UT DA

Layout Dependent Aging Mitigation for Critical Path Timing

Che-Lun Hsu[1], Shaofeng Guo[2], Yibo Lin[1], Xiaoqing Xu[1],
Meng Li[1], Runsheng Wang[2], Ru Huang[2], and David Z. Pan[1]

[1]ECE Department, University of Texas at Austin
[2]Institute of Microelectronics, Peking University

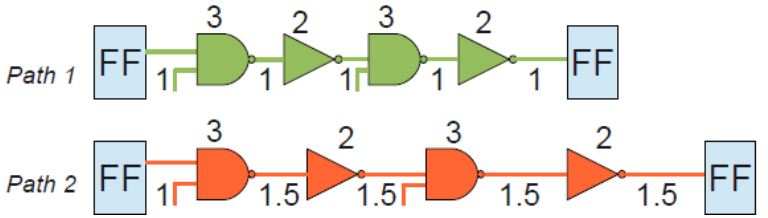
Motivations

- Aging effects are becoming increasingly important
  - Bias Temperature Instability (BTI): NBTI and PBTI
  - Hot Carrier Injection (HCI)
  - Increase gate delay by more than 10% in 28nm process [1]
- Degrade end-of-life (EOL) circuit performance significantly

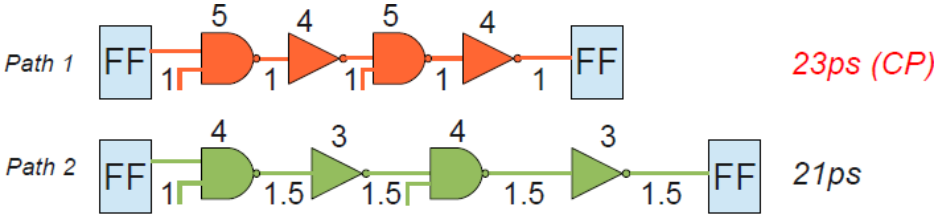
[1] P. Ren et al., IEDM 2015

Motivations

- Aging effects are observed to show strong layout dependency
  - Ignoring layout dependency leads to 11.5% inaccuracy [1]
  - Important factors considered in the industry flow for sub-20nm process
- Layout dependent aging (LDA) complicates EOL timing optimization
  - Timing degradation
  - Change of critical paths (CPs) in EOL compared to the early life design



Early life timing



EOL timing

[1] P. Ren et al., IEDM 2015

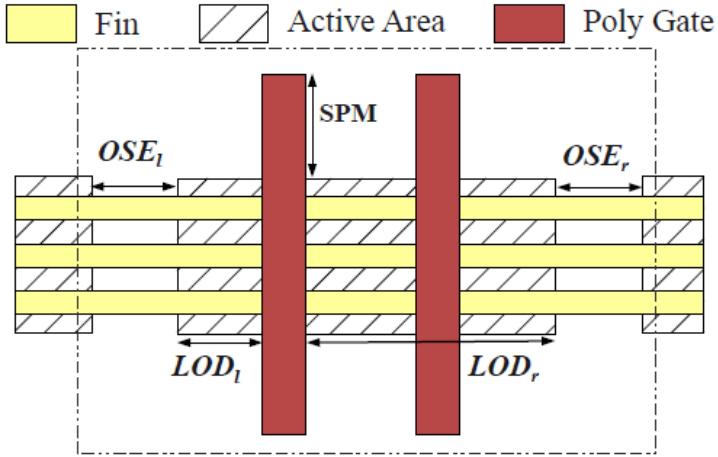
Motivations

- Aging effects are observed to show strong layout dependency
  - Ignoring layout dependency leads to 11.5% inaccuracy [1]
  - Important factors considered in the industry flow for sub-20nm process
- Layout dependent aging (LDA) complicates EOL timing optimization
  - Timing degradation
  - Change of critical paths (CPs) in EOL compared to the early life design
- Can we leverage the layout dependency to optimize the EOL timing?

[1] P. Ren et al., IEDM 2015

Important LDA Factors

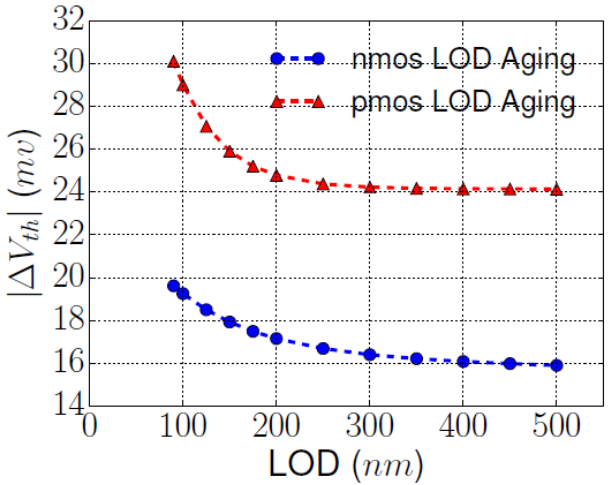
- Important LDA factors [1]:
  - LOD – Length between the gate and the edge of diffusion
  - OSE – Active to active spacing
  - SPM – Poly extension from active area
- SPM is usually fixed due to design rule check

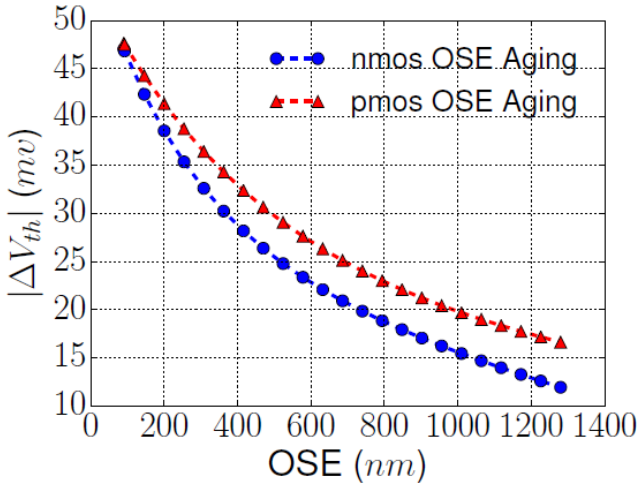


[1] P. Ren et al., IEDM 2015

Important LDA Factors

- Larger LOD and OSE lead to smaller threshold voltage degradation
  - ΔVth decreases 25% as LOD increases from 100nm to 500nm
  - ΔVth decreases 200% as OSE increases from 100nm to 1300nm
- LOD and OSE dependency enable optimization in placement stage
  - Both LOD and OSE are impacted by cell relative positions

