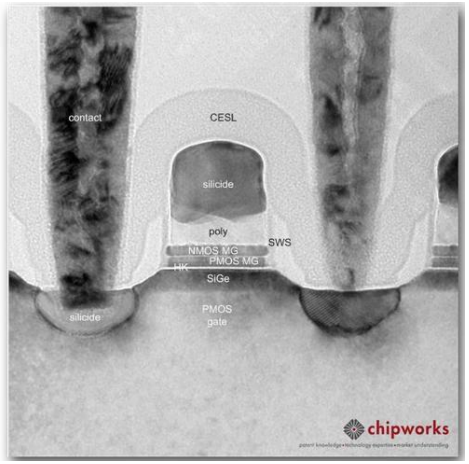
Cell Selection for High-Performance Designs in an Industrial Design Flow

Power

Power/Cooling bill

Battery life



Outline

Context in the Industrial Design Flow

Lagrangian relaxation formulation

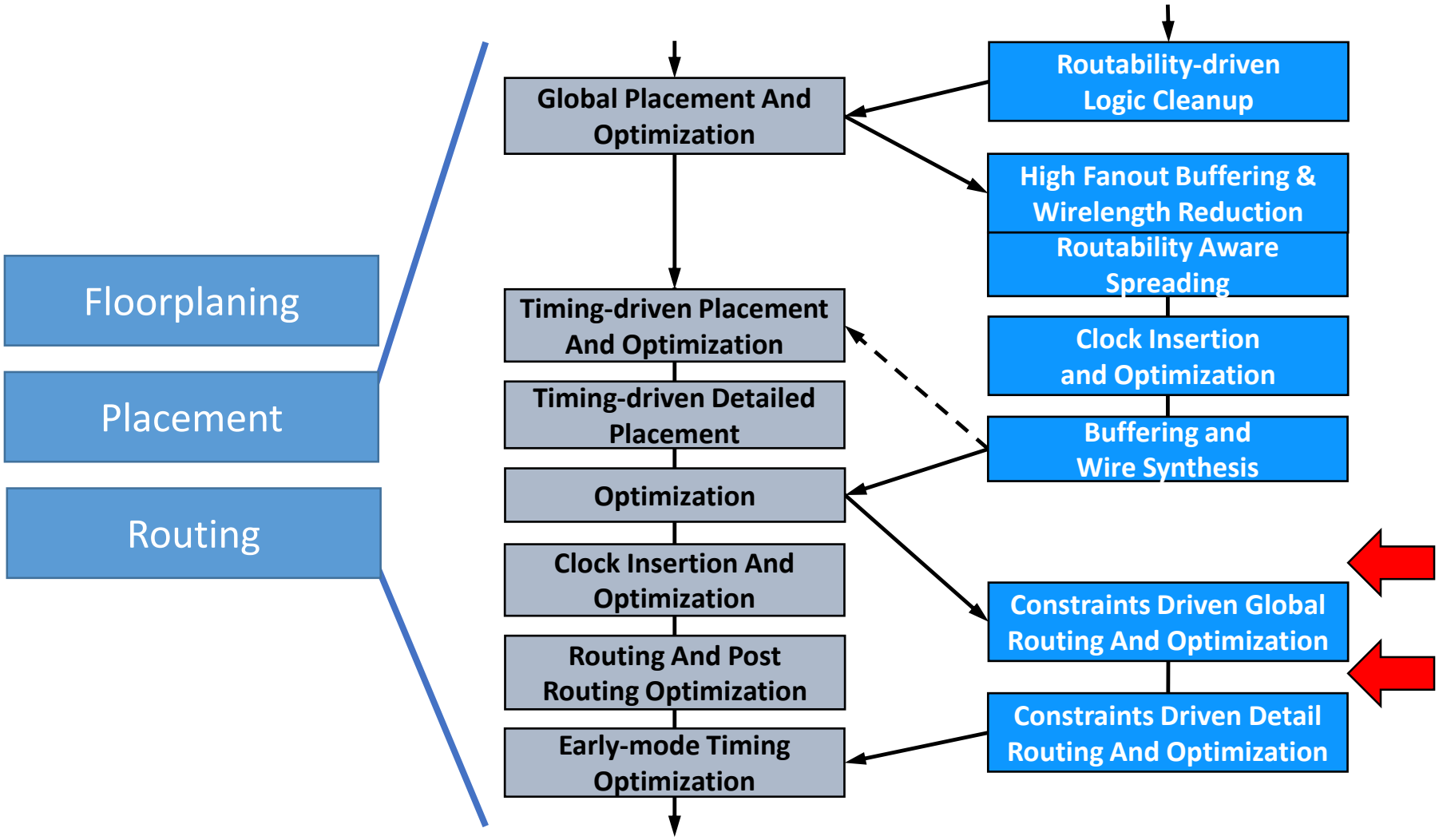
From ISPD Contest to industry: new challenges

