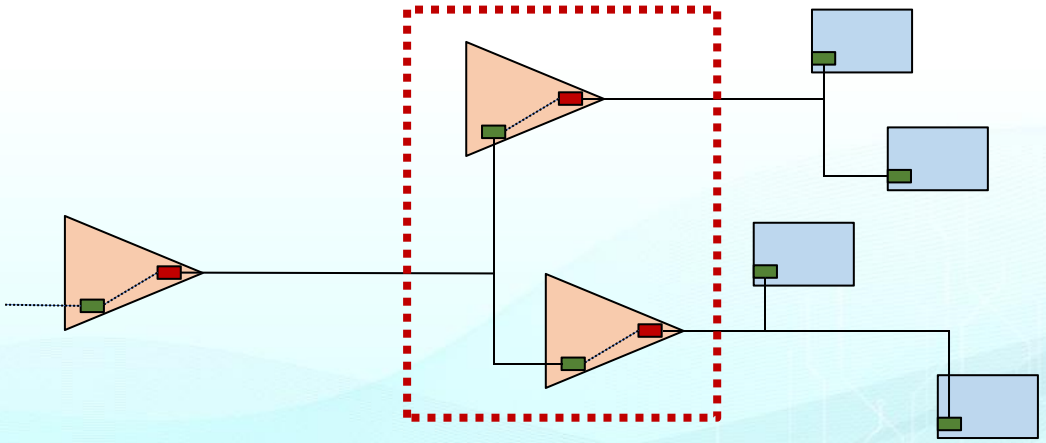
- Routing Topology
  - starting from the **driver pin**
  - connect all **load pins**
  - Steiner tree properties

  > *delay/wirelength/capacitance/...*

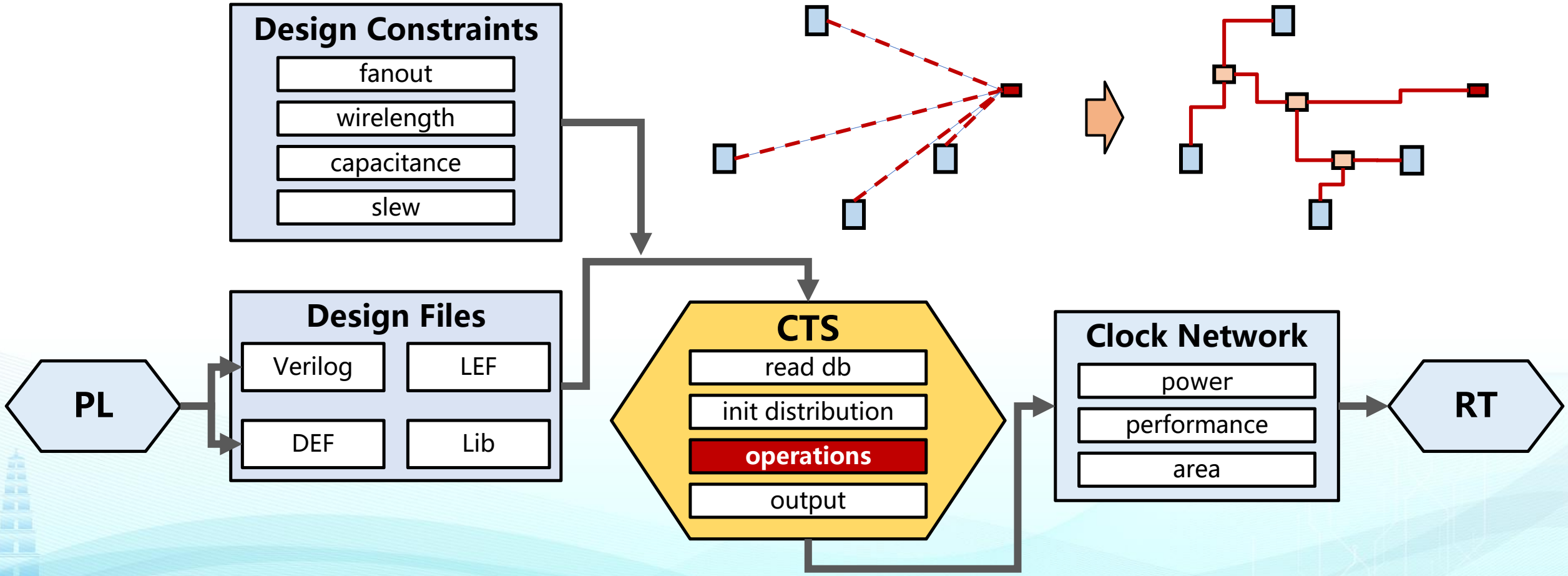
- Buffering
  - decompose the net
  - reduce fanout (4 → 2)
  - enhanced driving capability

  > *cell delay/area/power/slew/…*





What's we expecting



Overview

CTS Formula

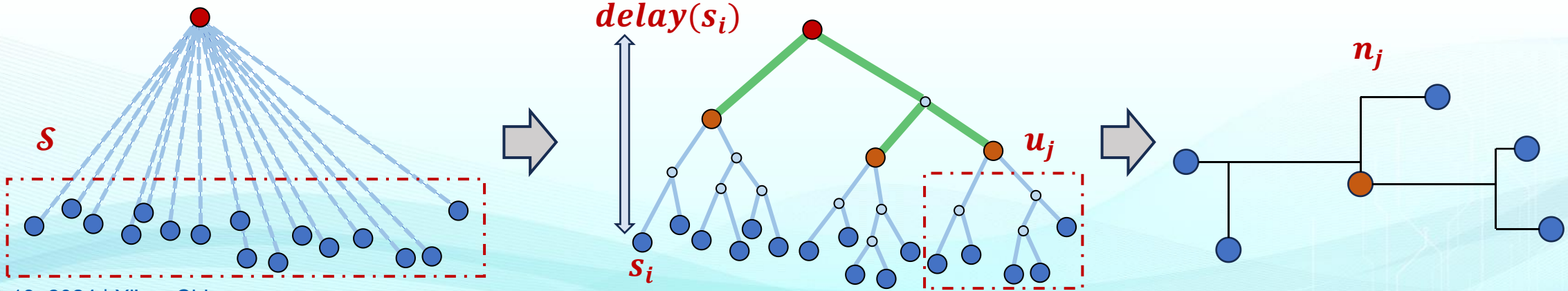
- **We define** the initial distribution of **flip-flops (sinks)** in the CTS stage as $\mathcal{S}$.

Given a clock source (root), we can establish **clock skew** constraint as follow:

$$\Delta delay(s_i) = \max_{s_i \in \mathcal{S}} \{delay(s_i)\} - \min_{s_i \in \mathcal{S}} \{delay(s_i)\} \leq skew\_bound,$$

where $delay(s_i)$ represents the delay from clock source to sink s_i (latency).

We define the set of load pins for each clock net as a **cluster**, i.e., $u_j \in \mathcal{U}$, and generate a clock net, i.e., $n_j \in \mathcal{N}$.



CTS Formula

- CTS needs to satisfy additional design constraints, such as the **fanout** constraint:

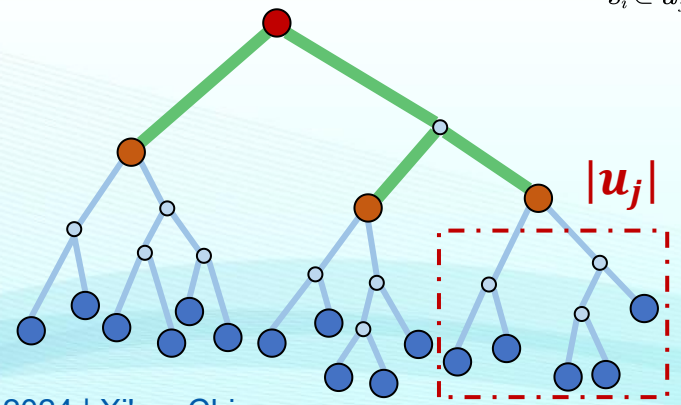
$$|u_j| \leq max\_fanout,$$

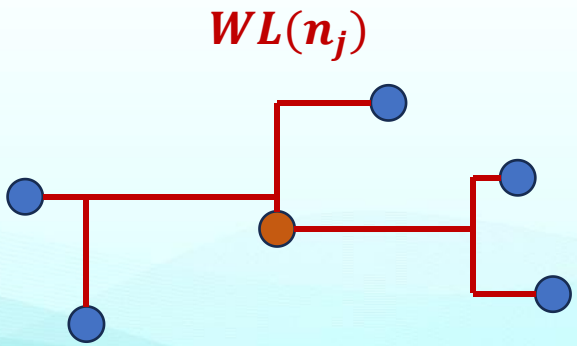
the maximum **wirelength** constraint:

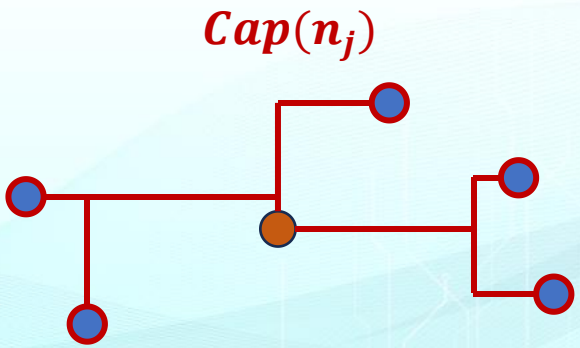
$$WL(n_j) \leq max\_length,$$

and the maximum clock net **capacitance** constraint:

$$\sum_{s_i \in u_j} Cap_{pin}(s_i) + c \cdot WL(n_j) \leq max\_cap.$$







CTS Formula

- CTS problem can be defined as a multi-objective optimization problem:

$$
\begin{aligned}
\min \quad & f(\mathcal{N},\mathcal{U},\mathcal{S}) \\
s.t. \quad & \Delta delay(s_i) \le skew\_bound, \\
& |u_j| \le max\_fanout, \\
& WL(n_j) \le max\_length, \\
& \sum_{s_i \in u_j} Cap_{pin}(s_i) + c \cdot WL(n_j) \le max\_cap,
\end{aligned}
$$

$f(\mathcal{N},\mathcal{U},\mathcal{S})$ represents the objective function, such as **design resources** and **power**.

