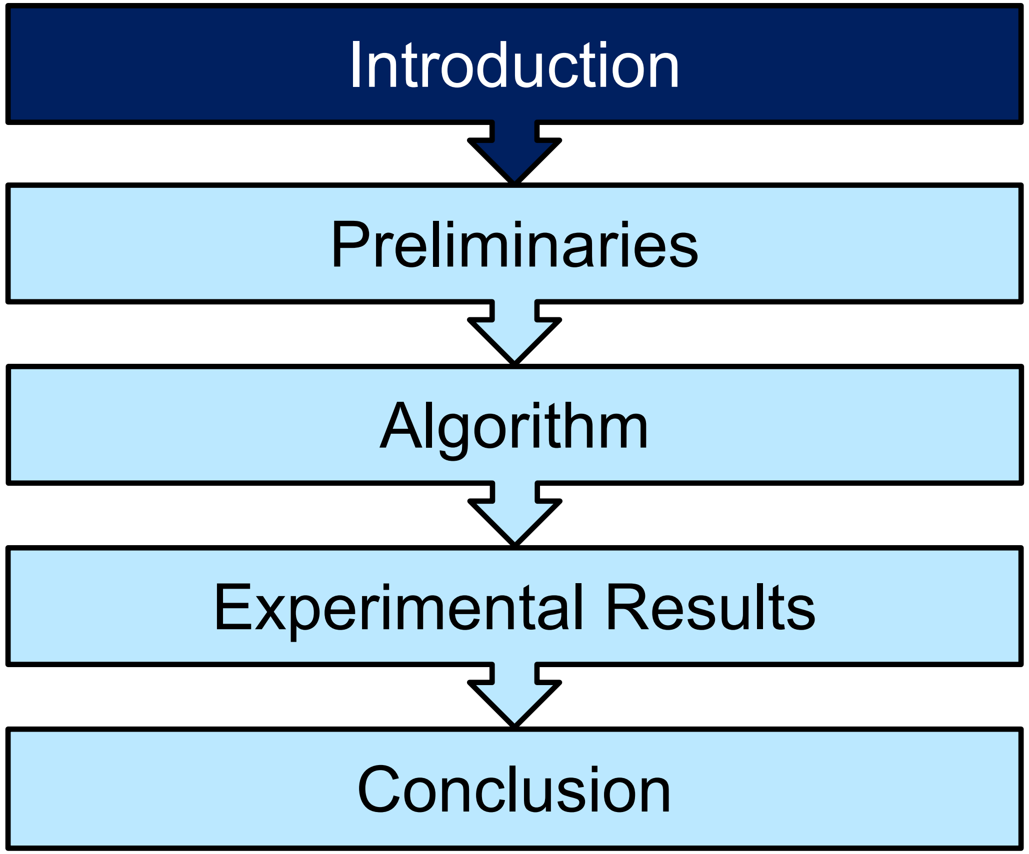
Slack Redistributed Register Clustering with Mixed-Driving Strength Multi-bit Flip-Flops