Proposed flow with warm start

Results

Conclusions

Physical Design Flows

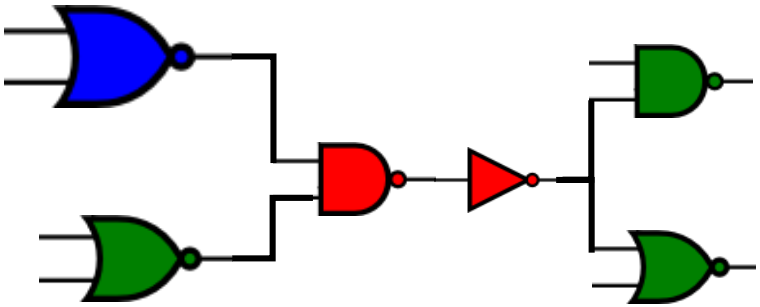


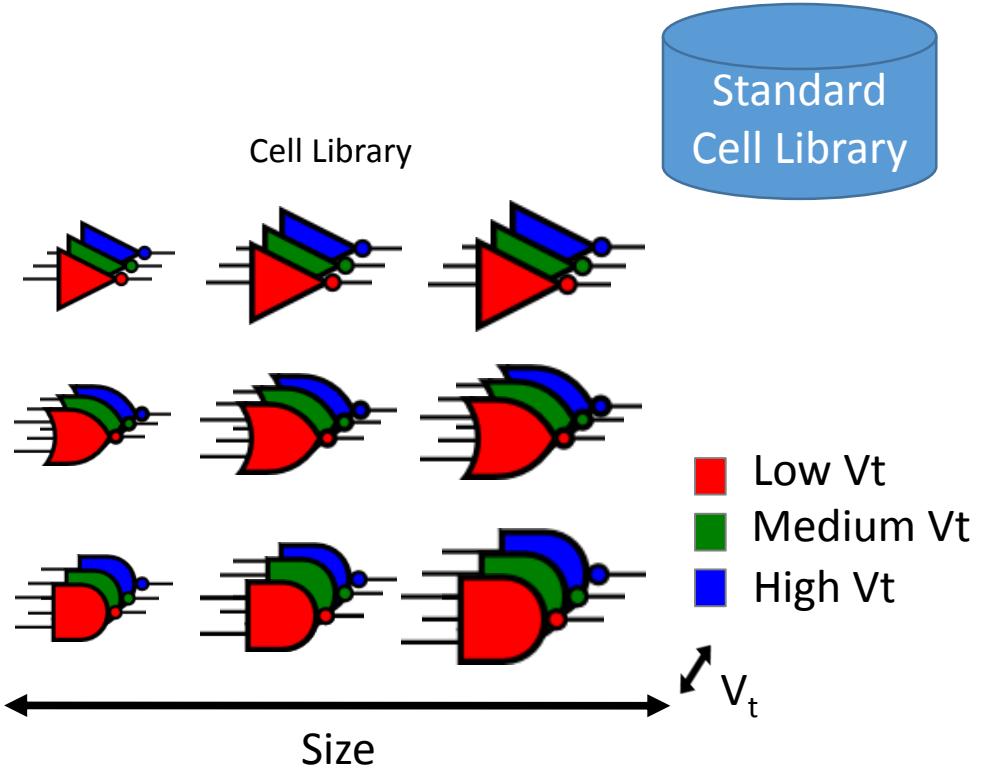
Problem Formulation

Cell Selection

Choose the appropriate cell version from a standard cell library to optimize:

Delay
Area
Power





Lagrangian Relaxation

Overview

Move constraints to the objective to simplify the problem - multiply them by Lagrange multipliers (**λ**).

Relaxed sub-problem (LRS) is a lower bound for the original problem for any set of λ ≥ 0. (Continuous case only)

But, optimality properties of LR do not hold for the discrete sizing problem

Moreover, non-convex timing model

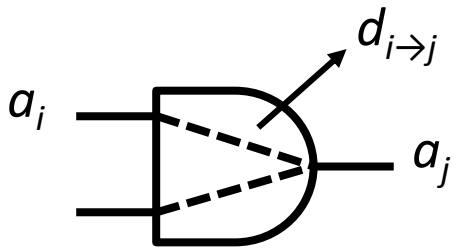
V_t adds more (discrete) dimensions to the solution space

Cell Selection

Mathematical Formulation

$$\begin{aligned} \textbf{minimize} \quad & leakage \\ \textbf{subject to} \quad & a_i + d_{i \to j} \leq a_j, \text{ for each timing arc } i \to j \\ & a_k \leq T, \text{ for each path output node } k \end{aligned}$$

where T is the clock period



a_i	arrival time at input of timing arc
a_j	arrival time at output of timing arc
$d_{i \to j}$	timing arc delay

Lagrangian Relaxation

Relaxing the constraints

Primal Problem

$$\textbf{minimize} \quad leakage$$
$$\textbf{subject to} \quad a_i + d_{i\to j} \le a_j, \text{ for each timing arc } i \to j$$
$$a_k \le T, \text{ for each path output node } k$$

Troublesome constraints moved to the objective.

Relaxed Problem LRS/λ

$$\textbf{minimize} \quad leakage + \sum \lambda_{i\to j}(a_i + d_{i\to j} - a_j) + \sum \lambda_k(a_k - T)$$