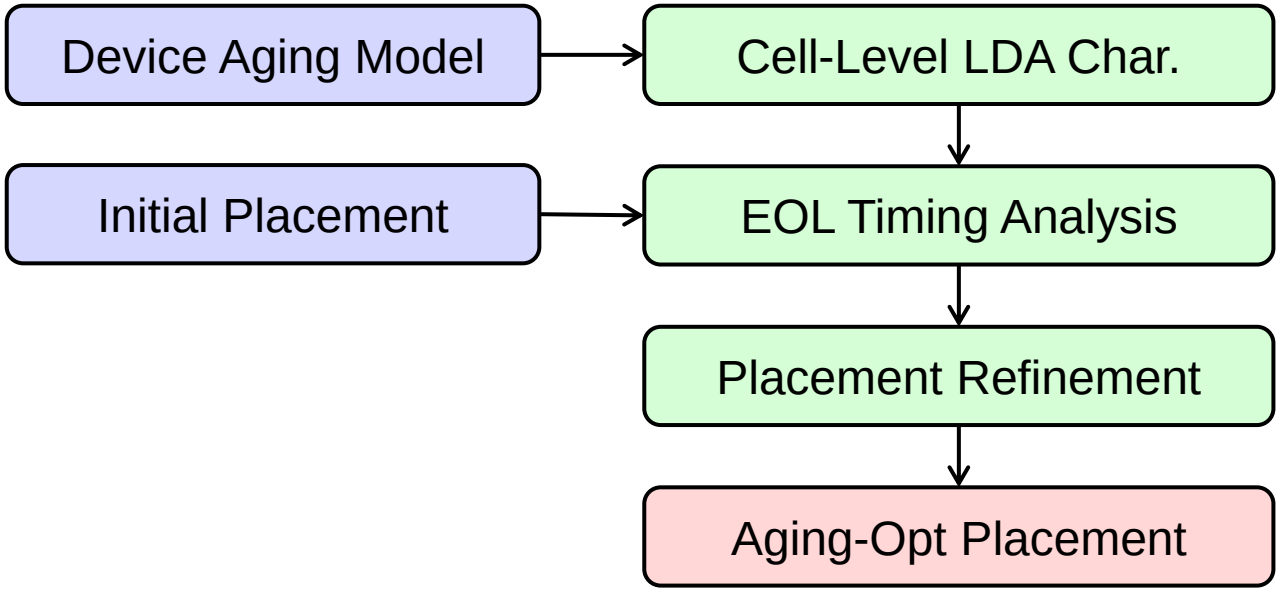




[1] P. Ren et al., IEDM 2015

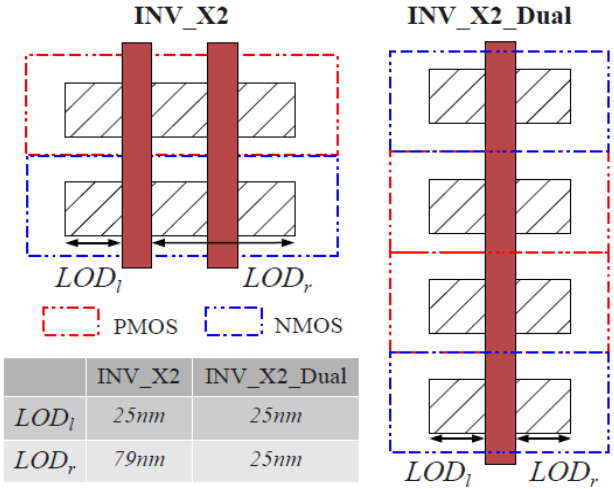
Aging Mitigation Framework

- Overall flow
  - *Improve EOL timing while maintaining early-life timing closure during the placement stage*



Cell-Level LDA Characterization

- Obtain cell-level aging impact through spice simulations
  - Leverage industrial strength LDA model [1] to build the EOL timing library
- Impact of cell layout configurations
  - Compare single-row and dual-row height cells with the same driving strength
  - Single-row height cells degrade less due to larger LOD



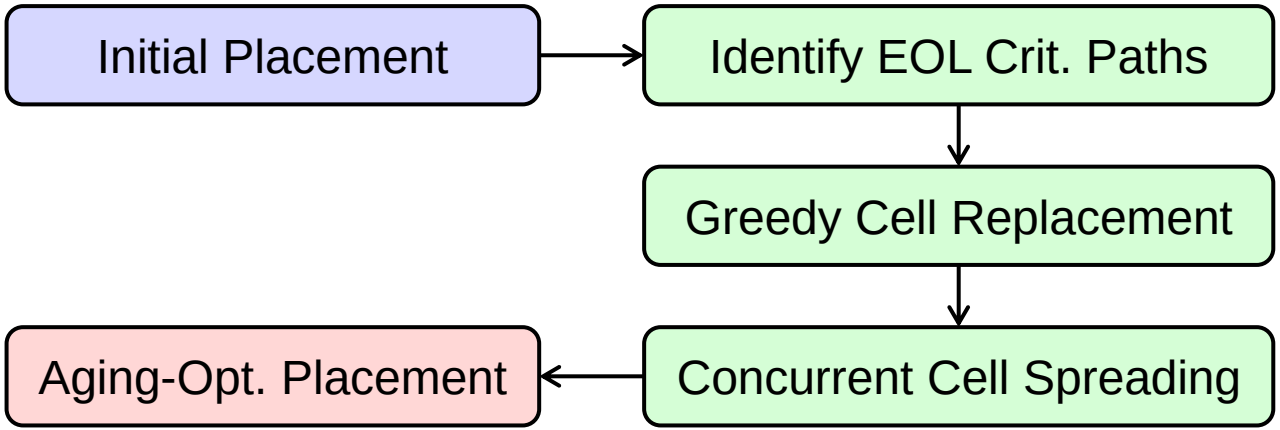
	INV_X2	INV_X2_Dual
LOD_l	25nm	25nm
LOD_r	79nm	25nm

	INV_X2	NAND2_X1
Single	9.24 ps	12.63 ps
Dual	10.94 ps	13.72 ps
Δ Delay	18.35%	8.56%

[1] P. Ren et al., IEDM 2015

Placement Refinement for Aging Mitigation

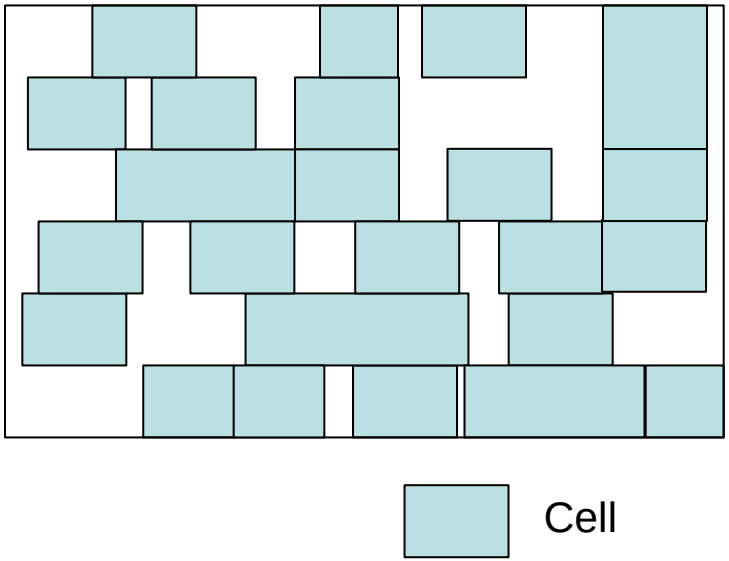
- Post detail placement stage for LDA friendly refinement
- Target at optimizing EOL timing by modifying
  - LOD: determined by cell structure (height) and type
  - OSE: determined by cell to cell spacing
- Overall flow



Placement Refinement for Aging Mitigation

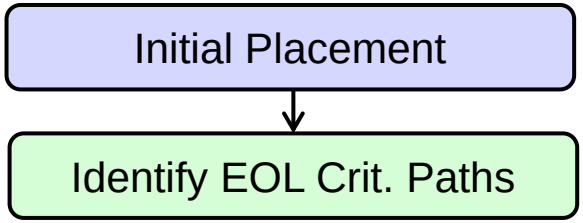
- An illustrative example for the LDA mitigation flow

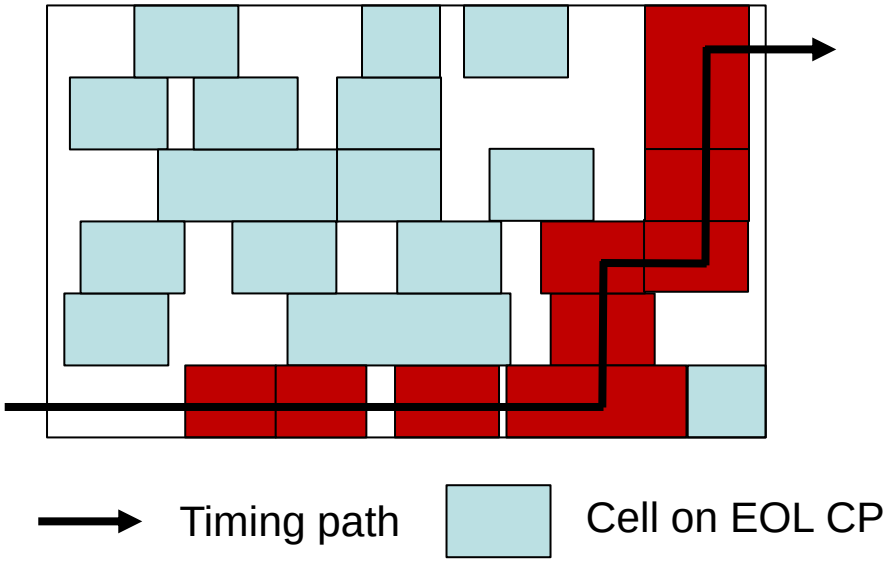
Initial Placement



Placement Refinement for Aging Mitigation

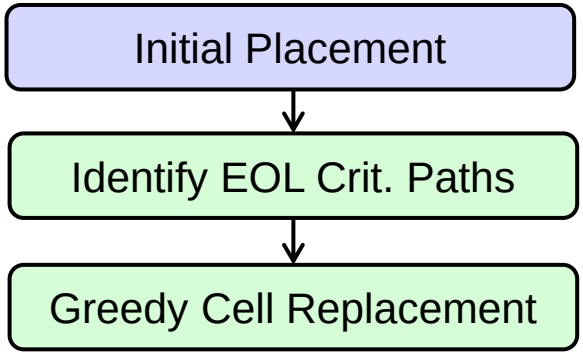
- An illustrative example for the LDA mitigation flow