Yen-Yu Chen, Hao-Yu Wu, Iris Hui-Ru Jiang,
Cheng-Hong Tsai, Chien-Cheng Wu

National Taiwan University

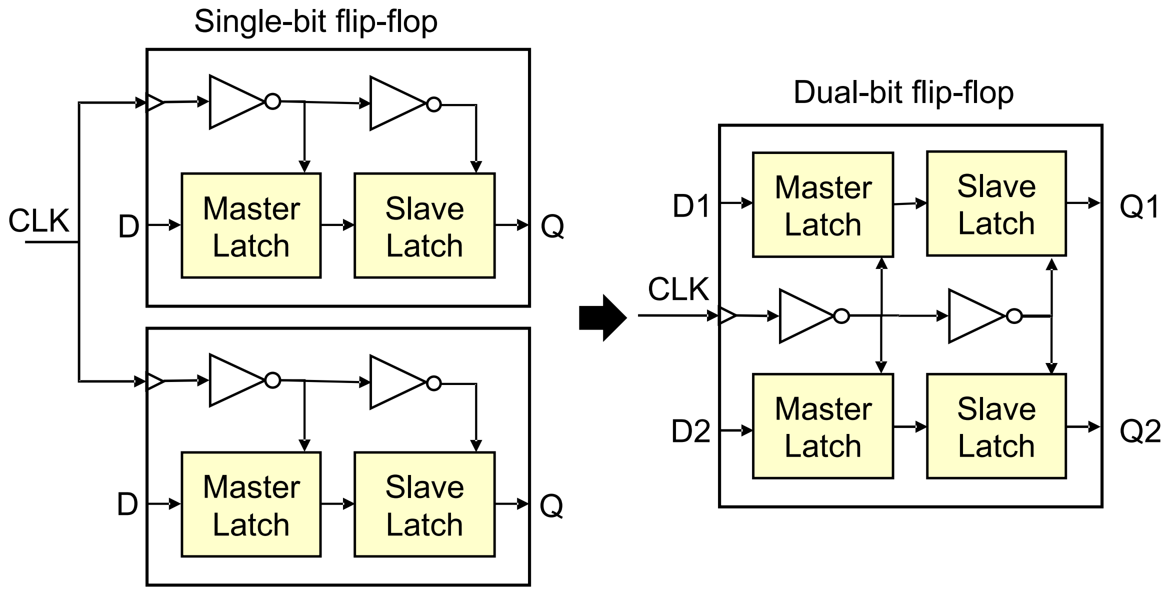
GUC

Outline


Introduction
Preliminaries
Algorithm
Experimental Results
Conclusion

Register Clustering

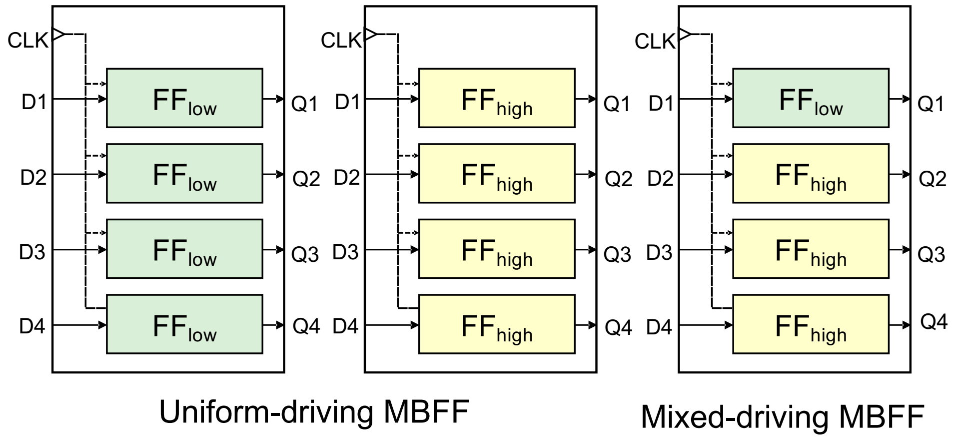
- Register clustering can reduce clock power.
  - Reduce the switching capacitance.
- Clock power dominates dynamic power.
- Timing and power trade-off.
  - Clustering FFs can reduce power, but displacement cause timing degradation



Mixed-Driving (Strength) MBFF

- FFs with high driving strength have faster signal propagation but more power consumption.
  - Driving strength: the current or voltage capability that a physical register can provide on its output signal.

- Liu *et al.* [1] introduces mixed-driving (strength) MBFF.
  - FFs in a MBFF can possess different driving strengths.

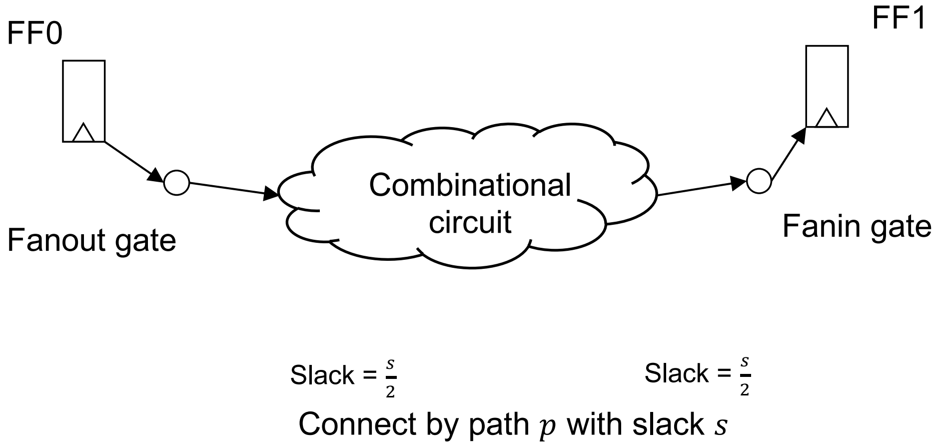


[1] M.-Y. Liu, Y.-C. Lai, W.-K. Mak, and T.-C. Wang, "Generation of mixed-driving multi-bit flip-flops for power optimization," in 2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9, 2022.