Karush-Kuhn-Tucker (KKT) Optimality Conditions

Simplified problem LRS/λ

$$\textbf{minimize} \quad leakage + \sum \lambda_{i\to j} d_{i\to j}$$

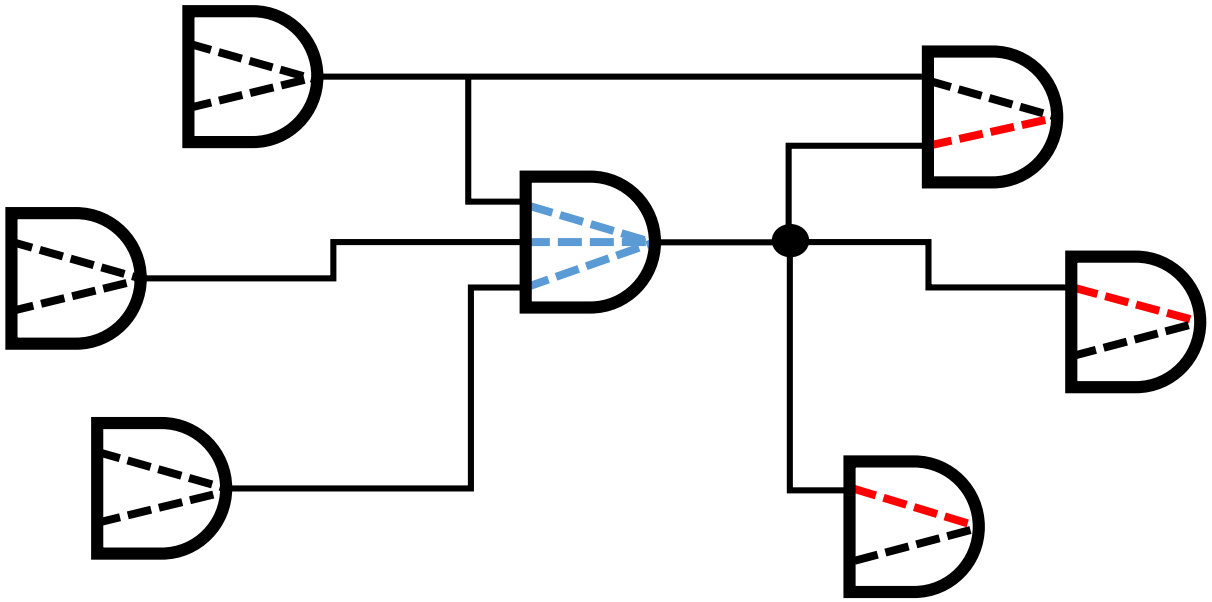
lambda-delay product

Lagrangian Relaxation

Karush-Kuhn-Tucker Conditions

Optimal solution property for inequality constraints:

$$\sum \text{input timing arc } \lambda s = \sum \text{output timing arc } \lambda s$$



New Challenges

From contest to industry

Where to apply global sizing in the flow?

- More timing accuracy is better ("late, post-cts" optimization)
- Runtime increases with accuracy – huge # of timer calls

Optimized initial solution (sizing should be incremental)

- Disruptions may affect quality of results and rest of the flow
- Some fixed cells: **sequential** and "don't touch" cells

Non-fixable negative slacks and max-slew/load violations

- **True total negative slack** (TTNS): include negative slack side paths
- Sizing **must not** degrade timing (for fair comparison)
- Tight timing constraints don't leave much room for improvement

New Challenges

From contest to industry

Placement changes

- Placement legalization must be performed after sizing
- Increase in area may displace cells and degrade timing

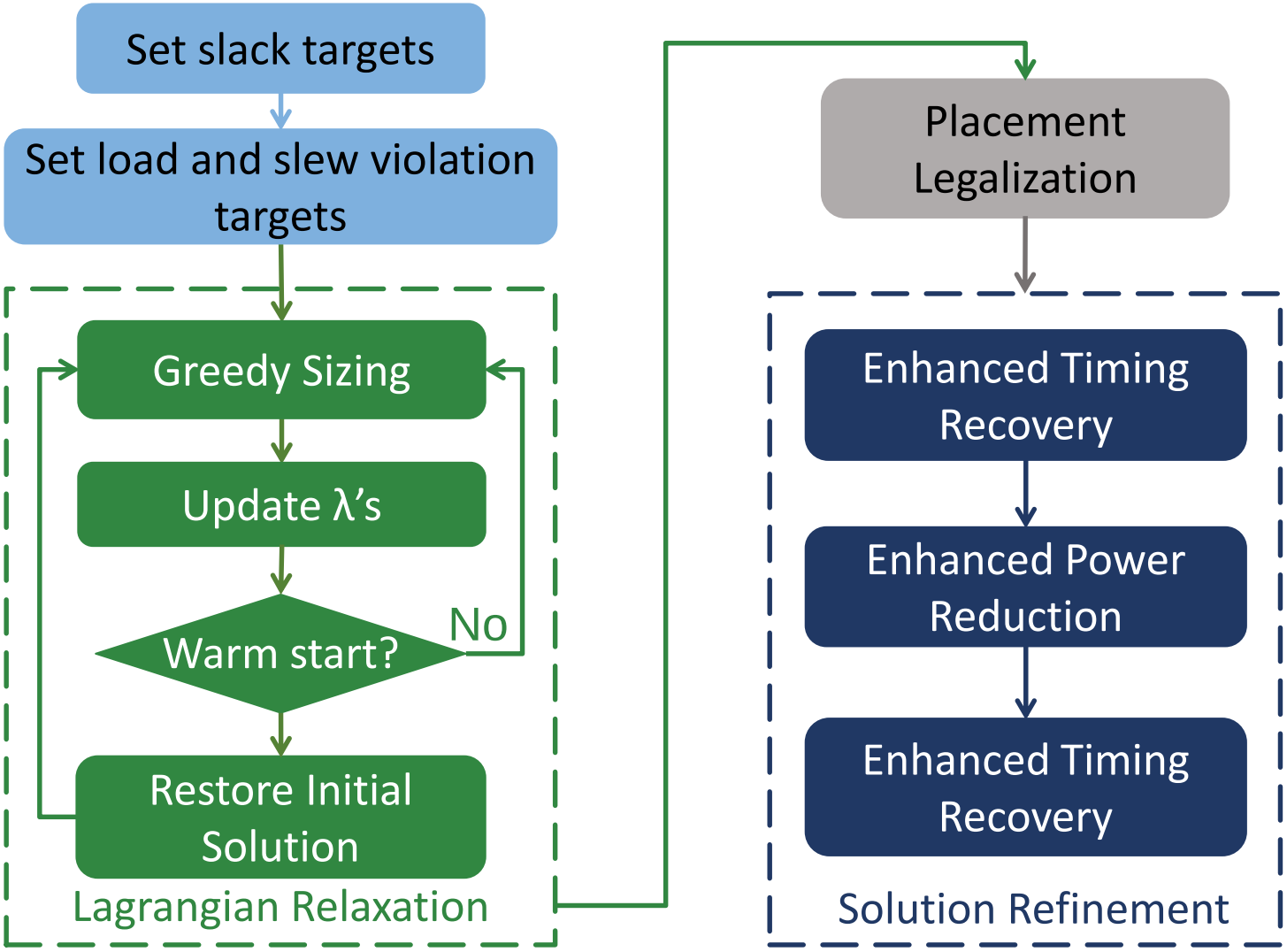
Area consideration

- Important when increasing V_t and area results in less leakage

Dynamic power

- Although calculator was not accurate at that time

New Proposed Flow



Warm Start

Improving Initial Lambda

Optimized initial solution provided

- Adjust lambdas to reflect initial solution
- Improves convergence and quality of results
- Sizes/V_t must reflect lambdas (stability and convergence)