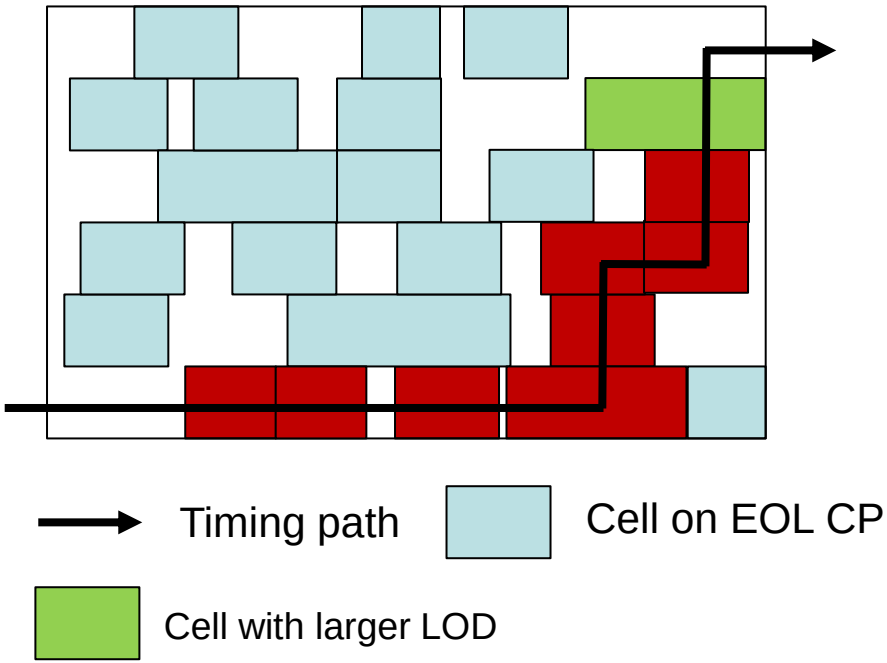




Placement Refinement for Aging Mitigation

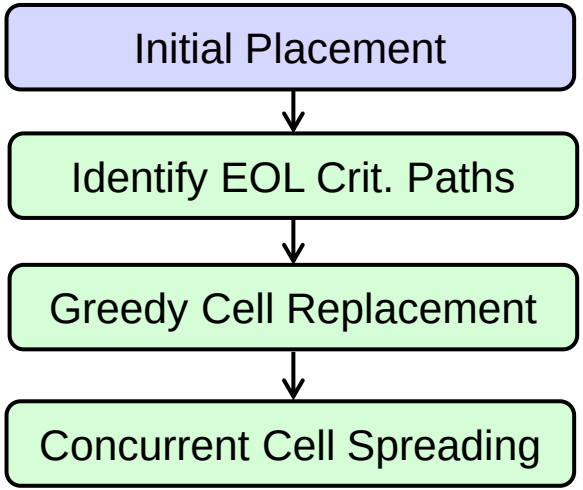
- An illustrative example for the LDA mitigation flow

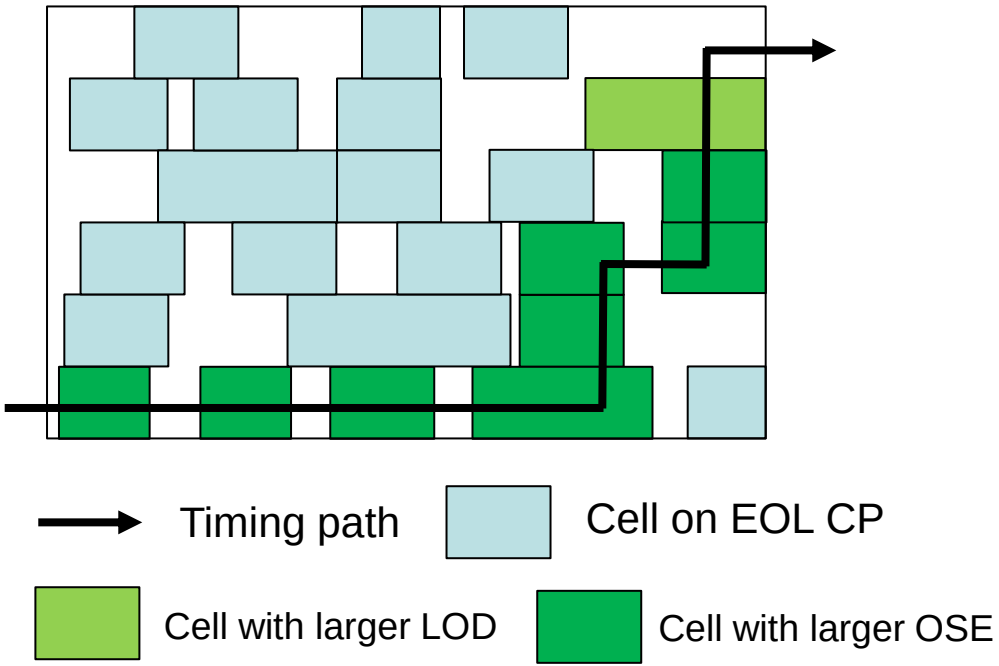




Placement Refinement for Aging Mitigation

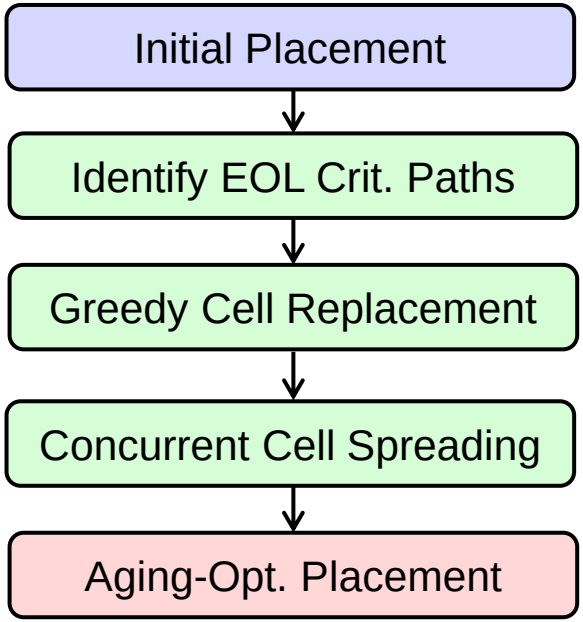
- An illustrative example for the LDA mitigation flow