Slack Redistribution

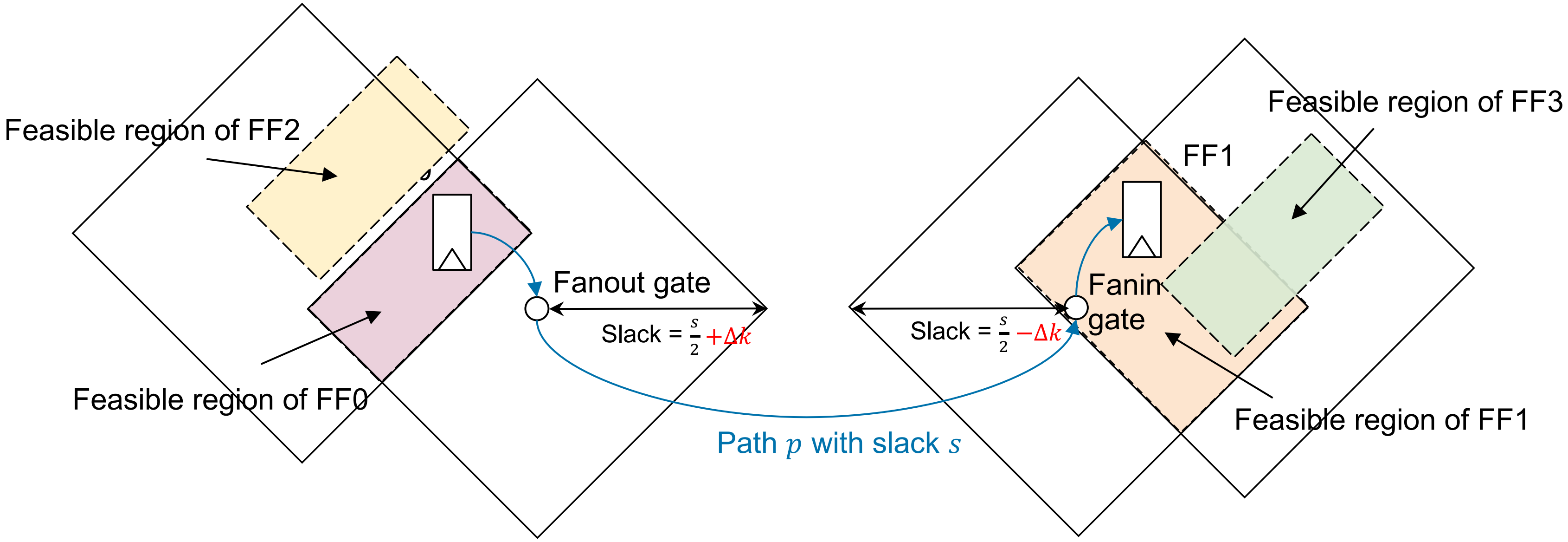
Motivation

- Typically, the available slack in a path is evenly distributed between the input and output FFs.