Warm start algorithm, with fast delay estimation

- Normal LR iteration
- Reset sizing after updating λ values
- Use fast delay estimation to improve runtime
- Accurate mode iterations for final tuning

Setting Timing Targets

Enabling optimization with negative slacks

LR formulation targets zero slack ($a_j \leq T$)

- Critical paths will increase lambdas too much
- Power would increase for no good reason

Set slack target (S_{init}) for each pin in the design

- LR will target the same timing from initial solution
- Not degrade timing for **every pin in the design** (*TTNS*)
- Not possible to set all slacks to zero
- Goal is to find a better balance between timing and power

Lagrange Multiplier Update

Algorithm 3: UpdateLagrangeMultipliers

1 $\rho_{inc} = \rho_{init} \times (1 + iter)$

2 $\rho_{dec} = \rho_{init} \times (15 + iter)$

3 **foreach** *timing arc* $i \rightarrow j$ **do**

4

$$\lambda_i \leftarrow \lambda_i \times \begin{cases} \left(1 - \frac{S_{curr} - S_{init}}{\Delta WNS \times \rho_{inc}}\right)^{1/k} & a_j \geq q_j - S_{init} \\ \left(1 + \frac{S_{curr} - S_{init}}{T \times \rho_{dec}}\right)^{-k} & a_j < q_j - S_{init} \end{cases}$$

5 **end**

6 KKT projection ($\lambda \in \Omega_\lambda$)

New method:

- Slack-based update
- More aggressive for initial iterations
- Faster and more stable convergence

$$k = \begin{cases} 5 & \text{for } \lambda \text{ estimation step} \\ k < 1 & \text{if timing degraded} \\ k > 1 & \text{if solution improved} \\ 1 & \text{otherwise} \end{cases}$$

LRS/λ Solver

With fast delay estimation

For each gate in topological order:

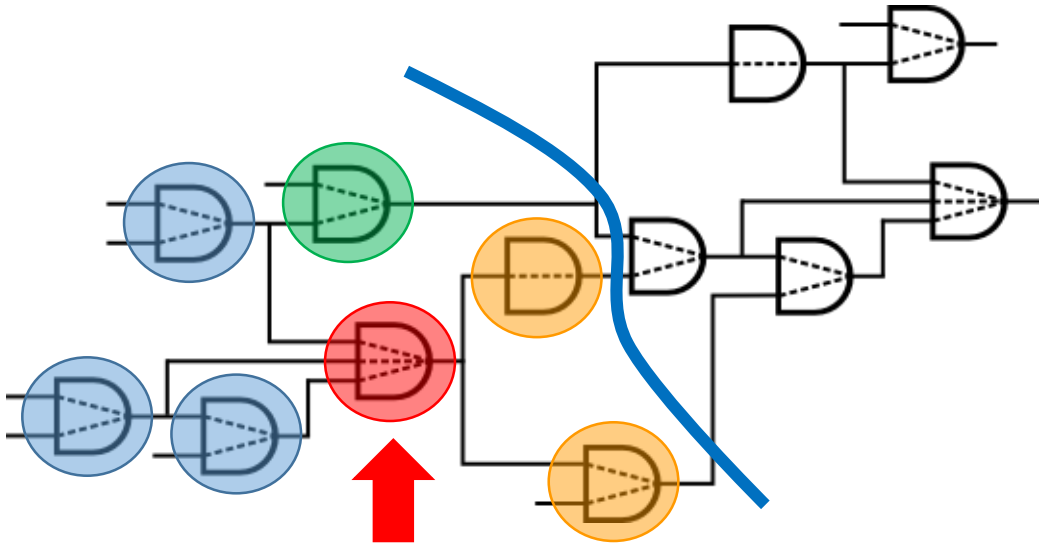
- Filtering method to reduce expensive timer calls
  - Estimate the delays for the new cell option
  - Order options based in the cost function
  - No violation control since it is not accurate
  - Approximation for C_{eff} and slew change
- Evaluate top 2 options with accurate timer
  - Check max-slew/load violations
  - Control local slack changes accurately

LRS/λ Solver

Cost of a Gate Option

Select the option which best trades-off power, area and delay.

$$\beta \times power + \theta \times area + \alpha \times \sum \lambda_{i \to j} d_{i \to j}$$



LRS/λ Solver

Scaling factors

$$\beta \times power + \theta \times area + \alpha \times \sum \lambda_{i \to j} d_{i \to j}$$

Allows the cost function to balance different objectives

- Remove all units
- Calculation based on library
- Changes in lambda-delay will be equally distributed between objectives

delay

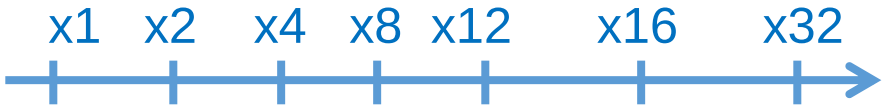
$$\alpha = \frac{N_{C_{REF}}}{D(c_0) - D(c_n)}$$ ps^{-1}

power

$$\beta = \frac{N_{C_{REF}}}{P_l(c_n) - P_l(c_0)}$$ µW^{-1}

area

$$\theta = \frac{N_{C_{REF}}}{A(c_n) - A(c_0)}$$ µm^{-2}



c_n : largest cell option

c_0 : smallest cell option

Nc_{REF} : # cell options

Greedy Solution Refinement

Enhanced Timing Recovery

- Change cells 1-by-1 to improve slack
- **Order cells by slack (most critical first)**
  - downsize sink cells of critical paths
  - upsize cells on critical paths
  - lower V_t of cells on critical paths
- Check violations, ensure no timing degradation

Enhanced Power Reduction

- Change cells 1-by-1 to improve power/area
- **Order cells by slack (most critical first)**
  - downsize cells
  - increase V_t
- Check violations, ensure no timing degradation