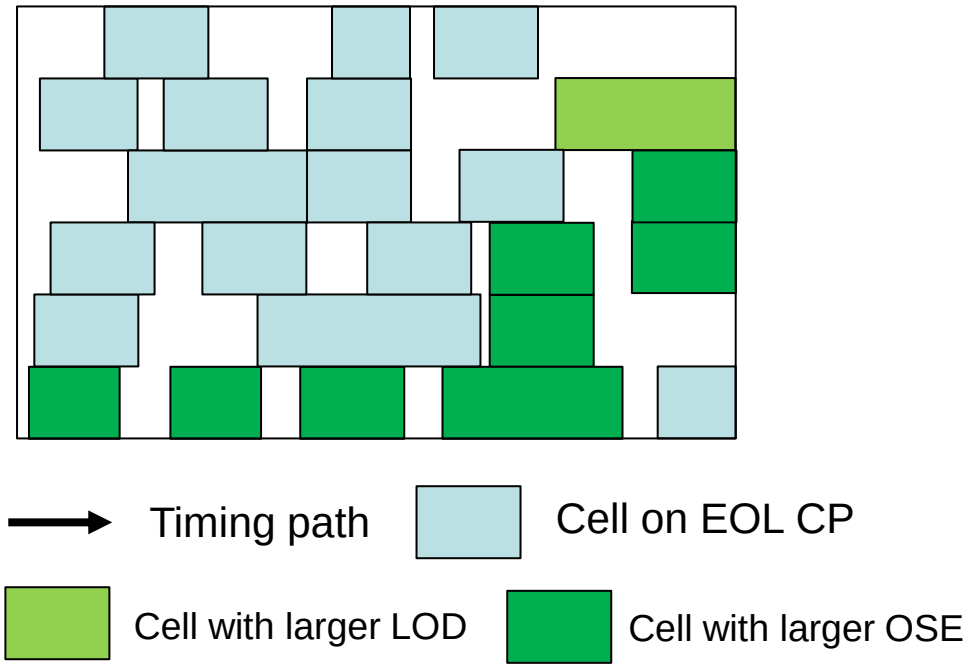




Placement Refinement for Aging Mitigation

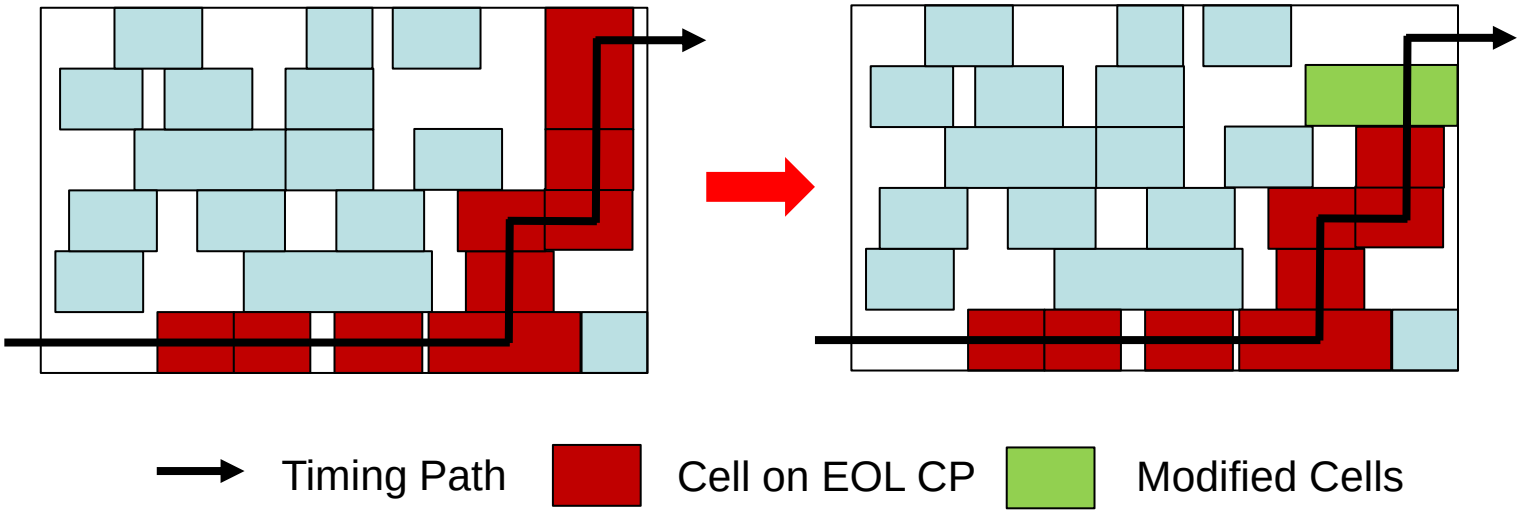
- An illustrative example for the LDA mitigation flow





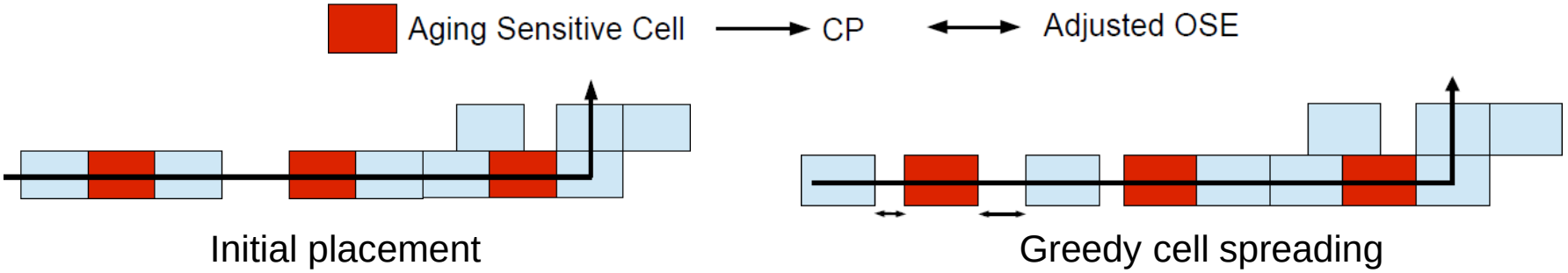
Greedy Cell Replacement

- Objective is to optimize LOD
- Greedy approach
  - Replace cell if the single-row height cell can fit in initial horizontal spacing
  - Congestion can be considered for pin accessibility



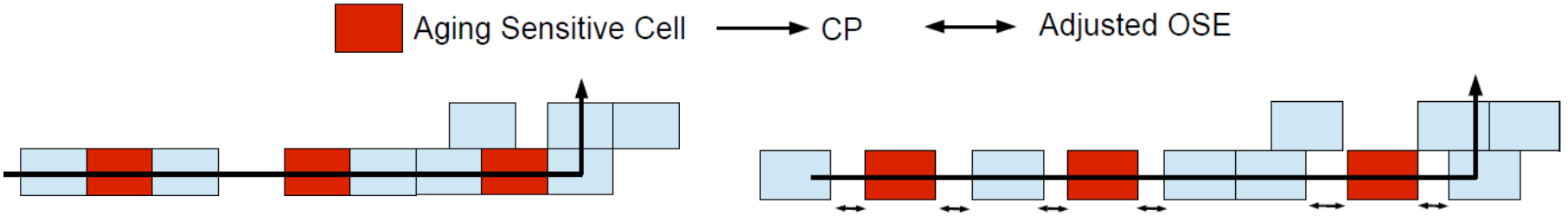
Greedy Cell Spreading

- Objective is to optimize OSE
- Naïve method: greedy cell spreading
  - Sort cells on critical paths based on the OSE sensitivity (Δdelay/ΔOSE)
  - Optimize cell with steepest gradient first
  - Simple implementation and fast run time
- However, greedy cell spreading affects other cells' OSE



Concurrent Cell Spreading

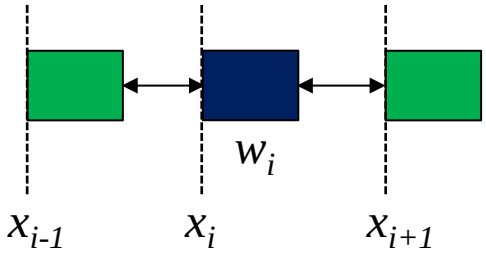
- Simultaneously optimize white space on critical paths
  - Provide holistic OSE improvements for different cells
- Can be formulated as a linear programming (LP) problem



LP-based Concurrent Cell Spreading

- Objective: balance between aging mitigation and cell displacements

$$\min \sum_i -f(x_{i+1}, x_i, x_{i-1}) + \sum_i |x_i - x_i^o|$$

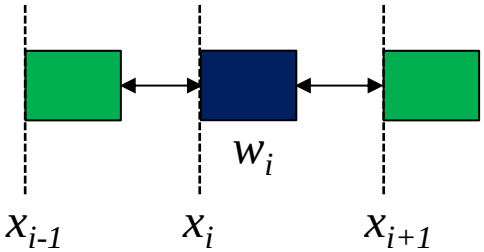


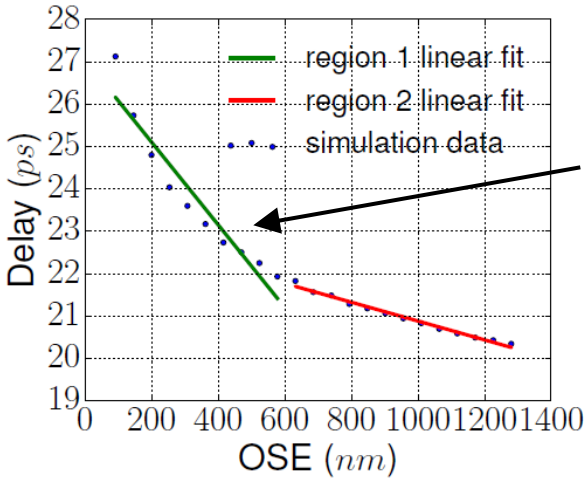
LP-based Concurrent Cell Spreading

- Objective: balance between aging mitigation and cell displacements

$$\min \sum_i -k_i(x_{i+1} - x_{i-1}) + \sum_i |x_i - x_i^o|$$

  - Aging mitigation approximated as a piecewise linear function of OSE





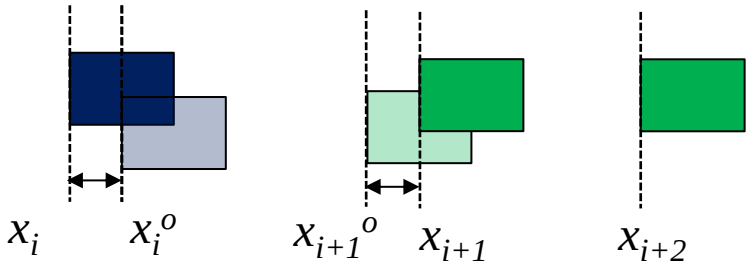
-k_i term comes from aging simulation

LP-based Concurrent Cell Spreading

- Objective: balance between aging mitigation and cell displacements

$$\min \sum_i -k_i(x_{i+1} - x_{i-1}) + \sum_i |x_i - x_i^o|$$

  - Nonlinear term is not preferred in the objective function
  - Define d_i^r / d_i^l is the right/left edge (upper/lower bound) of the displacement