It's a complex multi-objective optimization problem

Load Capacitance

- Equivalent Capacitance (Linear)

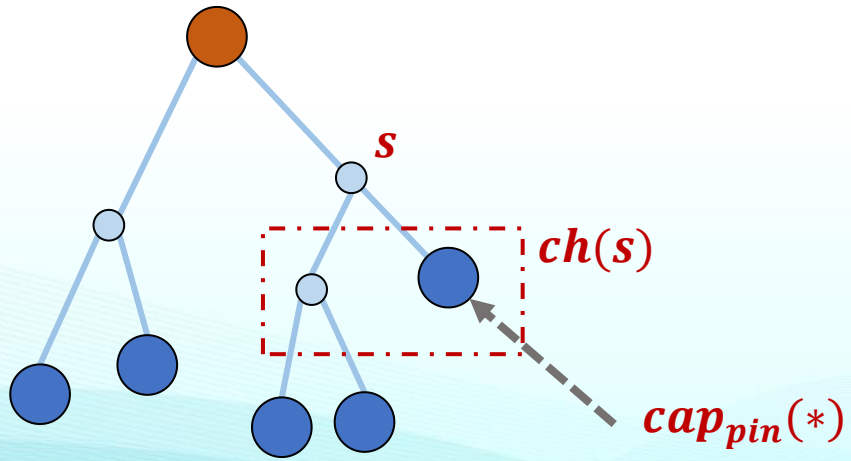
Capacitance per unit length/area

$$cap_{load}(s) = cap_{pin}(s) + \sum_{t \in ch(s)} (Cap_{load}(t) + c \cdot L(s,t)),$$

Pin Cap

Tree (Recursive)

Wire Cap



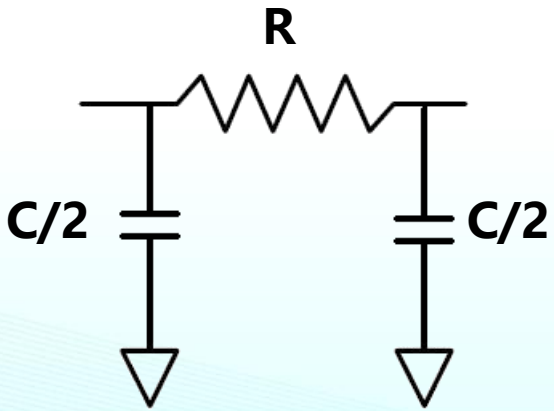
Bottom-up Calculation

Wire (Interconnect) Delay

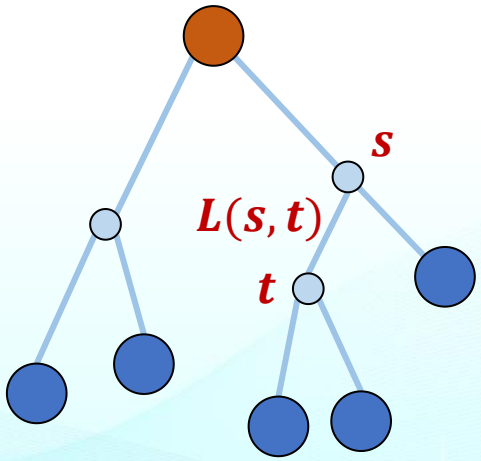
- Elmore π-Model

$$D_{wire}(s,t) = \underbrace{r \cdot L(s,t)}_{\text{Res}} \cdot \underbrace{\left[\frac{1}{2} \cdot c \cdot L(s,t) + Cap_{load}(t)\right]}_{\text{Cap}},$$

Resistance per unit length/area



Equivalent π-model of a wire segment

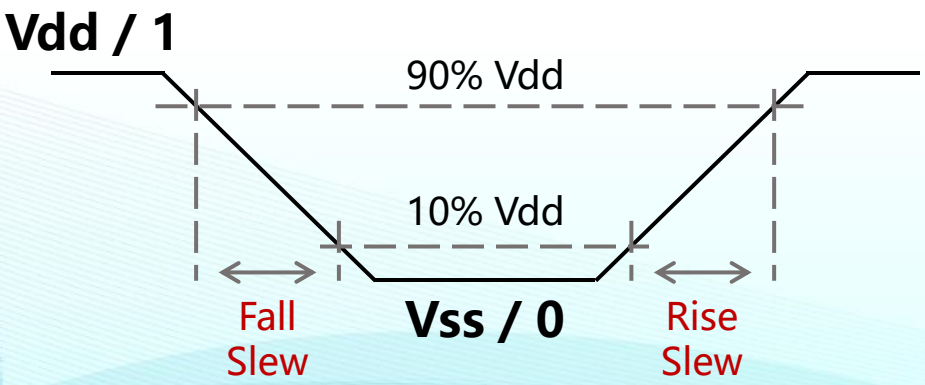


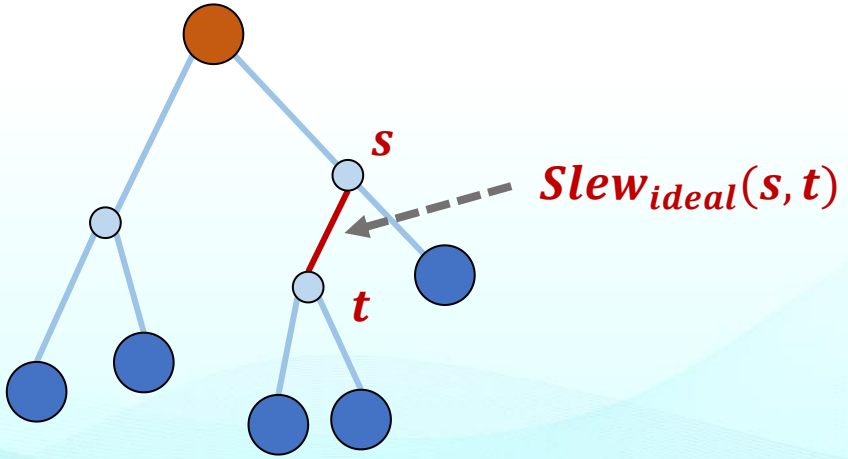
Slew (Transition)

- Bakoglu Metric[1] & PERI Model[2]

$$Slew_{ideal}(s,t) = \ln 9 \cdot D_{wire}(s,t),$$

$$Slew_{wire}(t) = \sqrt{Slew_{wire}^2(s) + Slew_{ideal}^2(s,t)}.$$



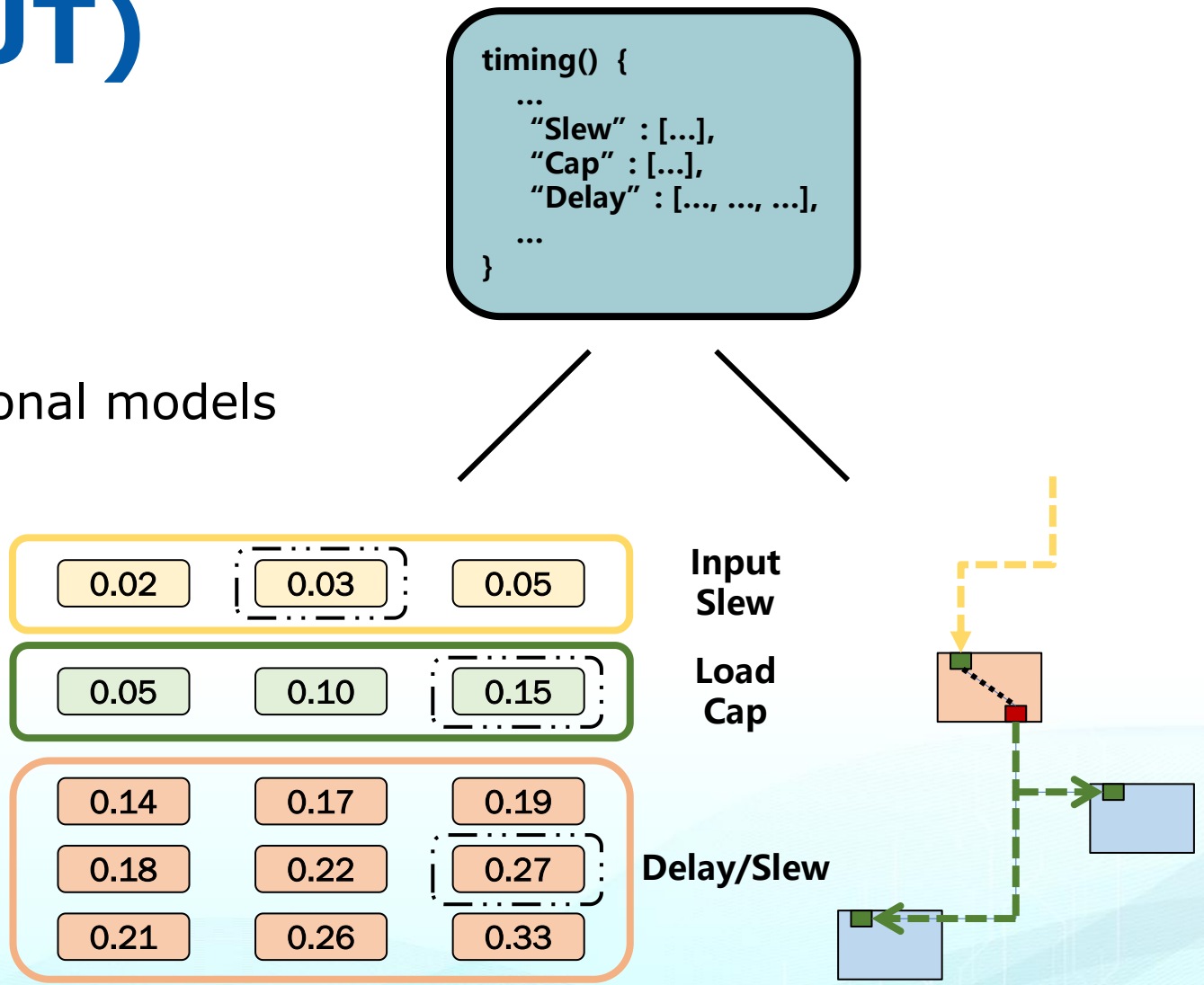


Top-down Calculation

Look-up Table (LUT)

- $Delay/Slew_{out} = LUT(Slew_{in}, Cap_{load})$
  - upstream input slew
  - downstream load capacitance
  - not reliant on complex computational models