Benchmarks

Several representative designs

22nm technology / 5GHz operating frequency / 174ps clock period
IBM Z mainframe microprocessor blocks

Design	#Gates	WNS	TNS	TTNS	Leakage Power	Dynamic Power	Total Power	Area
ibm2014uP_01	95K	-78.3	-144167	-910468	80.6	13.5	94.1	809.1
ibm2014uP_02	9K	-135.1	-2973	-22778	1.1	1.3	2.4	58.5
ibm2014uP_03	9K	8.9	-14	-30	2.8	51.4	54.1	67.3
ibm2014uP_04	7K	-8.4	-552	-560	1.6	1.3	2.9	72.7
ibm2014uP_05	15K	-82.1	-43263	-76068	19.1	45.3	64.4	134.9
ibm2014uP_06	75K	-142.6	-36833	-62323	37.7	112.0	149.7	777.0
ibm2014uP_07	70K	-39.4	-54401	-392551	61.0	12.6	73.6	637.2
ibm2014uP_08	18K	-72.9	-37290	-195777	16.7	68.4	85.1	148.5
ibm2014uP_09	17K	-32.7	-14491	-71828	14.7	33.0	47.7	150.9
ibm2014uP_10	124K	-34.2	-70737	-322544	86.2	304.5	390.7	990.4
ibm2014uP_11	24K	-165.3	-195168	-1054130	35.3	21.5	56.8	235.7
ibm2014uP_12	17K	-421.9	-359981	-777205	4.3	20.5	24.7	161.1
ibm2014uP_13	20K	-49.8	-26647	-133504	20.3	61.2	81.5	196.4
ibm2014uP_14	13K	-61.9	-6526	-11213	8.2	9.7	17.9	251.9

Results

Permitting TTNS Degradation

Global sizing/V_t in post global route optimization

Using baseline solution from synthesis flow
After power optimization step performed in original flow
Degrades side paths with negative slacks – TTNS

Design	#Gates	CPU (min)	Worst Slack before	Worst Slack after	TNS before	TNS diff	TTNS before	TTNS diff	Power Leakage	Power Dynamic	Area
ibm2014uP_01	70K	295	-39.40	-39.33	-54401	1541	-392551	-21700	**-16.8%**	**0.0%**	**-1.9%**
ibm2014uP_02	95K	402	-78.34	-77.72	-144167	11691	-910468	-10236	**-20.0%**	**-0.4%**	**-2.6%**
ibm2014uP_03	9K	42	8.87	8.95	-14	8	-30	20	**5.2%**	**-2.9%**	**-0.3%**
ibm2014uP_04	15K	32	-82.13	-82.02	-43263	753	-76068	-726	**-6.1%**	**-0.5%**	**-0.2%**
ibm2014uP_05	17K	79	-32.74	-32.27	-14491	1248	-71828	-16244	**-21.8%**	**0.1%**	**-2.7%**
ibm2014uP_06	20K	65	-49.79	-49.75	-26647	1060	-133504	-4192	**-15.5%**	**-0.5%**	**-2.0%**
ibm2014uP_07	24K	158	-165.27	-165.23	-195168	5521	-1054130	-21750	**-18.0%**	**0.0%**	**-0.5%**
ibm2014uP_08	124K	544	-34.20	-34.20	-70737	1862	-322544	-37498	**-20.7%**	**-3.3%**	**-3.5%**
ibm2014uP_09	18K	87	-72.89	-72.89	-37290	2785	-195777	-18585	**-19.8%**	**-1.0%**	**-2.7%**
ibm2014uP_10	13K	13	-61.86	-61.16	-6526	143	-11213	144	**-0.5%**	**0.0%**	**-0.1%**
ibm2014uP_11	17K	63	-421.88	-421.88	-359981	11012	-777205	90	**-11.3%**	**-2.3%**	**-7.5%**
ibm2014uP_12	9K	57	-135.13	-134.29	-2973	386	-22778	-4139	**-8.3%**	**-4.0%**	**-6.3%**
ibm2014uP_13	7K	13	-8.38	-7.40	-552	37	-560	39	**-0.4%**	**-0.5%**	**-0.1%**
ibm2014uP_14	75K	296	-142.57	-142.57	-36833	1294	-62323	-4712	**-4.0%**	**0.0%**	**-0.2%**
Average						2810		-9963	**-11.3%**	**-1.1%**	**-2.2%**
Total						39341		-139489	**-16.3%**	**-1.8%**	**-2.1%**