Slack Redistribution

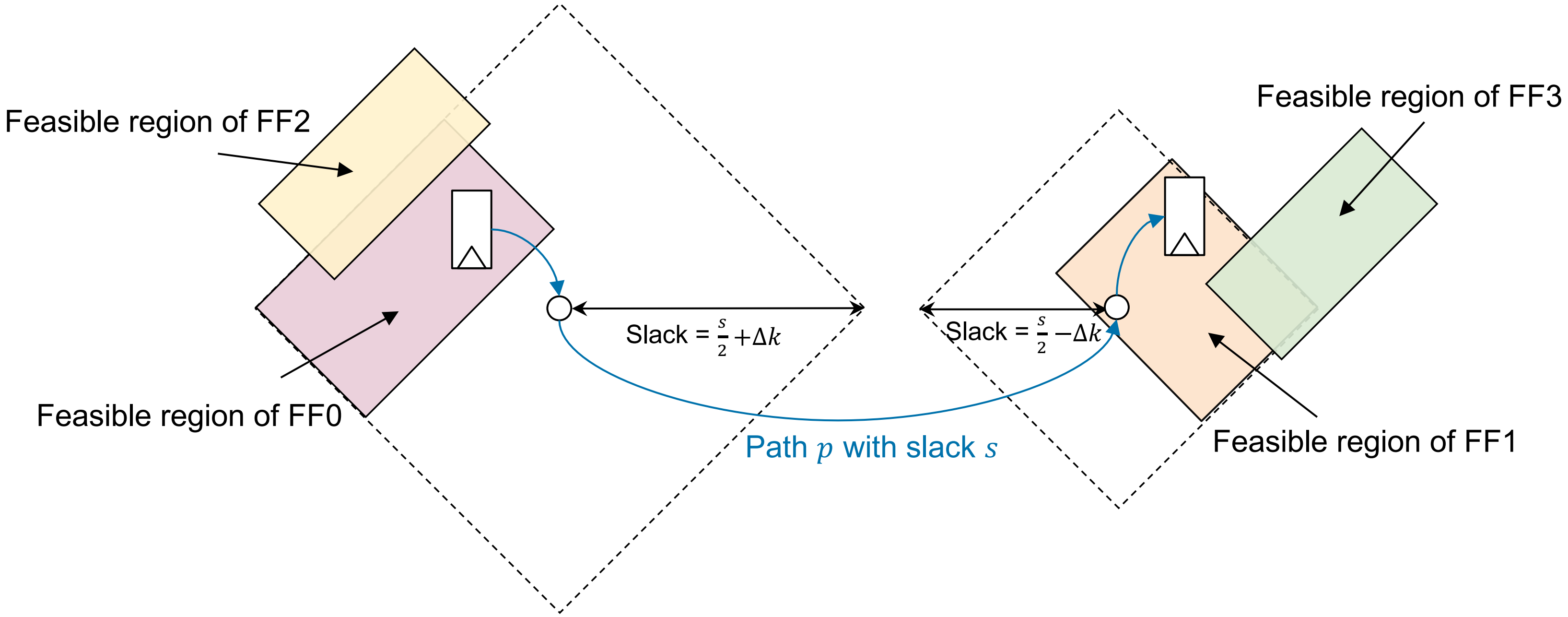
- Typically, the available slack in a path is evenly distributed between the input and output FFs.
  - FFs can be clustered if their feasible region overlap.



Slack Redistribution

Motivation

- Typically, the available slack in a path is evenly distributed between the input and output FFs.
  - FFs can be clustered if their feasible region overlap.
- However, FFs have different potentials for clustering.
- FFs can release their extra slack to other connected FFs.



Outline

- Introduction
- **Preliminaries**
- Algorithm
- Experimental Results
- Conclusion

Problem Formulation

- Given
  - Timing-driven placed design with single-bit FFs.
  - Mixed-driving strength MBFF library.
  - Static Timing Analysis results (paths with slack).

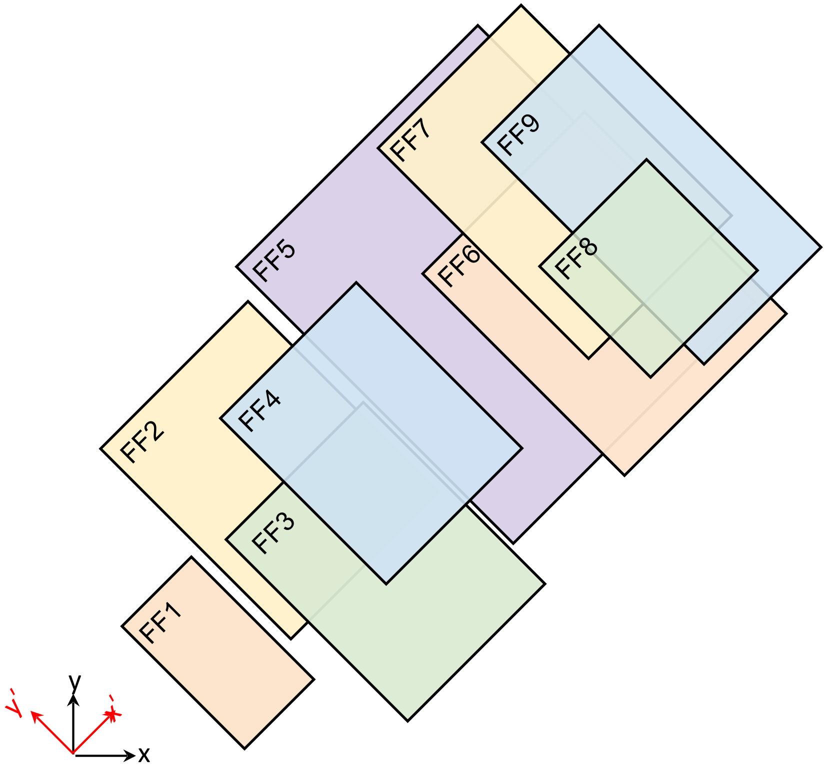
- Goal
  - Cluster FFs to MBFFs and determine their locations.