- Model
  - bilinear interpolation
  - ML fitting…



Buffer Delay

- Linear Fitting[3]

$$D_{buf}(*) = LUT_{delay}(Slew_{in}(*), Cap_{load}(*)),$$

$$D_{buf}(*) = \alpha \cdot Slew_{in}(*) + \beta \cdot Cap_{load}(*) + \gamma.$$

Intrinsic Delay

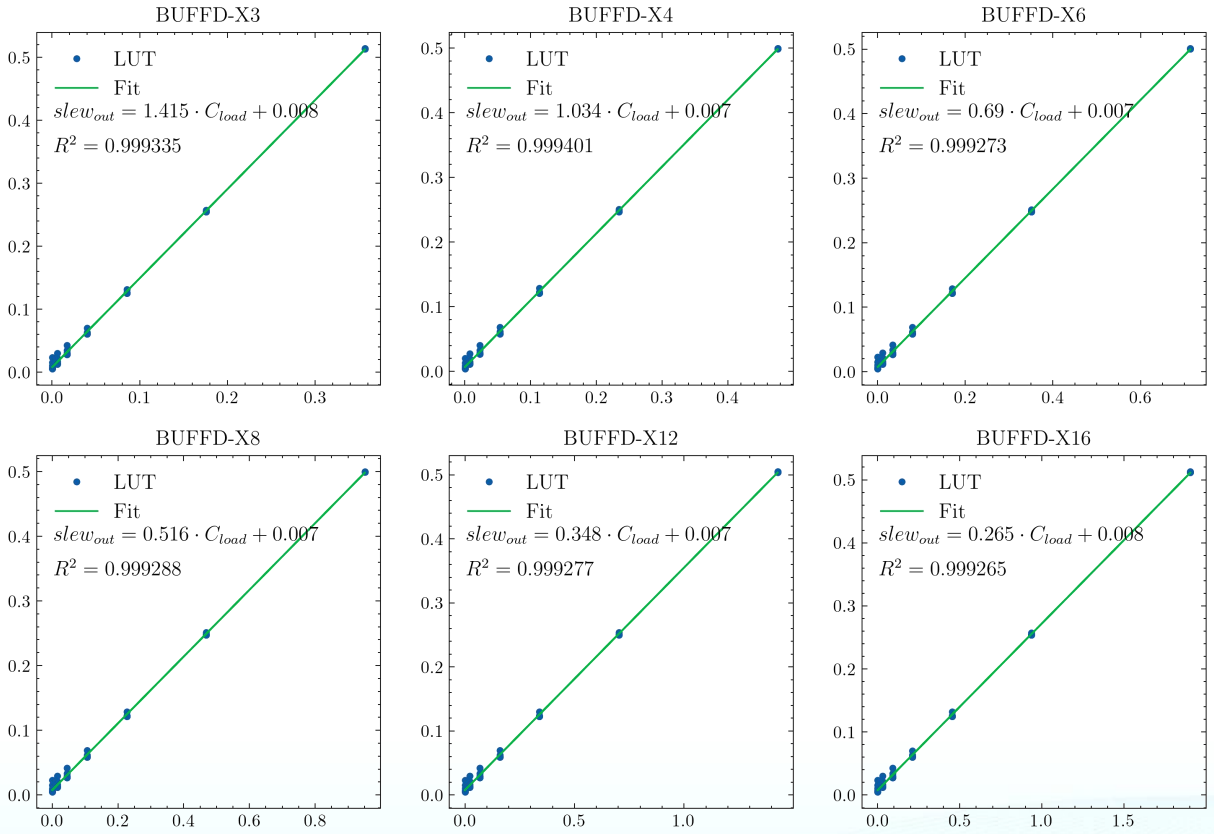
This type of modeling is only effective for clock buffers

Buffer Slew

- Linear Fitting[3]

$$Slew_{out}(*) = LUT_{slew}(Slew_{in}(*), Cap_{load}(*)),$$

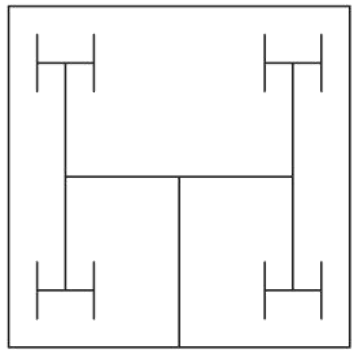
$$Slew_{out}(*) = \alpha \cdot Cap_{load}(*) + \beta.$$



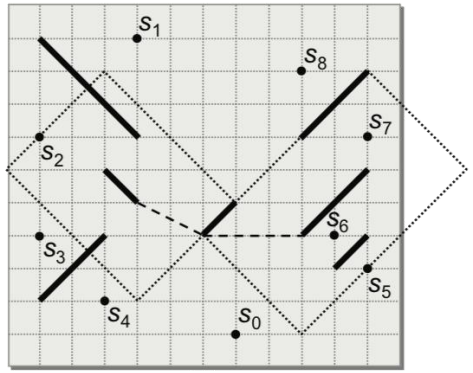
Fitting results under the 28nm process library

- Benefit
  - sufficient accuracy
  - independence from the upstream information (under certain conditions)

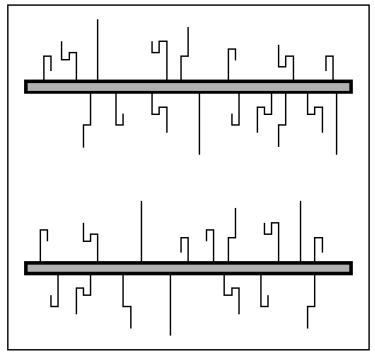
Clock Topology



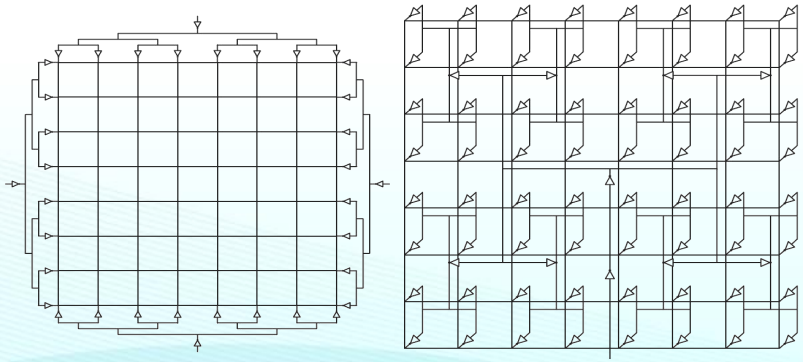
H-Tree[4]



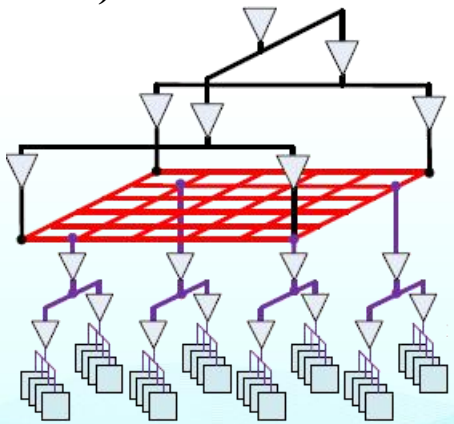
Deferred-Merge Embedding[5] (DME)

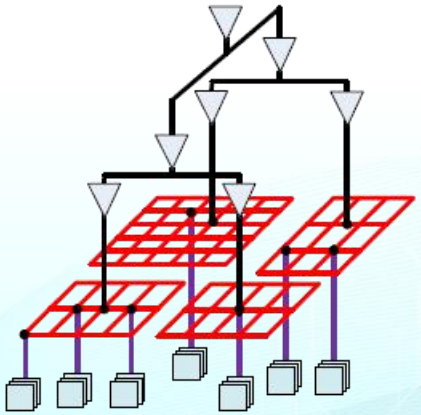


Fishbone/Spine[6]



Mesh Grid[7]

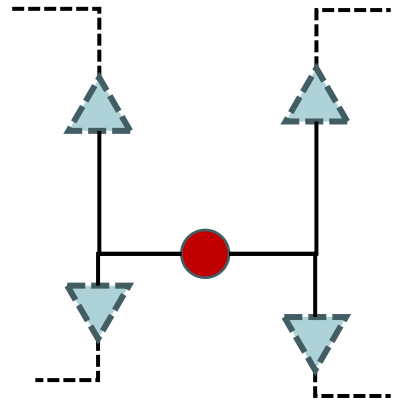




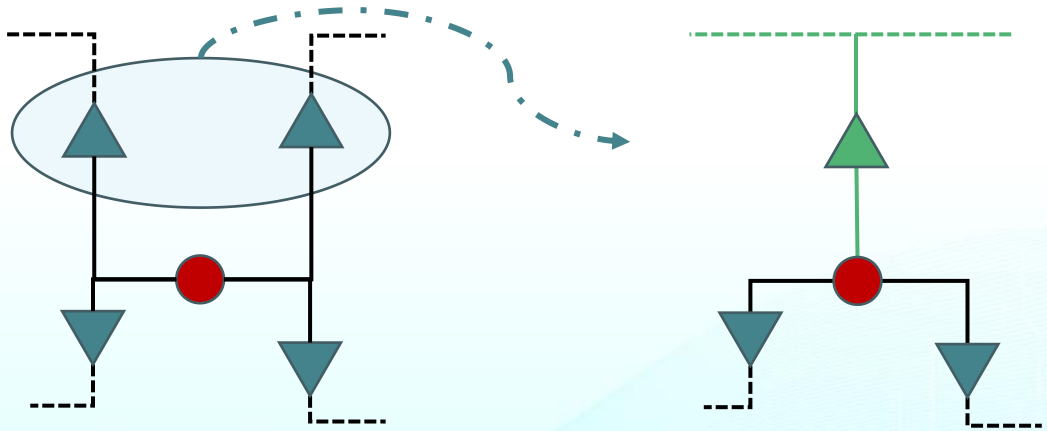
Hybrid[8]

Buffering

- Effect
  - reduce fanout
  - affect path delay
  - enhance driving capability (transition)
- Property
  - size/area
  - power/timing
  - library (for LUT)
- Optimization
  - insertion
  - resizing



Buffering (Insertion)