Results

Without TTNS Degradation

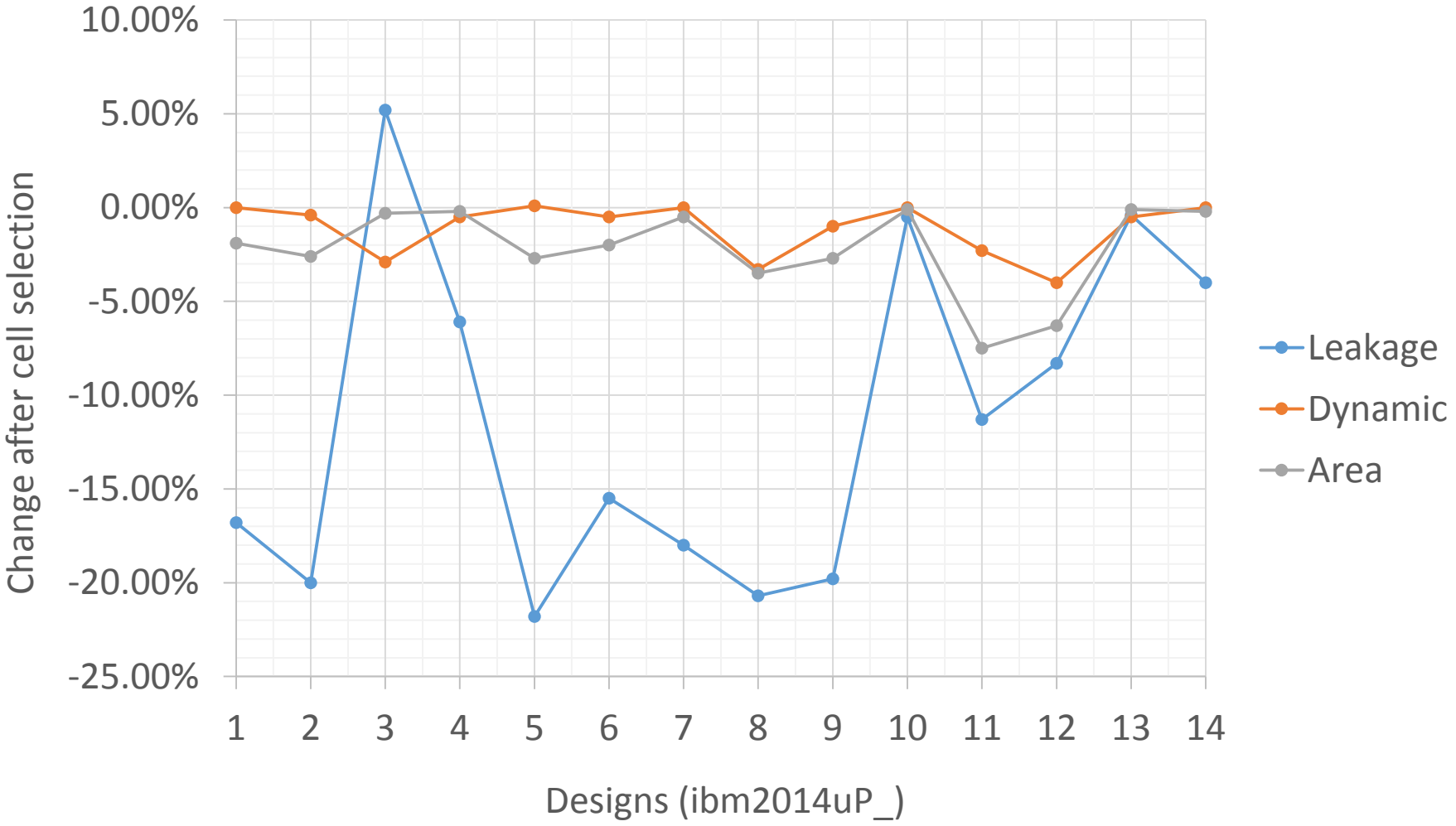
Global sizing/V_{th} post global route optimization