$$d_i^r = \max(x_i, x_i^o)$$

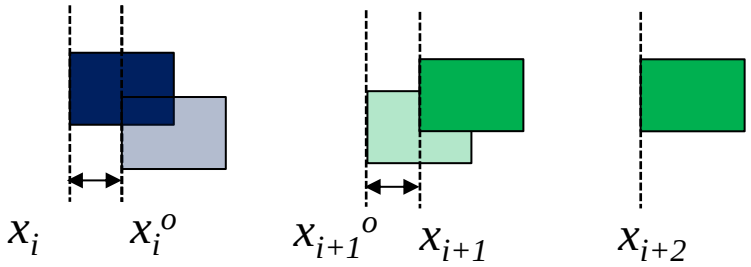
$$d_i^l = \min(x_i, x_i^o)$$

LP-based Concurrent Cell Spreading

- Objective: balance between aging mitigation and cell displacements

$$\min \sum_i -k_i(x_{i+1} - x_{i-1}) + \sum_i (d_i^r - d_i^l)$$

  - Nonlinear term is not preferred in the objective function
  - Define d_i^r / d_i^l is the right/left edge (upper/lower bound) of the displacement



$$d_i^r = \max(x_i, x_i^o)$$

$$d_i^l = \min(x_i, x_i^o)$$

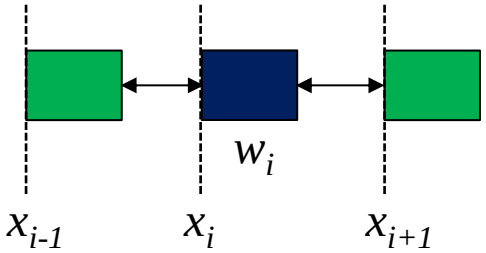
LP-based Concurrent Cell Spreading

- Constraints:
  - Guarantee no overlapping between cells

$$x_{i+1} - x_i \geq w_i$$

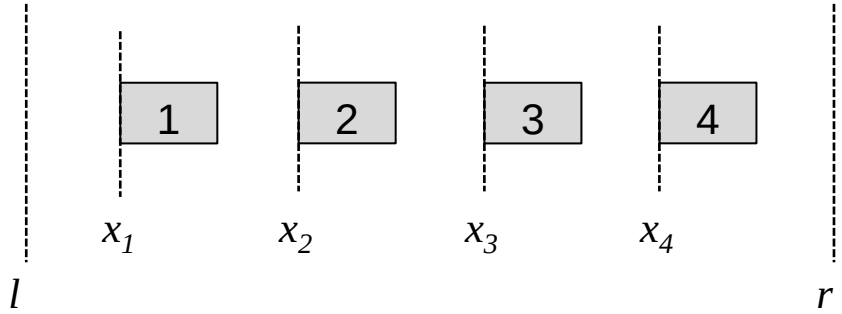
  - Guarantee the relationship between d_i^r, d_i^l, x_i^o and x_i

$$d_i^r - x_i^o \geq 0, \qquad d_i^r - x_i \geq 0$$
$$d_i^l - x_i^o \leq 0, \qquad d_i^l - x_i \leq 0$$



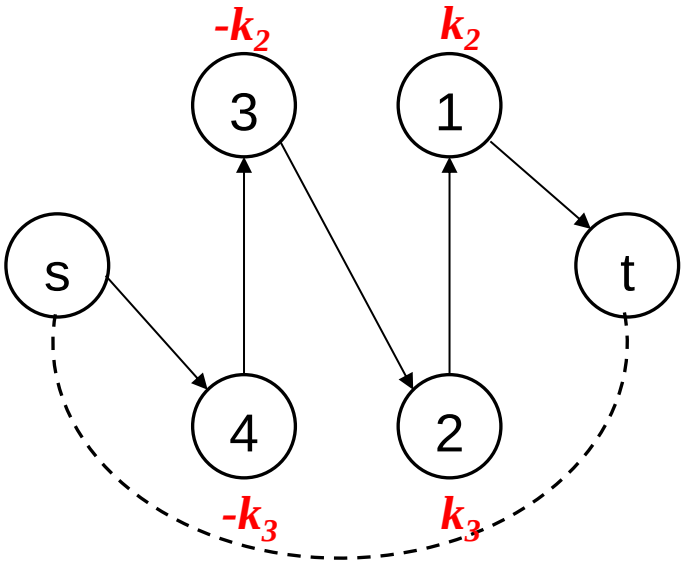
Dual Min-Cost Flow Transformation

- The LP problem can be solved by dual min-cost flow transformation
  - Enable to use fast graph algorithm for acceleration
- Example



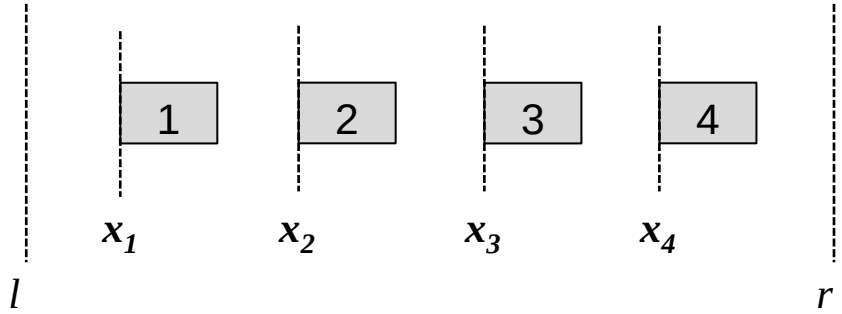
- Objective:

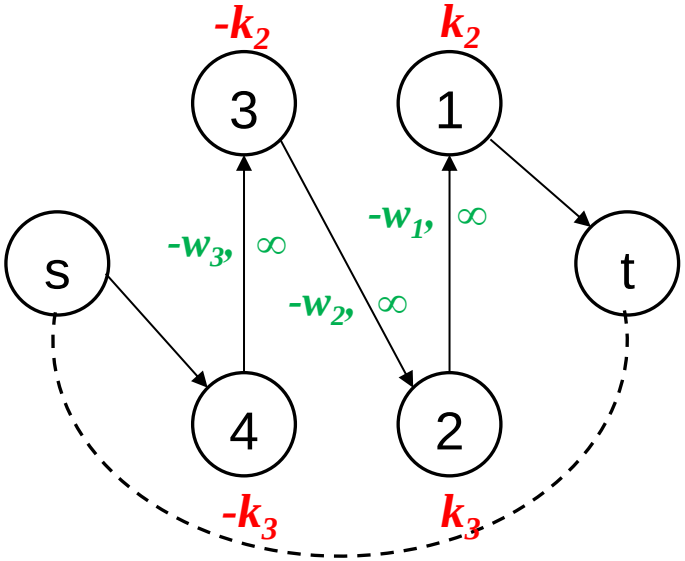
$$-k_2(x_3 - x_1) - k_3(x_4 - x_2)$$



Dual Min-Cost Flow Transformation

- The LP problem can be solved by dual min-cost flow transformation
  - Enable to use fast graph algorithm for acceleration
- Example