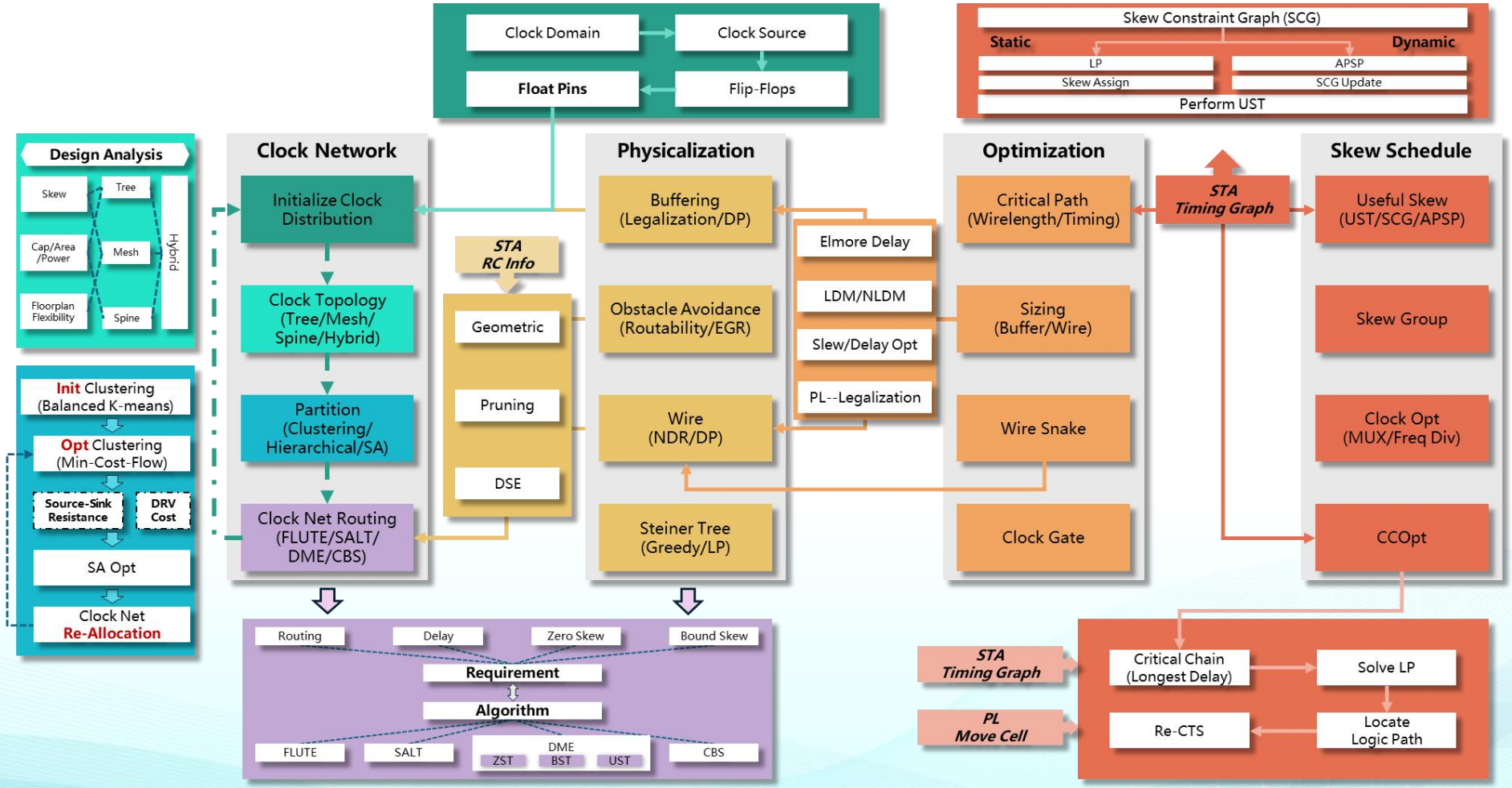


Timing (Buffering) Optimization

Overview

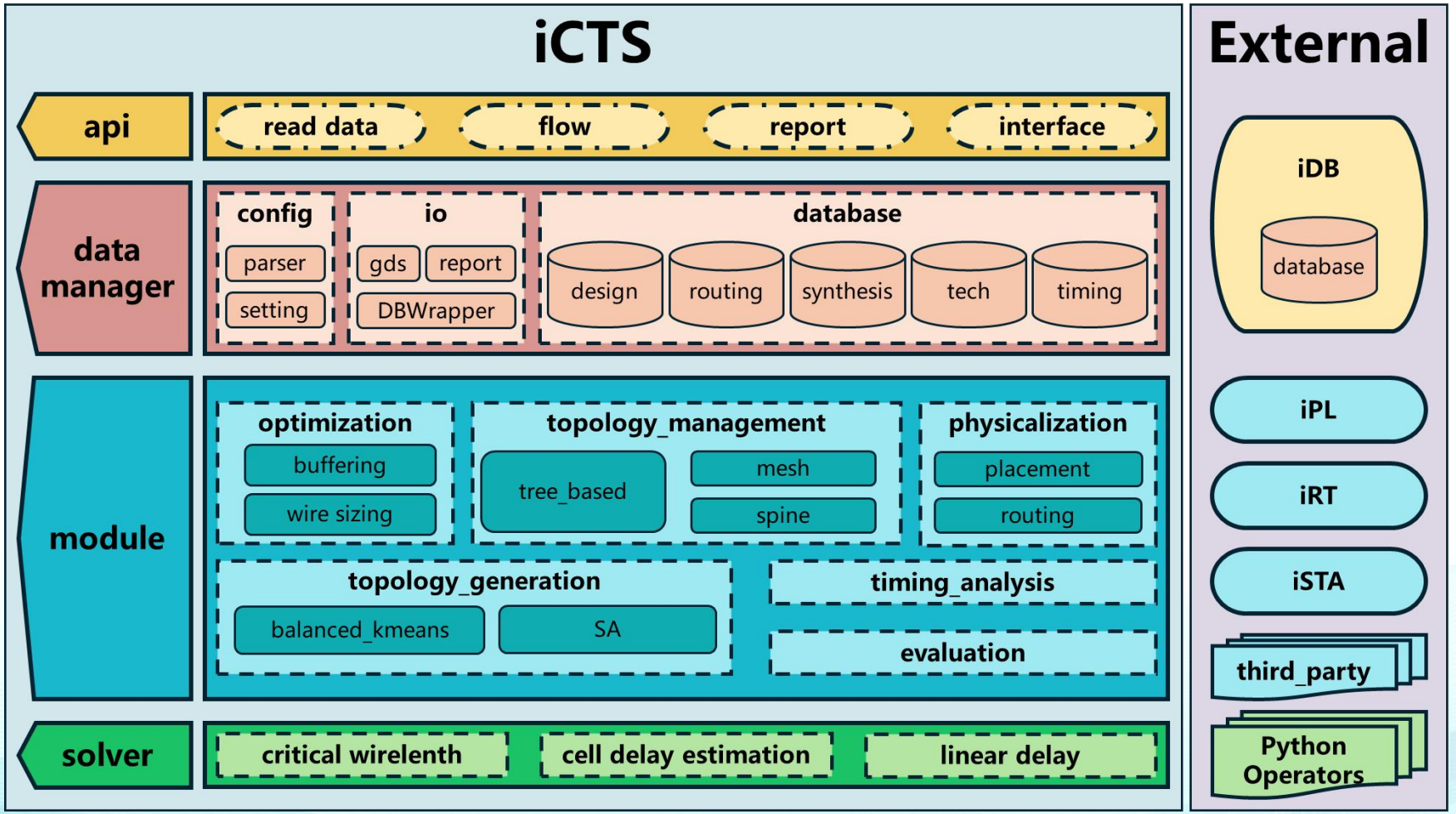
CTS Components

- Clock Network
- Physicalization
- Optimization
- Skew Schedule



iCTS Software

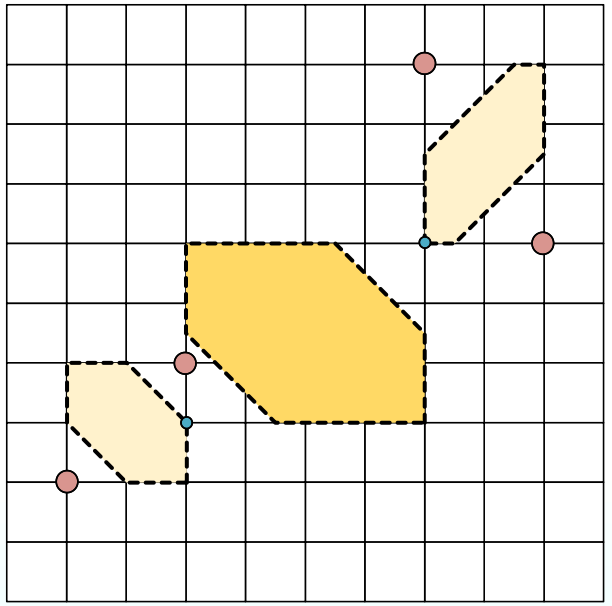
- api
- data manager
- module
- solver
- external lib



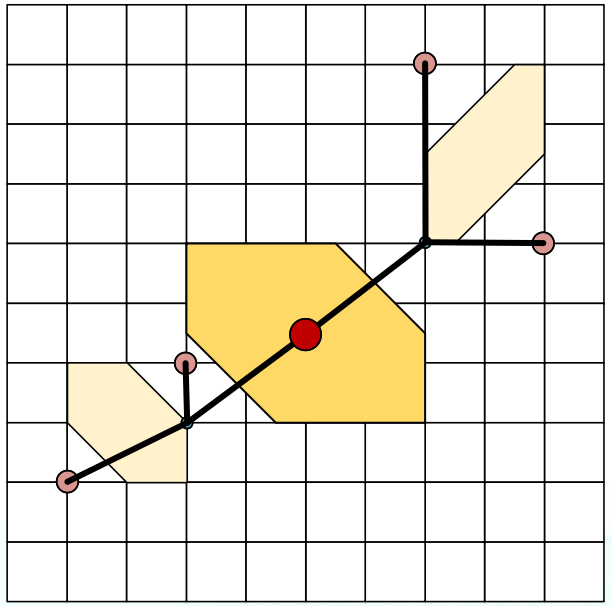
Overview

Bound Skew Tree (BST/DME)[9]

- Bottom-up Stage
  - merge two sub-tree
  - determine merge region
- Top-down Stage
  - location embedding
  - nearest endpoint
- Benefit
  - zero/bound skew
  - topology based



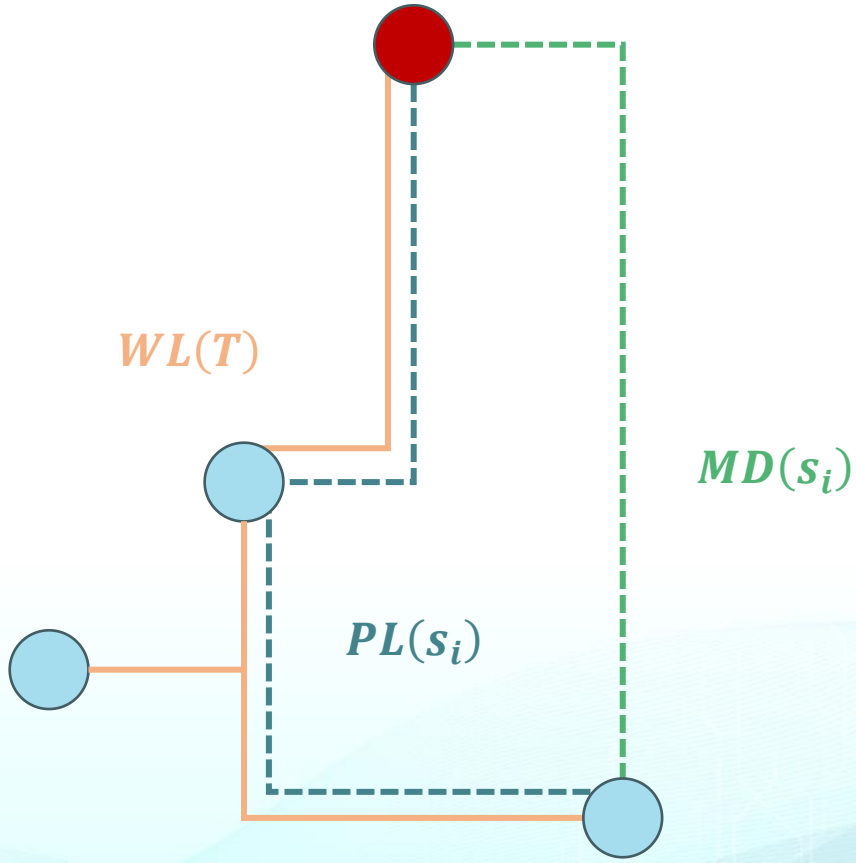
Bottom-up



Top-down

Fundamentals of Steiner Trees

- $PL(s_i)$, the **path length** from the source
- $MD(s_i)$, the **Manhattan distance** from the source
- $WL(T)$, the **total wirelength** of clock tree T
- $WL(T_{FLUTE})$, the **wirelength of FLUTE[10]** tree