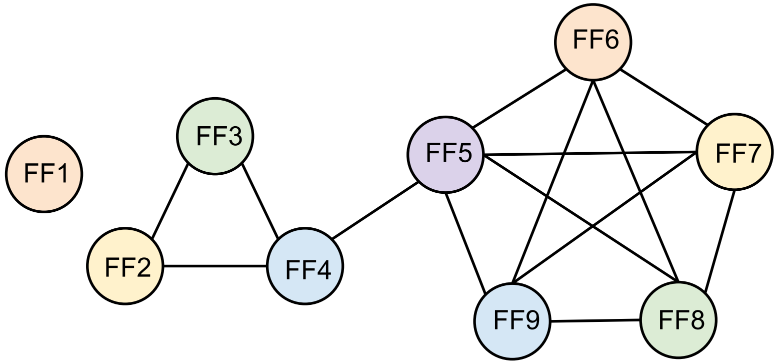
- Objectives
  - Minimize the total power consumption.
  - Minimize the worst negative slack (WNS).
  - Minimize the total negative slacks (TNS).

Intersection Graph

INTEGRA

- FFs whose feasible region overlap can be merged.
- The mergeable relationship can be captured by intersection graph

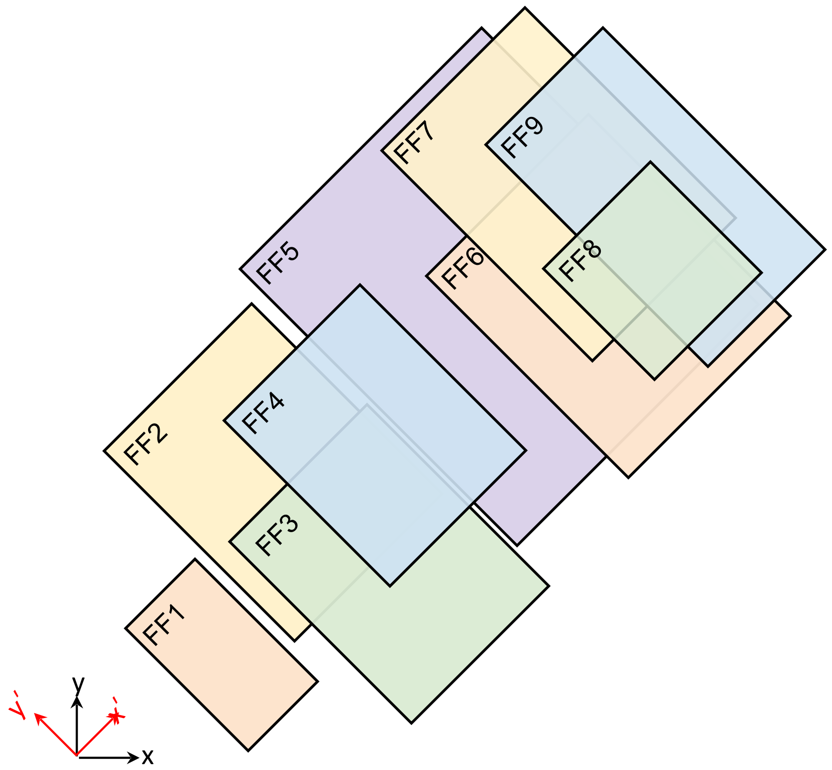


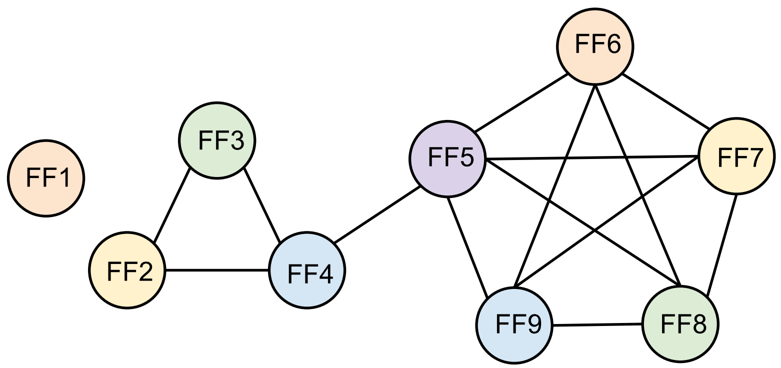


Coordinate Transformation

INTEGRA

- The feasible regions are rotated by 45° clockwise.
  - $\begin{cases} x' = y + x \\ y' = y - x \end{cases}$