No TTNS degradation

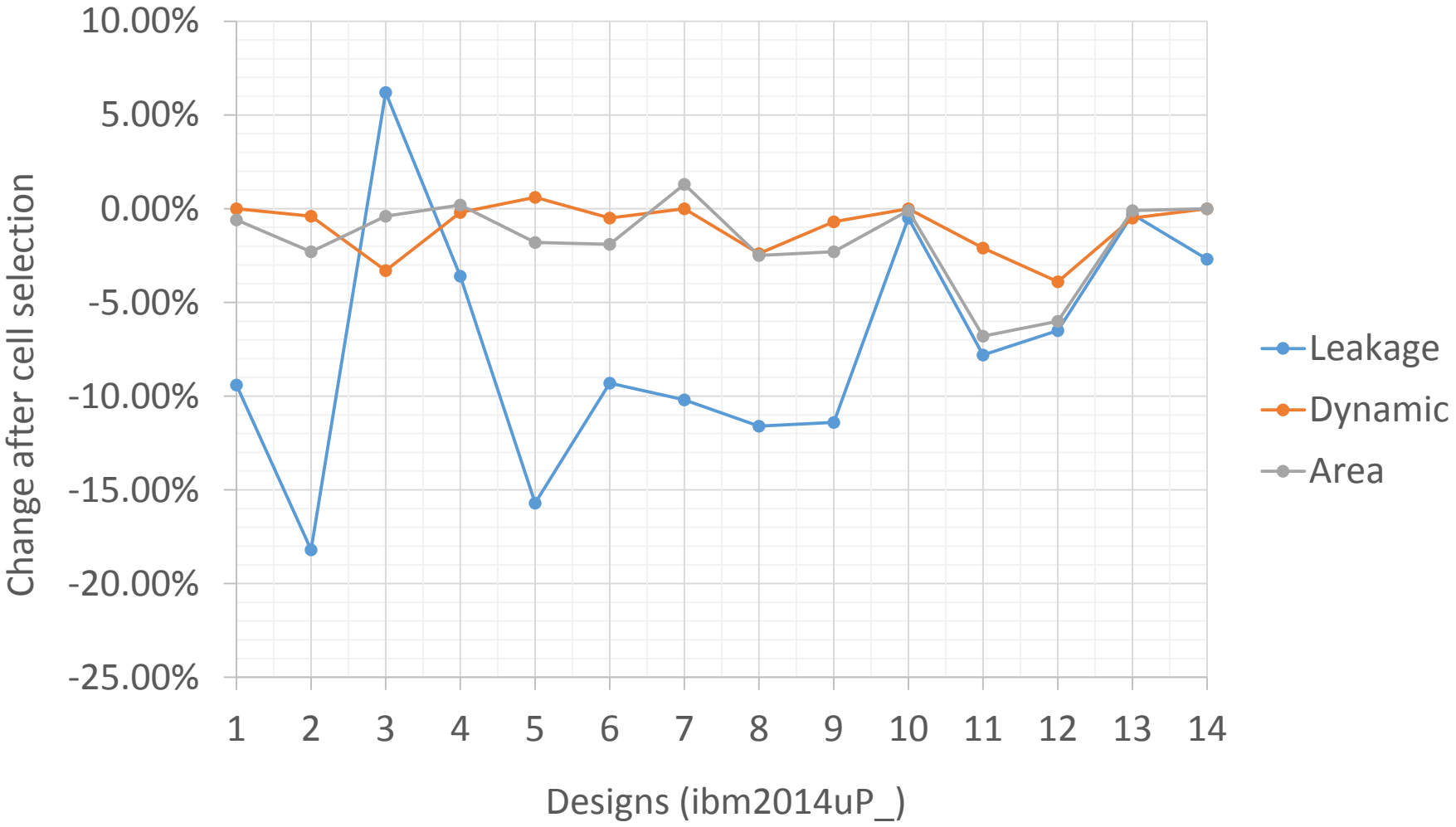
Less power reduction

Design	#Gates	CPU (min)	Worst Slack before	Worst Slack after	TNS before	TNS diff	TTNS before	TTNS diff	Power Leakage	Power Dynamic	Area
ibm2014uP_01	70K	310	-37.92	-37.92	-52108	1982	-376012	18627	**-9.4%**	**0.0%**	**-0.6%**
ibm2014uP_02	95K	377	-75.76	-75.13	-141471	3727	-906301	8442	**-18.2%**	**-0.4%**	**-2.3%**
ibm2014uP_03	9K	53	8.87	8.92	-14	8	-30	20	**6.2%**	**-3.3%**	**-0.4%**
ibm2014uP_04	15K	32	-82.13	-82.12	-43263	500	-76068	893	**-3.6%**	**-0.2%**	**0.2%**
ibm2014uP_05	17K	75	-33.84	-33.07	-14110	386	-72869	1232	**-15.7%**	**0.6%**	**-1.8%**
ibm2014uP_06	20K	55	-53.03	-53.16	-27185	2505	-134543	6187	**-9.3%**	**-0.5%**	**-1.9%**
ibm2014uP_07	24K	104	-165.32	-165.32	-191527	3092	-1034800	12100	**-10.2%**	**0.0%**	**1.3%**
ibm2014uP_08	125K	788	-35.12	-36.35	-67145	2635	-305053	14247	**-11.6%**	**-2.4%**	**-2.5%**
ibm2014uP_09	18K	110	-80.36	-80.36	-39823	2965	-211734	10000	**-11.4%**	**-0.7%**	**-2.3%**
ibm2014uP_10	13K	12	-58.95	-59.05	-6419	-6	-10950	-36	**-0.5%**	**0.0%**	**-0.1%**
ibm2014uP_11	17K	69	-421.65	-421.53	-359783	9748	-787596	23105	**-7.8%**	**-2.1%**	**-6.8%**
ibm2014uP_12	9K	40	-141.00	-140.28	-3030	278	-23004	1348	**-6.5%**	**-3.9%**	**-6.0%**
ibm2014uP_13	7K	14	-8.38	-7.40	-552	36	-560	41	**-0.3%**	**-0.5%**	**-0.1%**
ibm2014uP_14	75K	260	-133.37	-133.37	-31798	1556	-52800	3186	**-2.7%**	**0.0%**	**0.0%**
Average						**2101**		**7099**	**-7.2%**	**-1.0%**	**-1.7%**
Total						**29412**		**99392**	**-10.8%**	**-1.4%**	**-1.5%**