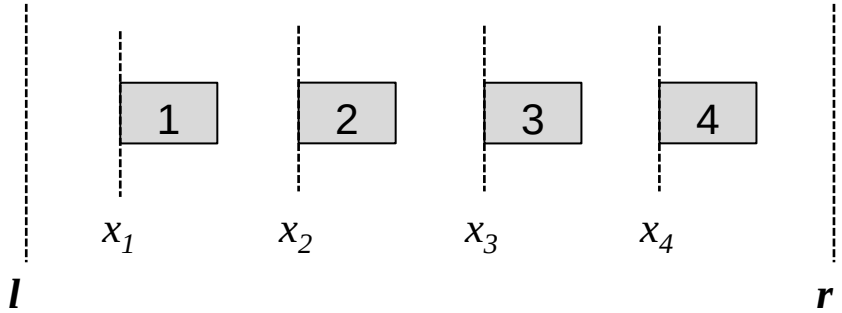


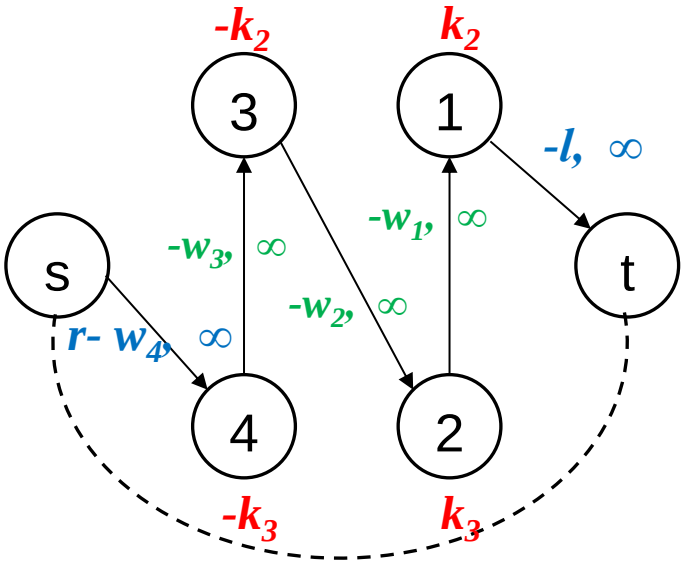
- Constraints:

$$x_2 - x_1 \geq w_1$$
$$x_3 - x_2 \geq w_2$$
$$x_4 - x_3 \geq w_3$$

Dual Min-Cost Flow Transformation

- The LP problem can be solved by dual min-cost flow transformation
  - Enable to use fast graph algorithm for acceleration
- Example





- Constraints:

$$x_1 \geq l$$

$$-x_4 \geq w_4 - r$$

Results

- IWLS 2005 benchmark
- Perform EOL timing analysis using Synopsys Primetime
- Compare three different scenarios
  - Without optimization
  - Greedy optimization
  - Concurrent optimization

Results

- Comparing EOL Total Negative Slack (TNS) and Worst Negative Slack (WNS) in greedy optimization
  - On average, TNS reduces by 29.28% after greedy opt.
  - On average, WNS reduces by 16.15% after greedy opt.

ckt	EOL Timing w/o Opt.		EOL Timing w/ Greedy Opt.				
	TNS (ps)	WNS (ps)	TNS (ps)	ΔTNS (%)	WNS (ps)	ΔWNS (%)	cpu (s)
ss_pcm	182.79	37.15	153.14	18.48%	33.69	9.31%	0.01
simple_spi	202.59	29.27	71.69	64.61%	15.54	46.90%	0.03
sasc	211.40	30.35	149.81	29.13%	27.36	9.85%	0.02
tv80s	2436.16	77.55	1860.37	23.64%	65.67	15.32%	0.18
ac97_ctrl	401.18	28.12	267.06	33.43%	22.31	20.66%	0.43
usb	352.80	61.76	331.46	6.05%	61.66	0.16%	0.49
aes_core	220.81	47.69	155.29	29.67%	42.53	10.82%	0.58
Average Reduction			—	**29.28%**	—	**16.15%**	—

Results

- Comparing EOL Total Negative Slack (TNS) and Worst Negative Slack (WNS) in concurrent optimization
  - On average, TNS reduces by 42.35% after concurrent opt.
  - On average, WNS reduces by 25.19% after concurrent opt.

ckt	EOL Timing w/o Opt.		EOL Timing w/ Concurrent Opt.				
	TNS (ps)	WNS (ps)	TNS (ps)	ΔTNS (%)	WNS (ps)	ΔWNS (%)	cpu (s)
ss_pcm	182.79	37.15	144.09	23.27%	32.29	13.08%	0.17
simple_spi	202.59	29.27	58.31	71.22%	14.06	51.96%	0.34
sasc	211.40	30.35	128.91	39.02%	25.61	15.62%	0.34
tv80s	2436.16	77.55	1598.48	34.39%	59.61	23.13%	2.99
ac97_ctrl	401.18	28.12	172.22	57.07%	16.59	41.00%	6.60
usb	352.80	61.76	235.34	33.29%	51.41	16.76%	7.51
aes_core	220.81	47.69	136.41	38.22%	40.65	14.76%	8.87
Average Reduction			—	**42.35%**	—	**25.19%**	—

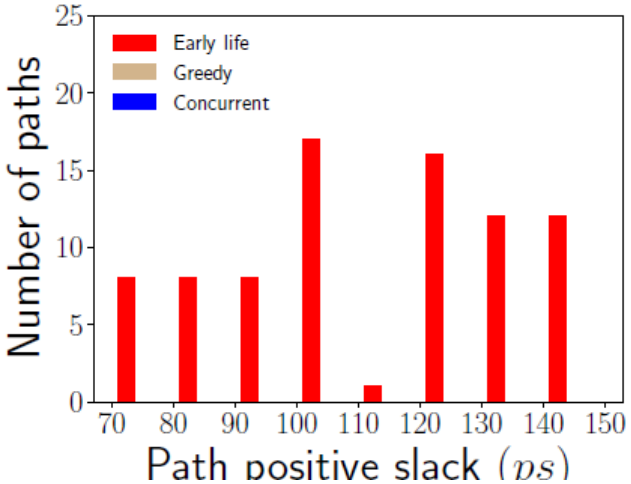
Results

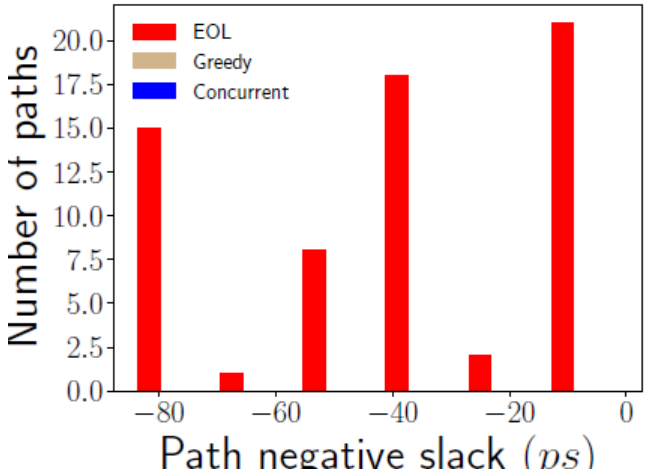
- Comparing early-life timing before and after greedy/concurrent opt.
  - D_o: CP delay for original placement
  - D_g: CP delay after greedy opt.
  - D_c: CP delay after concurrent opt.
  - On average, less than 0.2% of timing change

ckt	Early-life Timing			% change from original	
	D_o (ps)	D_g (ps)	D_c (ps)	ΔD_g (%)	ΔD_c (%)
ss_pcm	336.42	336.76	337.18	0.10%	0.23%
simple_spi	342.03	342.02	342.37	0.00%	0.10%
sasc	290.96	291.78	292.82	0.28%	0.64%
tv80s	749.93	753.79	753.48	0.51%	0.47%
ac97_ctrl	440.41	441.33	439.06	0.21%	-0.31%
usb	626.84	627.53	637.91	0.11%	1.76%
aes_core	575.48	561.67	563.10	-2.40%	-2.20%
Average Change				-0.17%	0.10%

Results

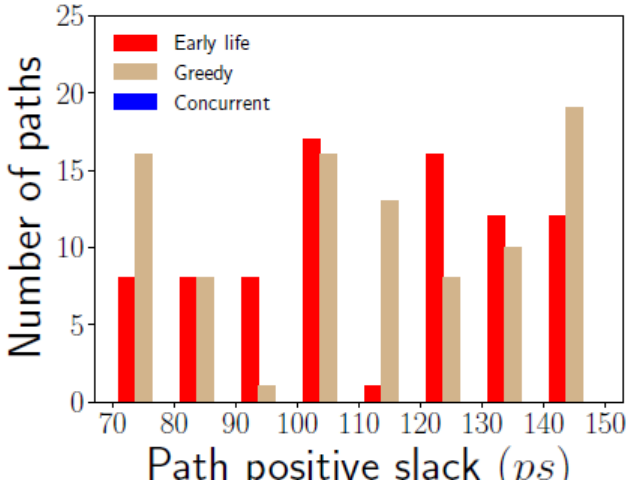
- Detained statistics on *ckt tv80s*
  - Original placement (Early life and EOL)