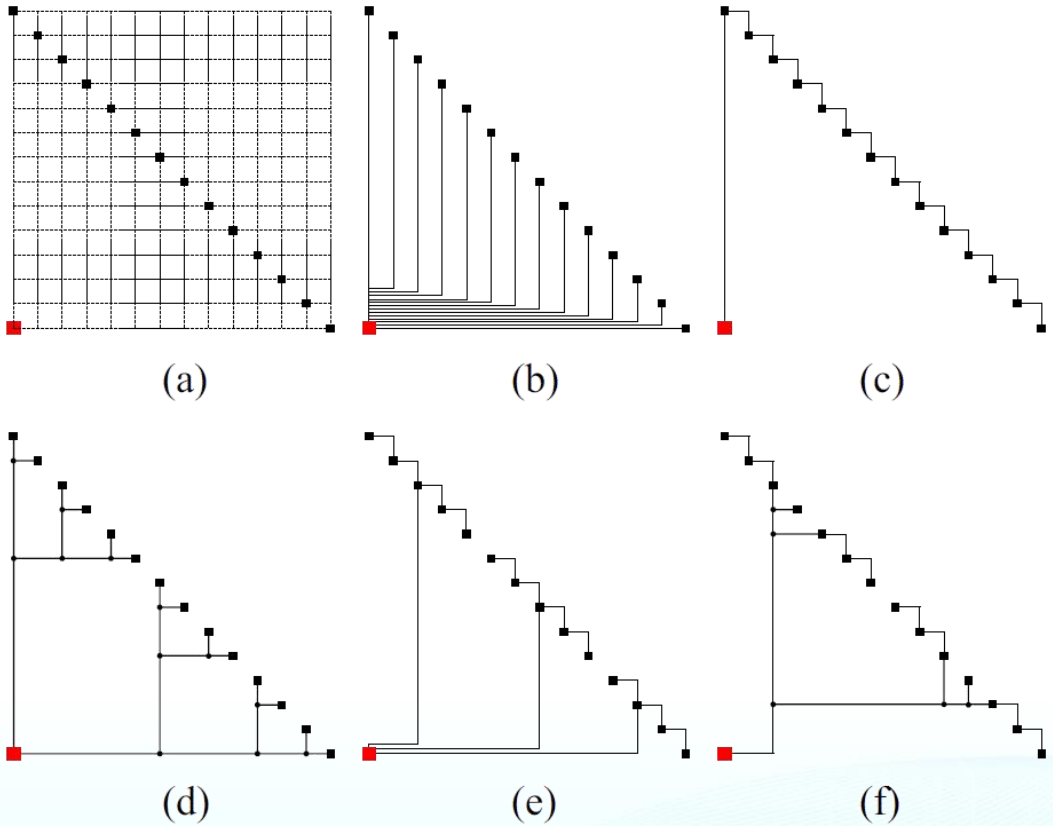
SALT[11]

- Shallowness (path length)

$$\alpha = \max_{s_i \in S} \left\{ \frac{PL(s_i)}{MD(s_i)} \right\},$$

- Lightness (tree weight)

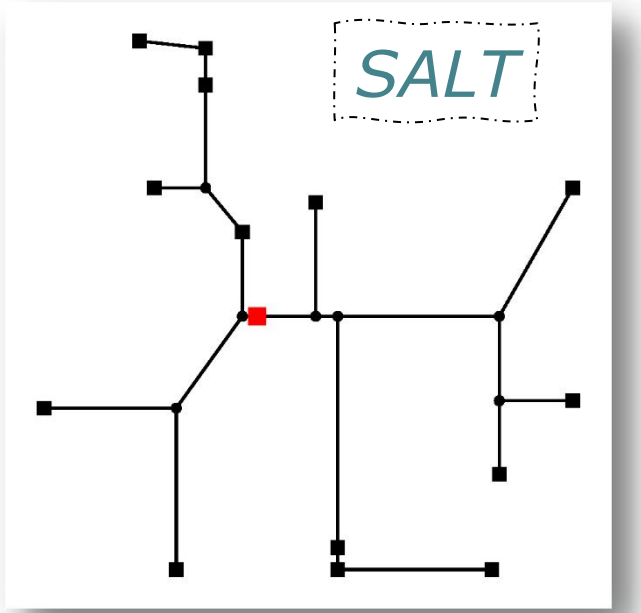
$$\beta = \frac{WL(T)}{WL(MST(G))} \approx \frac{WL(T)}{WL(T_{FLUTE})}.$$

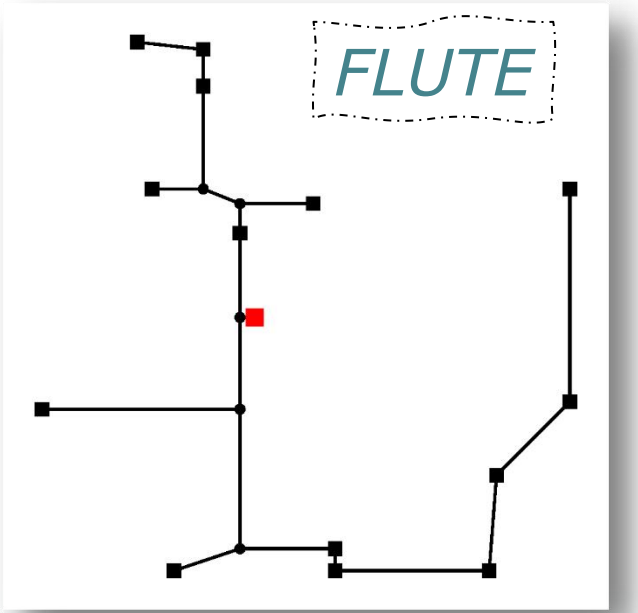


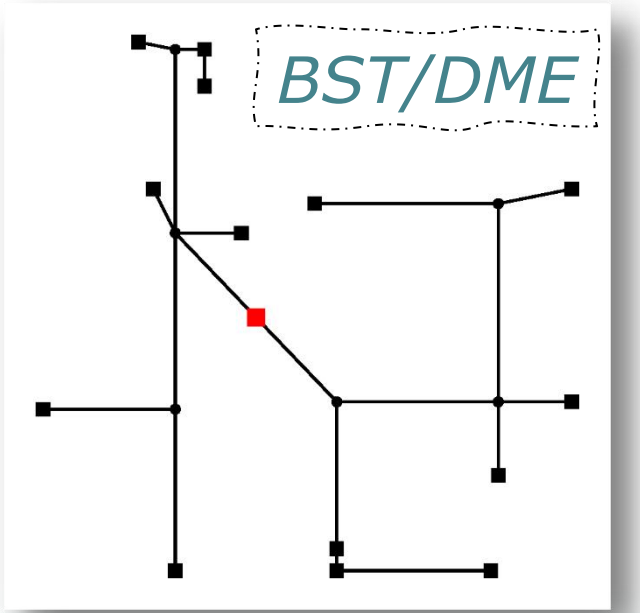
Different Shallow-Light Tree on the same net

Given ϵ, SALT realizes $\alpha \leq 1 + \epsilon$ and $\beta \leq 2 + \left\lceil log \frac{2}{\epsilon} \right\rceil$

Metric Mapping







$$\frac{PL_i}{MD_i} \leq 1$$

Shallowness

Latency (Delay)

$$min\{\sum WL\}$$

Lightness

Load Capacitance

$$skew \leq skew_{bound}$$

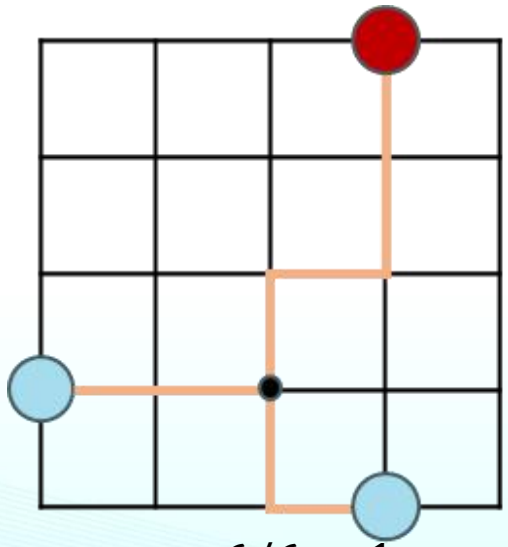
?

Skew

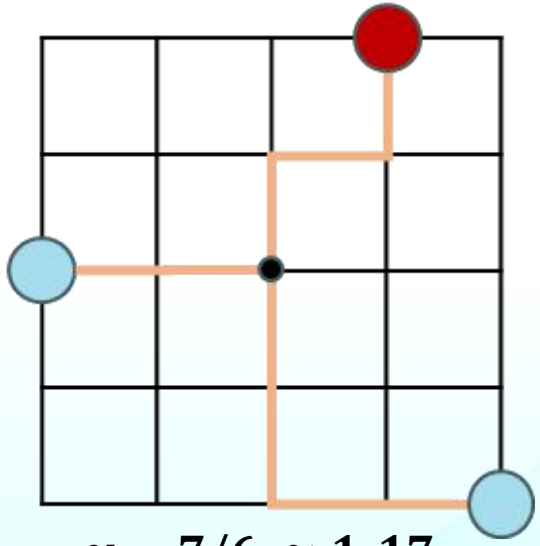
Skewness*

- The skewness of the Steiner tree is defined as:

$$\gamma = \frac{\max\limits_{s_i \in \mathcal{S}} \{PL(s_i)\}}{\overline{PL}}.$$



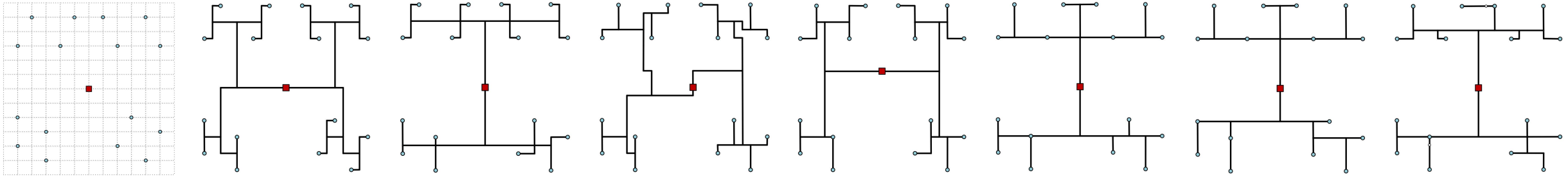
$\gamma = 6/6 = 1$
(Zero Skew Tree)



$\gamma = 7/6 \approx 1.17$
(Bound Skew Tree)

Skew-Latency-Load Tree (SLLT)*

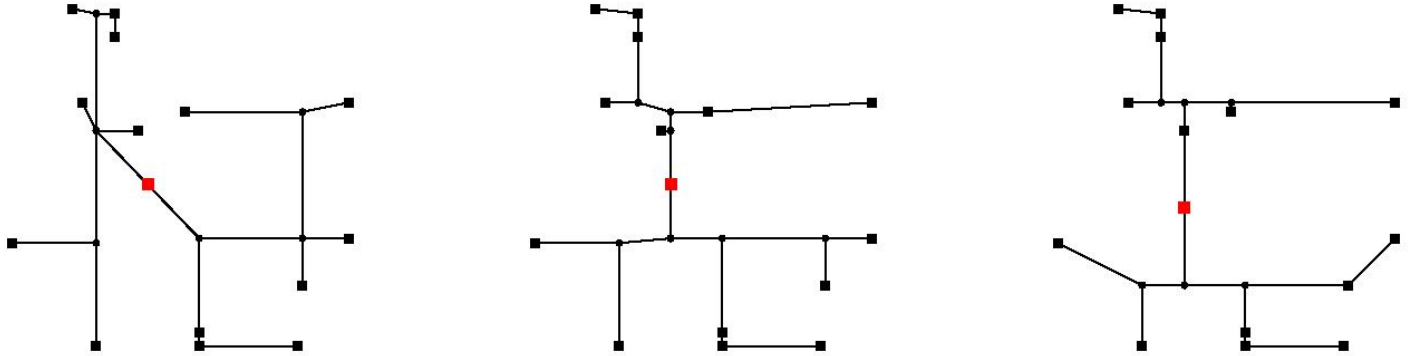
- A rectilinear Steiner tree with $\alpha \leq \overline{\alpha}$, $\beta \leq \overline{\beta}$, and $\gamma \leq \overline{\gamma}$, is denoted as $(\overline{\alpha}, \overline{\beta}, \overline{\gamma}) - SLLT$.