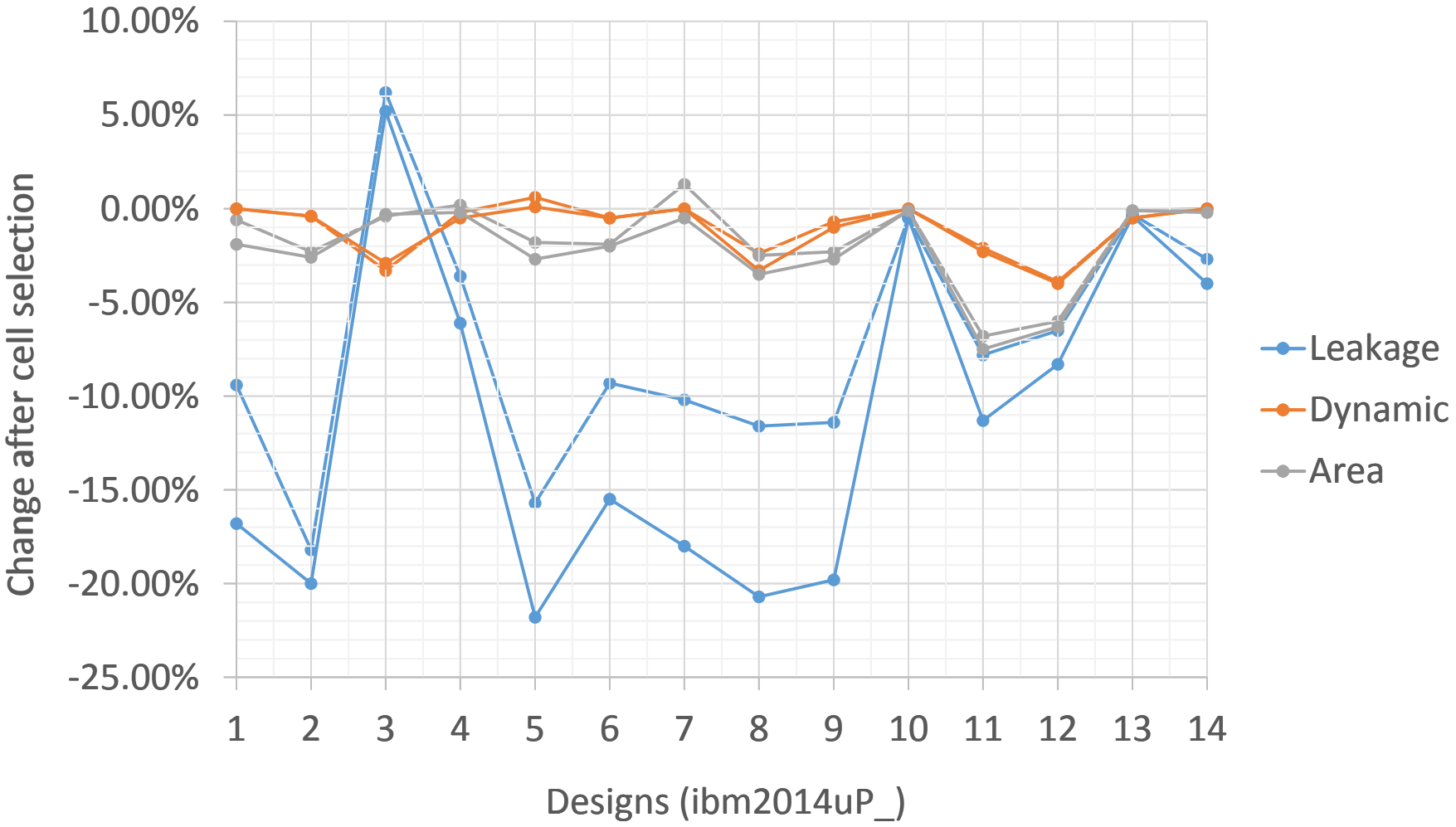
Results


10.00%
5.00%
0.00%
-5.00%
-10.00%
-15.00%
-20.00%
-25.00%
Change after cell selection
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Designs (ibm2014uP_)
Leakage
Dynamic
Area

Results

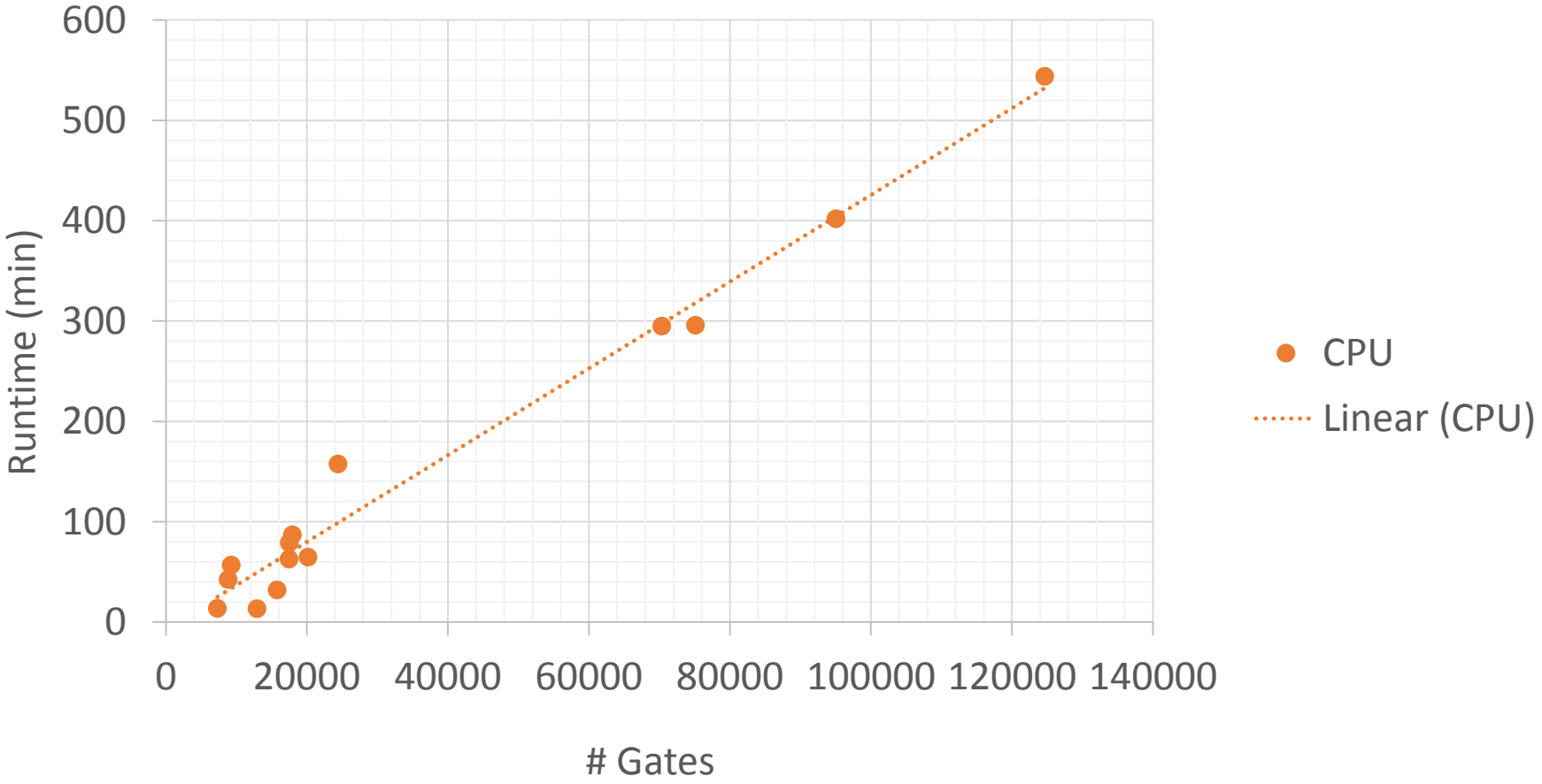

10.00%
5.00%
0.00%
-5.00%
-10.00%
-15.00%
-20.00%
-25.00%
Change after cell selection
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Designs (ibm2014uP_)
Leakage
Dynamic
Area

Results


10.00%
5.00%
0.00%
-5.00%
-10.00%
-15.00%
-20.00%
-25.00%
Change after cell selection
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Designs (ibm2014uP_)
Leakage
Dynamic
Area

Results

Runtime



Conclusions

Lagrangian Relaxation can be adapted to handle a variety of cell selection problems

It is a very robust method even without optimality guarantee

Warm start significantly reduces the number of LR iterations

The proposed methodologies effectively balance all objectives and respect all the imposed constraints

IC design flows still present considerable room for improvements in leakage power

Around 10% leakage power reduction on optimized designs

Cell Selection for High-Performance Designs in an Industrial Design Flow

THANK YOU