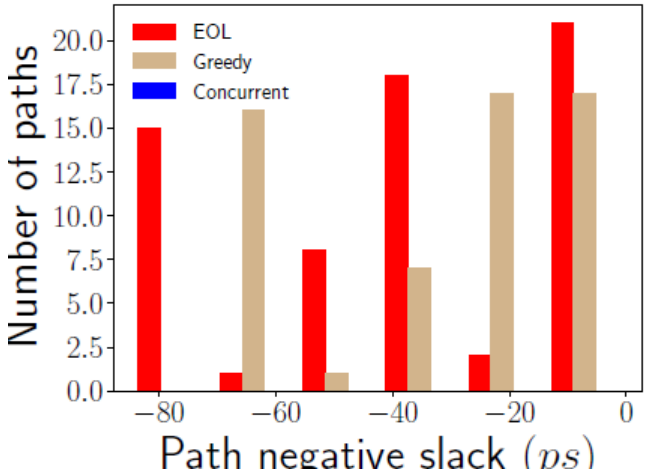




Results

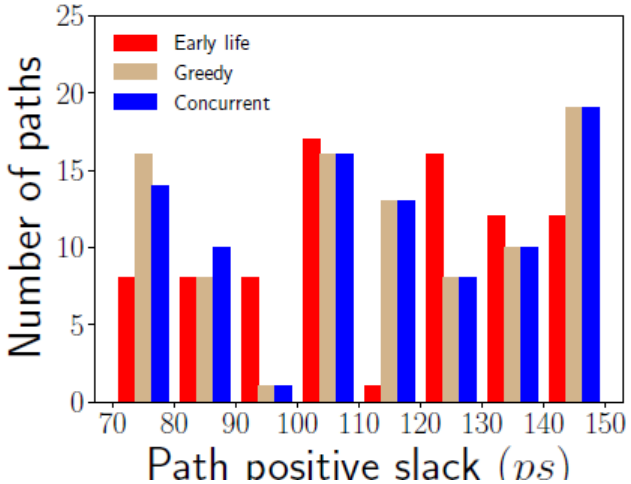
- Detained statistics on *ckt tv80s*
  - Greedy optimization

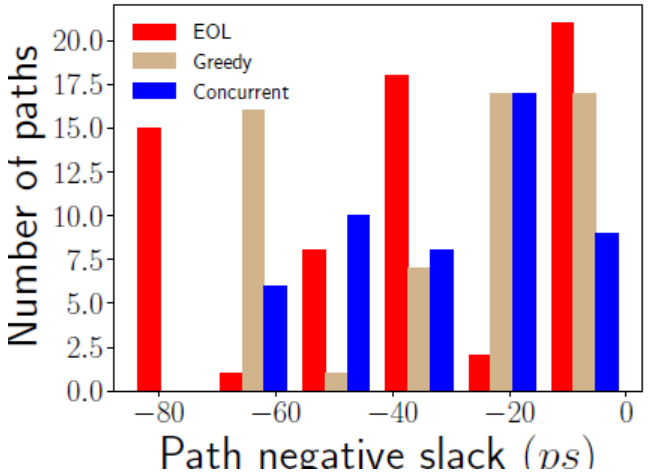




Results

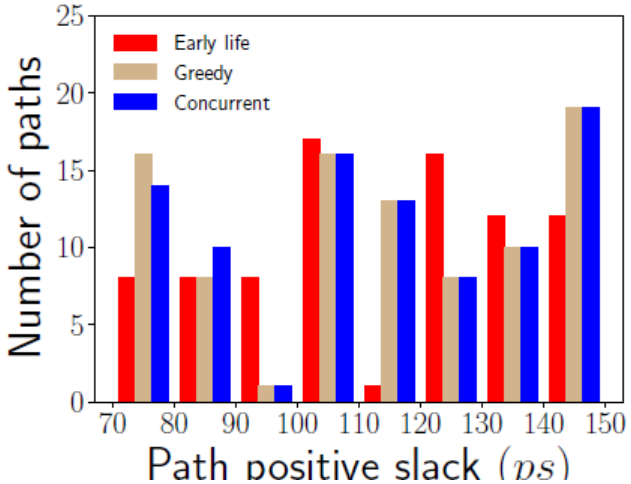
- Detained statistics on *ckt tv80s*
  - Concurrent optimization

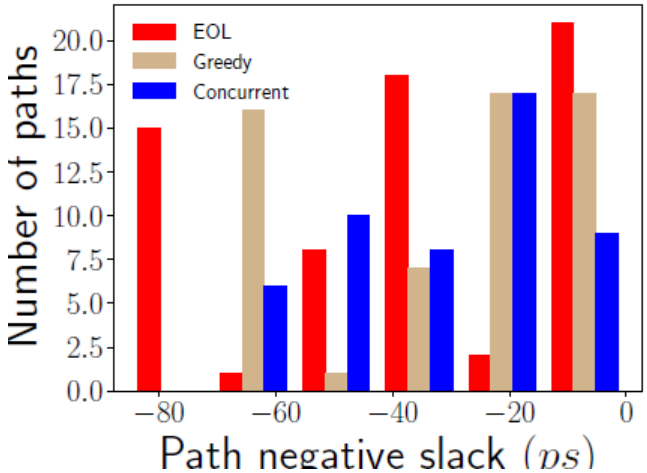




Results

- Detained statistics on *ckt tv80s*
  - Distribution shifts right means more timing slack





Conclusions

- LDA degrades EOL timing performance
- We propose Cell replacement and cell spreading techniques for placement refinement
- Proposed framework reduced EOL total negative slack by 42% and worst negative slack by 25% on average

THANK YOU!

Dual min-cost flow transformation

- Original form to Prime form and then transform to dual form

$$\min_{x_i} \sum_{i=1}^{N} c_i x_i$$
$$\text{s.t. } x_i - x_j \geq b_{ij}, \ (i,j) \in E,$$
$$l_i \leq x_i \leq u_i, \ i = 1, 2, ..., N,$$
$$x_i \in Z$$

Prime

$$\min_{y_i} \sum_{i=0}^{N} c'_i y_i,$$
$$\text{s.t. } y_i - y_j \geq b'_{ij}, \ (i,j) \in E',$$
$$y_i \in Z,$$

where

$$x_i = y_i - y_0, \quad i = 1, 2, ..., N$$

$$c'_i = \begin{cases} c_i & i = 1, 2, ..., N \\ -\sum_{i=1}^{N} c_i & i = 0 \end{cases}$$

$$b'_{ij} = \begin{cases} b_{ij} & (i,j) \in E \\ l_i & i = 1, 2, ..., N, j = 0 \\ -u_i & i = 0, j = 1, 2, ..., N \end{cases}$$

Dual

$$\max_{f_{ij}} \sum_{i,j} b'_{ij} f_{ij},$$
$$\text{s.t. } \sum_i f_{ij} - \sum_k f_{jk} = -c'_j,$$
$$f_{ij} \geq 0$$

Dual min-cost flow transformation

- Change original problem to min-cost flow problem

Original problem

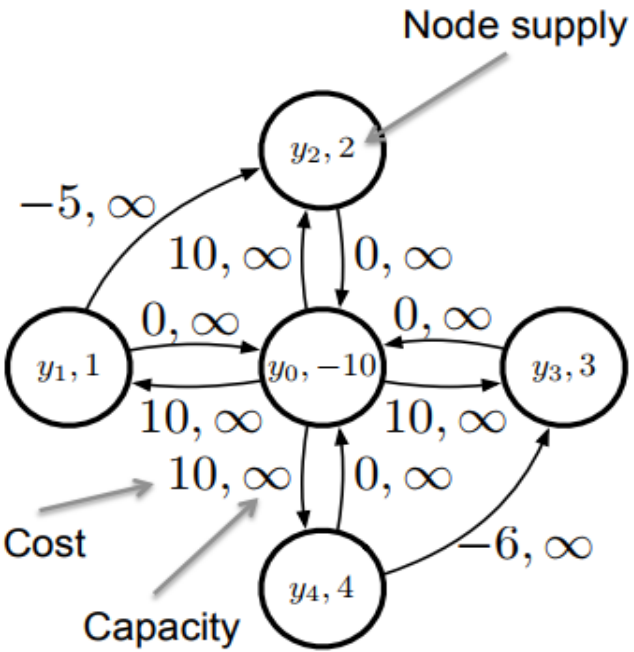
$$
\begin{aligned}
\min \quad & x_1 + 2x_2 + 3x_3 + 4x_4, \\
\text{s.t.} \quad & x_1 - x_2 \geq 5, \\
& x_4 - x_3 \geq 6, \\
& 0 \leq x_i \leq 10, \; i = 1,2,3,4
\end{aligned}
$$