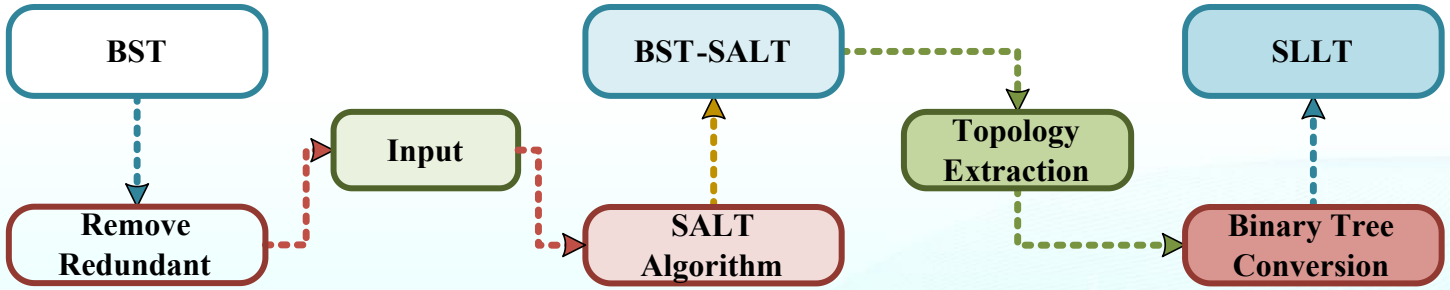
Algorithm	Max PL	Min PL	Total WL	Mean PL	α	β	γ	Mean	Skew Control
H-tree	10	9	55.5	9.75	2	1.32	1.03	1.45	√
GH-tree	10	7	47.5	8.5	1.6	1.13	1.18	1.3	√
ZST	10.5	10.5	55.5	10.5	2.33	1.32	**1.00**	1.55	√
BST	10	8	50	9.25	2.25	1.19	1.08	1.51	√
FLUTE	9	5	42	7.44	1.4	**1.00**	1.21	1.2	×
R-SALT	9	5	43	7.06	**1.00**	1.02	1.27	**1.10**	×
CBS*	9	7	45	8.13	1.4	1.07	1.11	**1.19**	√

Concurrent BST and SALT (CBS)*

- Bound Skew Tree (BST)
  - Benefit
    - Provides bounded skew control
  - Weakness
    - High cost of net (wirelength, cap)



- Steiner SLT (SALT)
  - Benefit
    - Acceptable total wirelength and smaller path length (delay)
  - Weakness
    - Lacks the ability to balance skew



Comparison

Table 1: Wirelength (um) comparison between R-SALT and CBS.

	GreedyDist			GreedyMerge			BiPartition		
Skew (ps)	80	10	5	80	10	5	80	10	5
R-SALT	314.4	314.3	315.1	312.6	313.0	315.6	312.2	312.4	312.7
CBS	306.0	307.1	316.1	305.2	306.3	314.3	305.3	305.6	312.7
Reduce	**2.67%**	**2.29%**	**-0.32%**	**2.37%**	**2.14%**	**0.41%**	**2.21%**	**2.18%**	**0.00%**

Table 2: Comparison on wirelength, cap and delay between BST-DME and CBS.

	Wirelength (um)			Cap (fF)			Wire Delay (ps)		
Skew (ps)	80	10	5	80	10	5	80	10	5
BST-DME	363.3	367.6	373.2	77.4	78.1	79.1	15.3	11.5	10.2
CBS	305.6	306.1	314	67.5	67.6	68.9	11.2	9.2	7.4
Reduce	**15.90%**	**16.70%**	**15.90%**	**12.80%**	**13.50%**	**12.80%**	**26.60%**	**20.50%**	**26.80%**

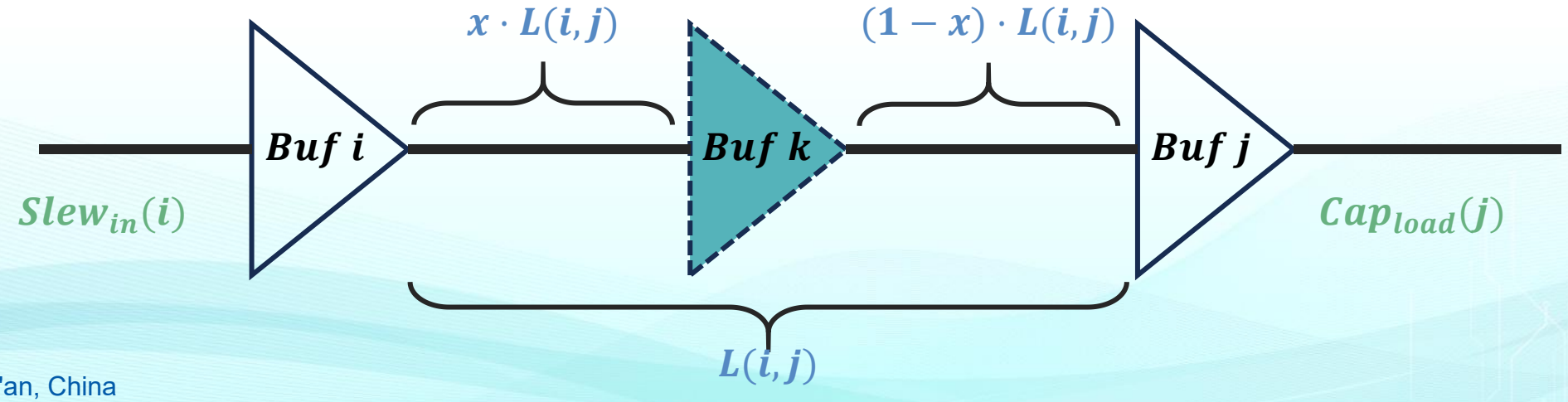
Overview

Critical Wirelength Model*

- Special Net: Buffering between Buffers.

$$T(i,j) = D_{buf}(i) + D_{wire}(i,j) + D_{buf}(j),$$

$$T'(i,j) = D'_{buf}(i) + D'_{wire}(i,k) + D'_{buf}(k) + D'_{wire}(k,j) + D'_{buf}(j).$$



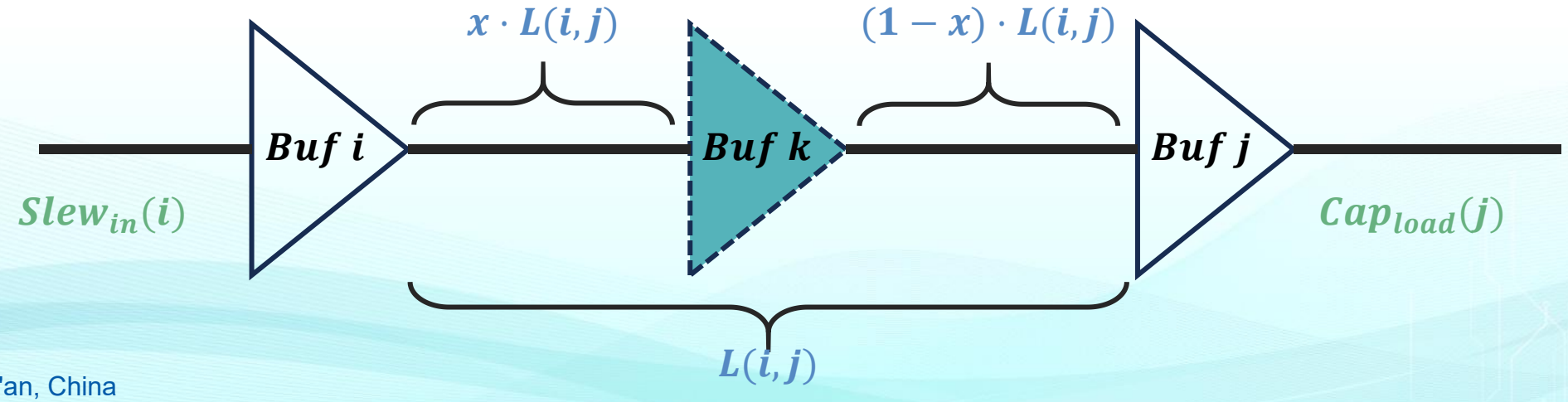
Critical Wirelength Model*

- Special Net: Buffering between Buffers.

$$T(i,j) - T'(i,j) = x \cdot (1-x) \cdot r \cdot c \cdot (\ln 9 \cdot \alpha + 1) \cdot L(i,j)^2 - \beta \cdot Cap_{pin} - \gamma,$$

$$CL_{buf}(i,j) = \sqrt{\frac{\beta \cdot Cap_{pin} + \gamma}{x \cdot (1-x) \cdot r \cdot c \cdot (\ln 9 \cdot \alpha + 1)}}$$

$$\geq 2 \cdot \sqrt{\frac{\beta \cdot Cap_{pin} + \gamma}{r \cdot c \cdot (\ln 9 \cdot \alpha + 1)}}.$$