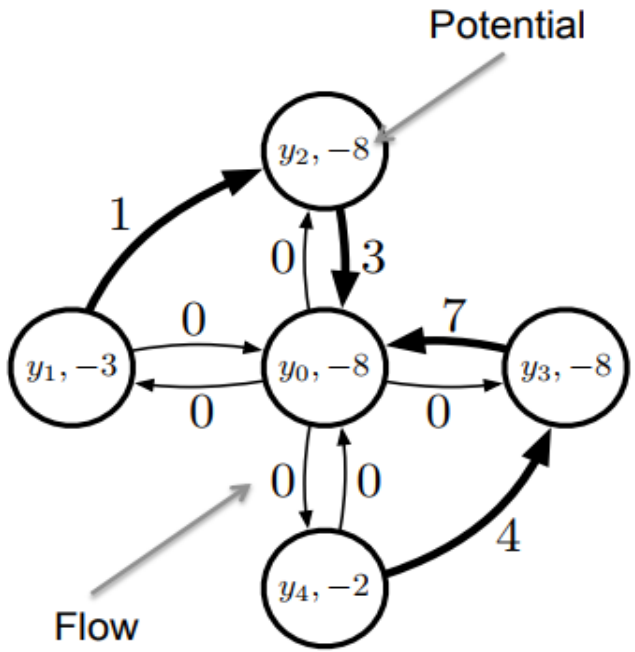
Min-cost flow problem

$$
\begin{aligned}
\min \quad & 10f_{01} + 10f_{02} + 10f_{03} + 10f_{04} - 5f_{12} - 6f_{43}, \\
\text{s.t.} \quad & f_{10} + f_{20} + f_{30} + f_{40} - f_{01} - f_{02} - f_{03} - f_{04} = 10 \\
& f_{01} - f_{10} - f_{12} = -1, \\
& f_{12} + f_{02} - f_{20} = -2, \\
& f_{43} + f_{03} - f_{30} = -3, \\
& f_{04} - f_{40} - f_{43} = -4,
\end{aligned}
$$

Dual min-cost flow transformation

- Graph representation

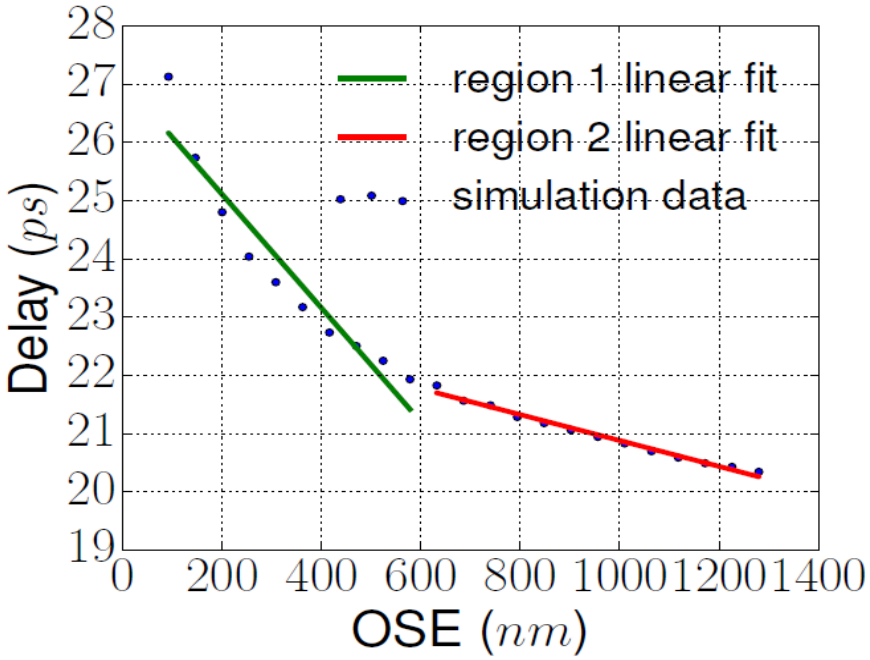




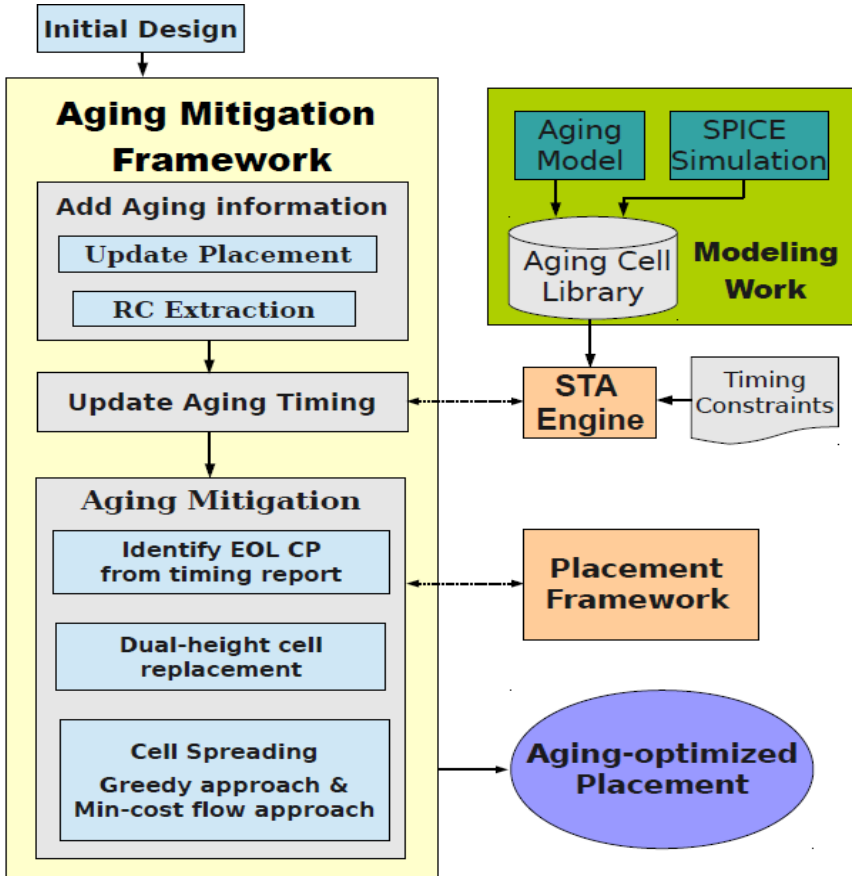
$$x_i = y_i - y_0, \quad i = 1, 2, ..., N$$

Aging Simulation

- Piecewise linear approximation



LDA Mitigation Framework



Results

Cell delay comparison for different OSE

cell	Early-life delay (ps)	Δ $Delay$ ($92nm$)	Δ $Delay$ ($740nm$)
INV_X1	9.01	37.96%	11.30%
NAND2_X1	8.64	46.16%	11.97%
NOR2_X1	13.42	24.63%	12.07%

Cell delay comparison for different cell height.

cell	Early Life			EOL		
	Single (ps)	Dual (ps)	$\Delta Delay$ (%)	Single (ps)	Dual (ps)	$\Delta Delay$ (%)
INV_X2	6.18	7.31	18.27%	9.24	10.94	18.35%
NAND2_X1	8.64	9.50	9.88%	12.63	13.72	8.56%

Results

Comparisons on solution qualities of different aging mitigation approaches

ckt [26]	Early-life Timing			EOL Timing w/o Opt.		EOL Timing w/ Greedy Opt.					EOL Timing w/ Concurrent Opt.				
	D_o (ps)	D_g (ps)	D_c (ps)	TNS (ps)	WNS (ps)	TNS (ps)	ΔTNS (%)	WNS (ps)	ΔWNS (%)	cpu (s)	TNS (ps)	ΔTNS (%)	WNS (ps)	ΔWNS (%)	cpu (s)
ss_pcm	336.42	336.76	337.18	182.79	37.15	153.14	18.48%	33.69	9.31%	0.01	144.09	23.27%	32.29	13.08%	0.17
simple_spi	342.03	342.02	342.37	202.59	29.27	71.69	64.61%	15.54	46.90%	0.03	58.31	71.22%	14.06	51.96%	0.34
sasc	290.96	291.78	292.82	211.40	30.35	149.81	29.13%	27.36	9.85%	0.02	128.91	39.02%	25.61	15.62%	0.34
tv80s	749.93	753.79	753.48	2436.16	77.55	1860.37	23.64%	65.67	15.32%	0.18	1598.48	34.39%	59.61	23.13%	2.99
ac97_ctrl	440.41	441.33	439.06	401.18	28.12	267.06	33.43%	22.31	20.66%	0.43	172.22	57.07%	16.59	41.00%	6.60
usb	626.84	627.53	637.91	352.80	61.76	331.46	6.05%	61.66	0.16%	0.49	235.34	33.29%	51.41	16.76%	7.51
aes_core	575.48	561.67	563.10	220.81	47.69	155.29	29.67%	42.53	10.82%	0.58	136.41	38.22%	40.65	14.76%	8.87
Average Reduction				—	—	—	29.28%	—	16.15%	—	—	42.35%	—	25.19%	—