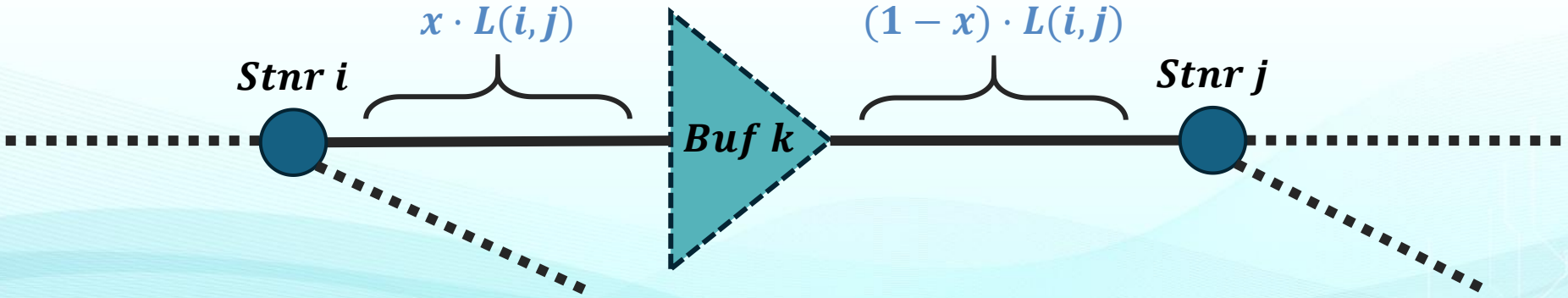


Critical Wirelength Model*

- General Net: Buffering between Steiner Points.

$$T(i,j) = D_{wire}(i,j),$$

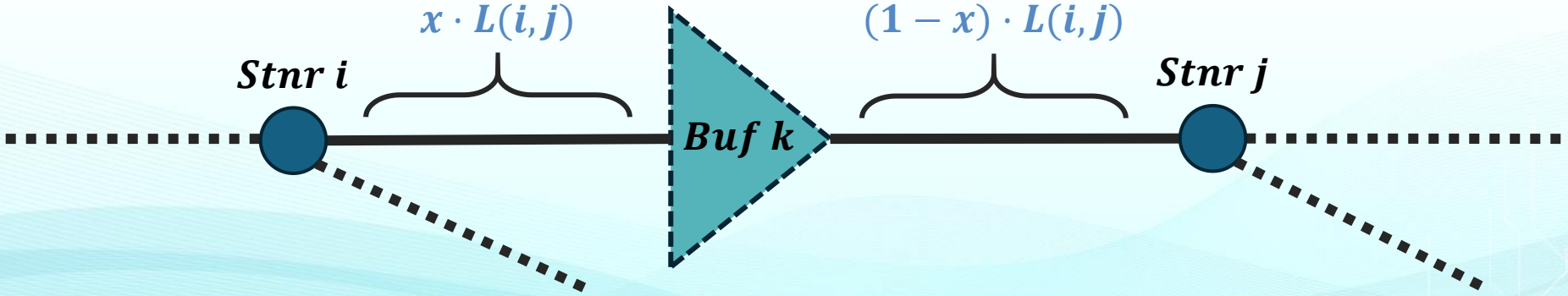
$$T'(i,j) = D'_{wire}(i,k) + D'_{buf}(k) + D_{wire}(k,j).$$



Critical Wirelength Model*

- General Net: Buffering between Steiner Points.

$$\Delta T(i,j) = \frac{1}{2} \cdot r \cdot c \cdot [(2 + \alpha \cdot \ln 9) \cdot x^2 - 2 \cdot x] \cdot L(i,j)^2$$
$$+ \{r \cdot x \cdot [(1 + \alpha \cdot \ln 9) \cdot Cap_{pin} - Cap^*_{load}(j)]$$
$$+ \beta \cdot c \cdot (1 - x)\} \cdot L(i,j)$$
$$+ \beta \cdot Cap^*_{load}(j) + \gamma$$
$$- \beta \cdot [c \cdot (1 - x) \cdot L(i,j) + Cap^*_{load}(j)] \le 0,$$



Critical Wirelength Model*

- General Net: Buffering between Steiner Points.

$$\widehat{CL}_{stnr}(i,j) = 2\sqrt{\frac{\beta \cdot \frac{cap_{max}}{2} + \gamma}{r \cdot c \cdot (\ln 9 \cdot \alpha + 1)}},$$

Table 3: Critical wirelength statistics.

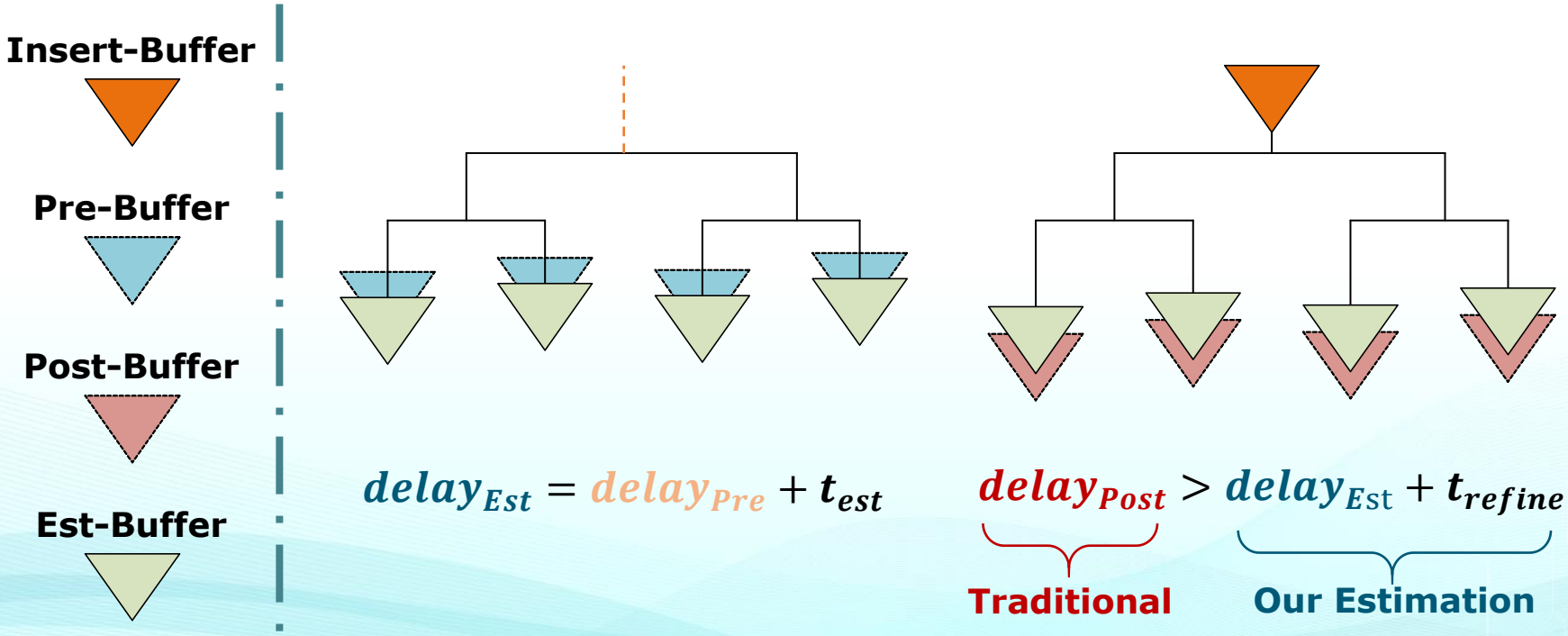
$Cell$	CL_{buf}	$\widehat{CL}_{stnr}$	CL_{stnr}	x
BUFFD-X3	157.043	211.842	351.185	0.27
BUFFD-X4	146.492	206.286	275.137	0.31
BUFFD-X6	174.212	242.255	279.531	0.35
BUFFD-X8	187.388	257.016	270.459	0.38
BUFFD-X12	195.638	269.881	254.932	0.41
BUFFD-X16	211.98	287.216	258.906	0.42

Insertion Delay Estimation Model*

- Estimate the contribution of downstream load capacitance to delay:

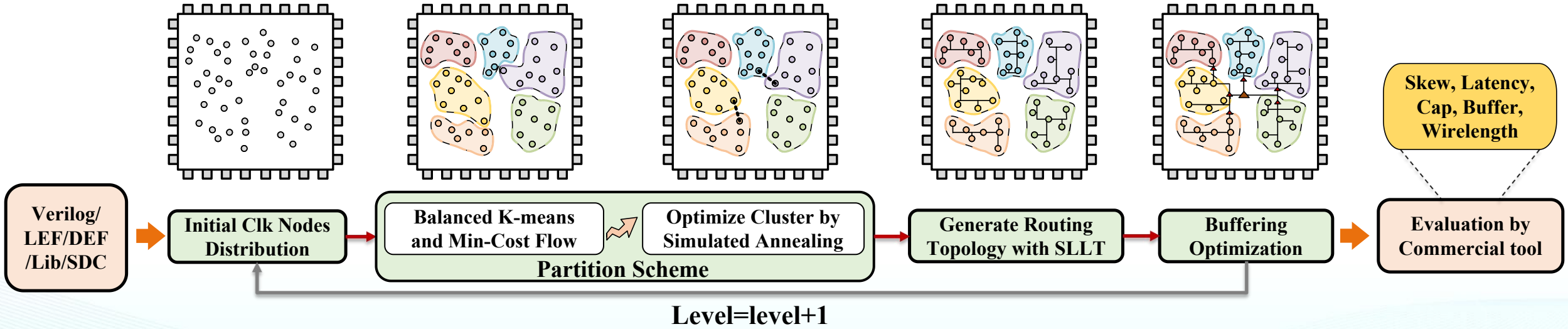
$$delay_{est} = LUT(Cap_{load}, Slew_{in}) - \boldsymbol{\omega_s \cdot Slew_{in}}$$

$$\approx \omega_c \cdot Cap_{load} + \omega_{inherent}$$



Overview

Hierarchical Framework*



Overview

Comparison of Open Source Test Cases

Table 4: Comparison between clock tree solutions from ours, commercial tool, and OpenROAD.

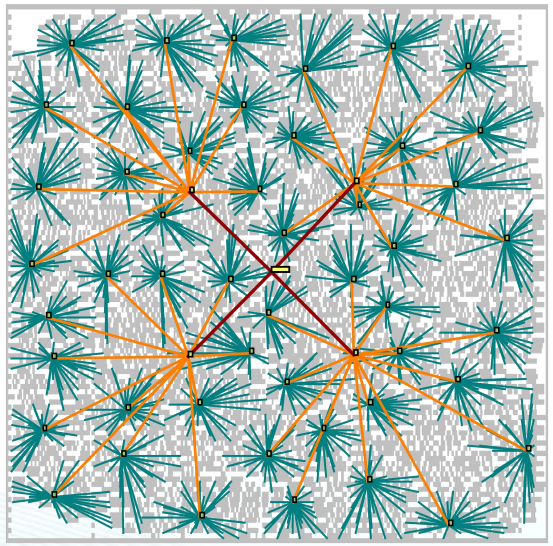
Case	Latency (ps)			Skew (ps)			#Buffers			Buf Area (μm^2)			Clk Cap (fF)			Clk WL (μm)		
	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.
s38584	**71**	76	93	13	**8**	17	**43**	**43**	45	**26.6**	26.8	39.7	**904**	1019	1087	3382.0	**3366.5**	3478.9
s38417	**75**	84	94	**10**	19	16	**53**	54	55	**32.4**	33.6	48.5	**1083**	1235	1284	**3755.6**	3867.4	3870.5
s35932	**80**	81	100	13	**10**	17	**58**	59	64	**35.5**	36.5	56.4	**1217**	1380	1433	**4380.0**	4420.9	4449.4
salsa20	**82**	87	112	**19**	21	29	**81**	83	109	**49.6**	51.8	96.1	**1715**	2050	2160	**6446.9**	6580.9	6863.5
ethernet	**97**	104	159	34	**31**	51	**337**	352	455	**315.5**	320.4	408.9	**7314**	8823	9210	26113.9	**26105.5**	27248.8
vga_lcd	**134**	146	206	**41**	49	**92**	**575**	597	775	**416.9**	451.8	812.1	**12380**	14920	15815	46763.1	**45969.8**	47484.1
Avg.	**1.000**	1.072	1.417	**1.000**	1.062	1.708	**1.000**	1.036	1.310	**1.000**	1.051	1.668	**1.000**	1.196	1.259	1.000	**0.994**	1.028

Comparison of ysyx designs

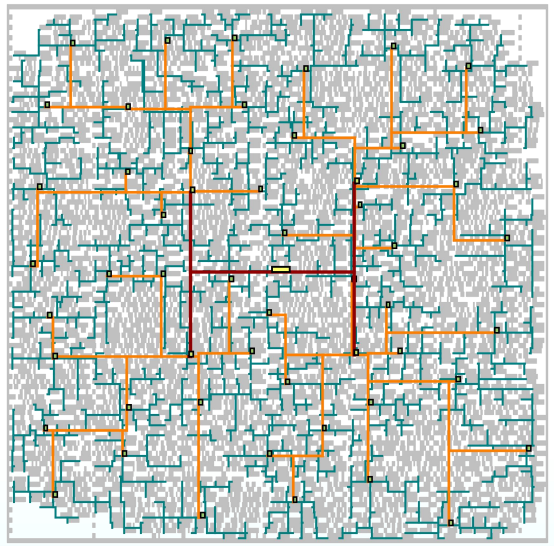
Table 5: Comparison of ysyx designs among clock tree solutions from ours, commercial tool, and OpenROAD.

Case	Latency (ps)			Skew (ps)			#Buffers			Buf Area (μm^2)			Clk Cap (fF)			Clk WL (μm)		
	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.	Ours	Com.	OR.
ysyx_0	**113**	120	156	31	**15**	63	**639**	654	799	**550.4**	561.5	1719.0	29032	31662	**17130**	**42698.49**	42935.09	45389.69
ysyx_1	**118**	122	206	29	**12**	**110**	**656**	674	863	568.4	**568.3**	1896.4	29455	32204	**17907**	**44538.49**	45142.38	48113.24
ysyx_2	**140**	143	191	38	**16**	68	**943**	958	1113	**801.2**	814.6	2401.7	35272	39256	**25491**	**64884.11**	64901.76	69753.65
ysyx_3	144	**139**	193	36	**16**	59	**798**	808	914	**671.8**	689.0	1970.4	32483	35830	**21484**	57229.57	**56918.98**	59314.99
Avg.	**1.000**	1.017	1.449	1.000	**0.44**	2.239	**1.000**	1.019	1.215	**1.000**	1.016	3.082	1.000	1.101	**0.65**	**1.000**	1.003	1.063

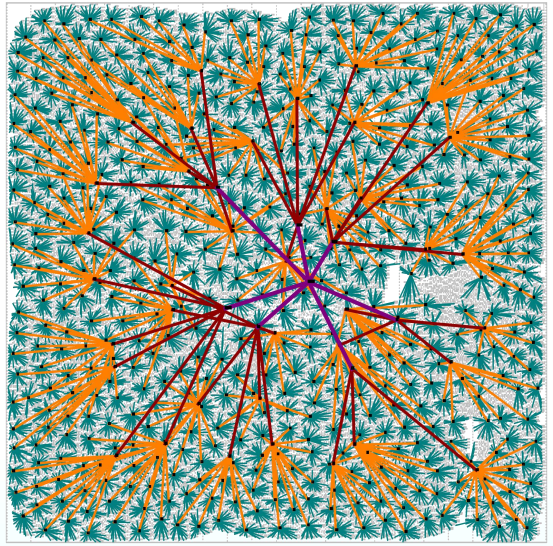
Topology & Routing



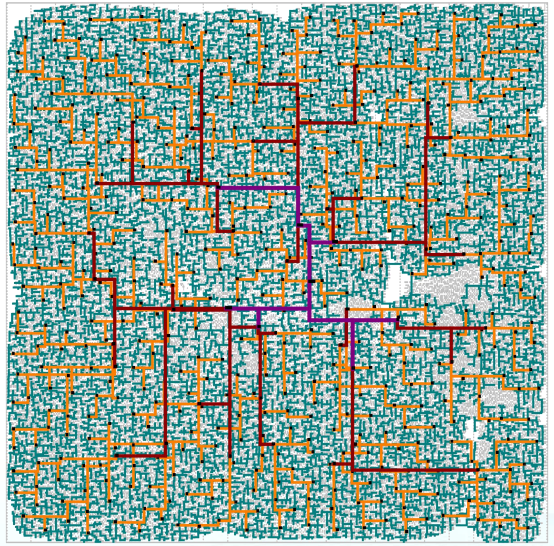
s38417 Topology



s38417 Routing



vga_enh_top Topology

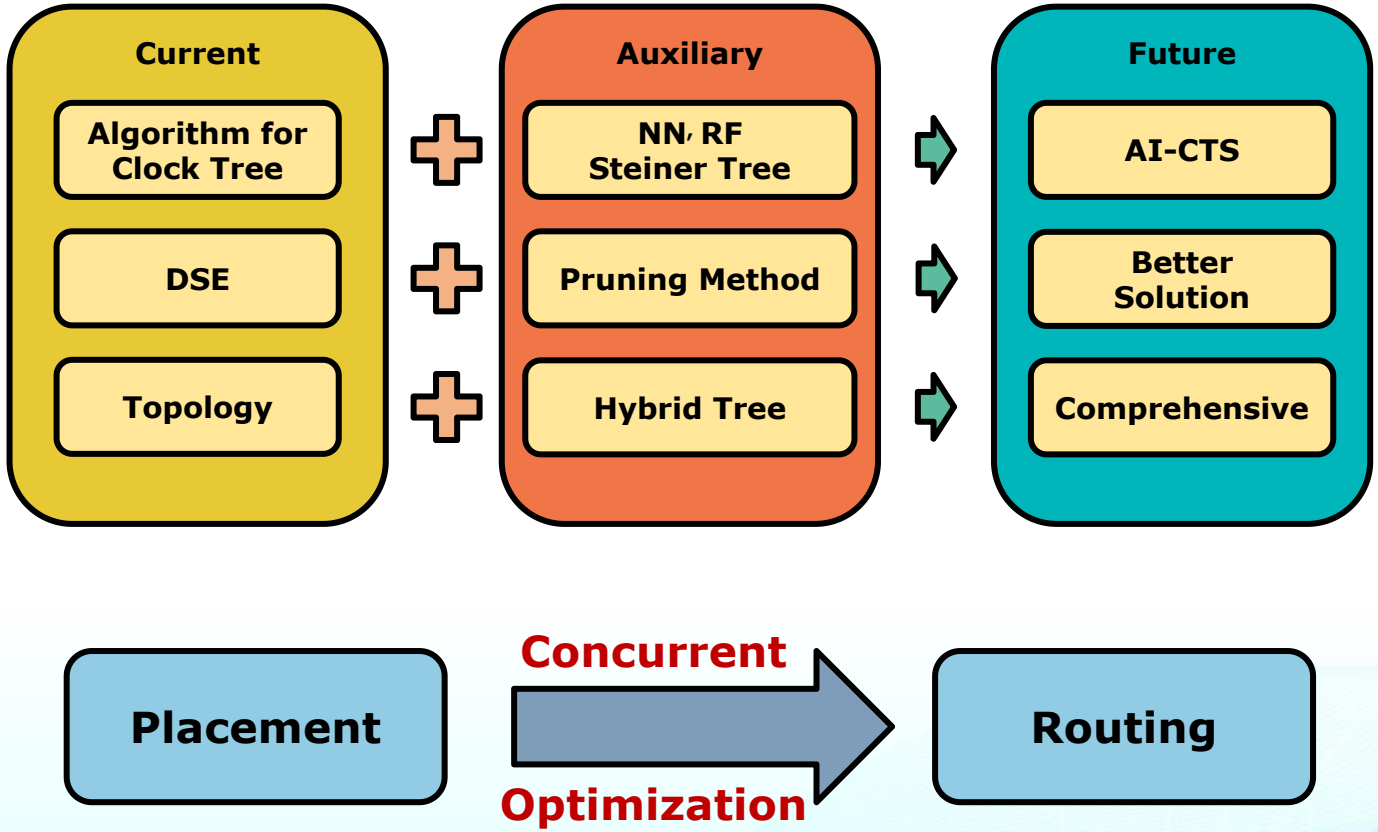


vga_enh_top Routing

Overview

Future

- Software
  - Clock Topology
  - Community
  - iCTS API
- Flow
  - Timing Representation
  - Clock Routing
  - Buffering Optimization
- Technology
  - DSE
  - CCOpt
  - AI for CTS



Reference

[1] Bakoglu H B. Circuits, interconnections, and packaging for VLSI[J]. (No Title), 1990.

[2] Kashyap C V, Alpert C J, Liu F, et al. Closed-form expressions for extending step delay and slew metrics to ramp inputs for RC trees[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2004, 23(4): 509-516.

[3] Sitik C, Lerner S, Taskin B. Timing characterization of clock buffers for clock tree synthesis[C]. 2014 IEEE 32nd International Conference on Computer Design (ICCD), 2014: 230-236.

[4] Bakoglu H B. Circuits, interconnections, and packaging for VLSI[J]. 1990.

[5] Boese K D, Kahng A B. Zero-skew clock routing trees with minimum wirelength[C]. [1992] Proceedings. Fifth Annual IEEE International ASIC Conference and Exhibit, 1992: 17-21.

[6] Andreev A, Nikishin A, Gribok S, et al. Clock network fishbone architecture for a structured ASIC manufactured on a 28 NM CMOS process lithographic node: Google Patents, 2014.

[7] Chakrabarti P, Bhatt V, Hill D, et al. Clock mesh framework[C]. Thirteenth International Symposium on Quality Electronic Design (ISQED), 2012: 424-431.

[8] Abdelhadi A, Ginosar R, Kolodny A, et al. Timing–driven variation–aware synthesis of hybrid mesh/tree clock distribution networks[J]. Integration, 2013, 46(4): 382-391.

[9] Cong J, Kahng A B, Koh C-K, et al. Bounded-skew clock and Steiner routing[J]. ACM Transactions on Design Automation of Electronic Systems (TODAES), 1998, 3(3): 341-388.

[10] Chu C, Wong Y-C. FLUTE: Fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2007, 27(1): 70-83.

[11] Chen G, Young E F. Salt: provably good routing topology by a novel steiner shallow-light tree algorithm[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019, 39(6): 1217-1230.

Thanks