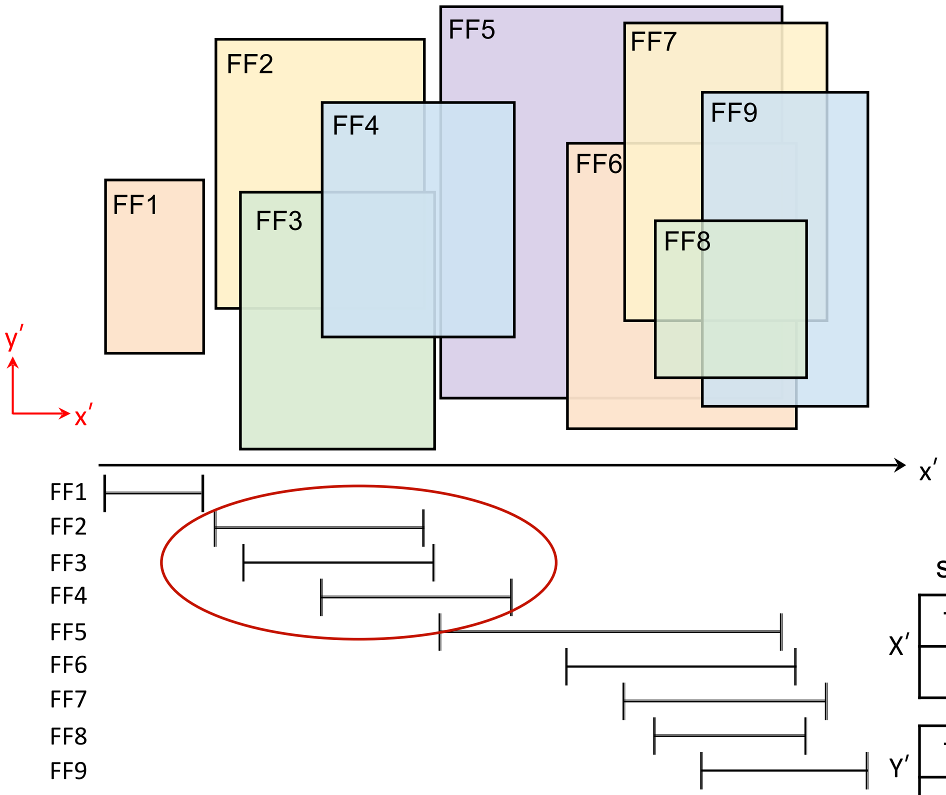


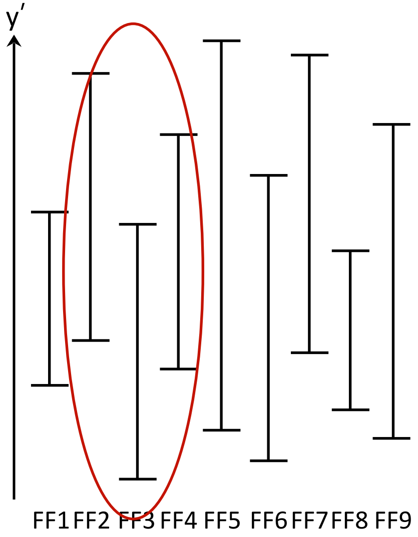


[2] I. H. -R. Jiang, C. -L. Chang and Y. -M. Yang, "INTEGRA: Fast Multibit Flip-Flop Clustering for Clock Power Saving," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 31, no. 2, pp. 192-204, Feb. 2012,

Interval Graph and Sequence

- The intersection graph is represented by two interval graphs.
  - The two interval graphs are encoded as two sequences, X' and Y'.





s: starting e: end

X'

Type	S	E	S	S	S	E	E	S	E	S	S	S	S	E	E	E	E	E
FF#	1	1	2	3	4	2	3	5	4	6	7	8	9	5	6	8	7	9

Y'

Type	S	S	S	S	S	S	S	S	S	E	E	E	E	E	E	E	E	E
FF#	3	6	9	5	8	1	4	7	2	8	3	1	6	4	9	2	7	5

MBFF Generation

INTEGRA

- Indentify a decision point, essential FFs and related FFs.
  - Single FFs are removed.

Remove single FF (FF1) | Related FFs (S 2, S 3, S 4) | Essential FFs (E 2, E 3)

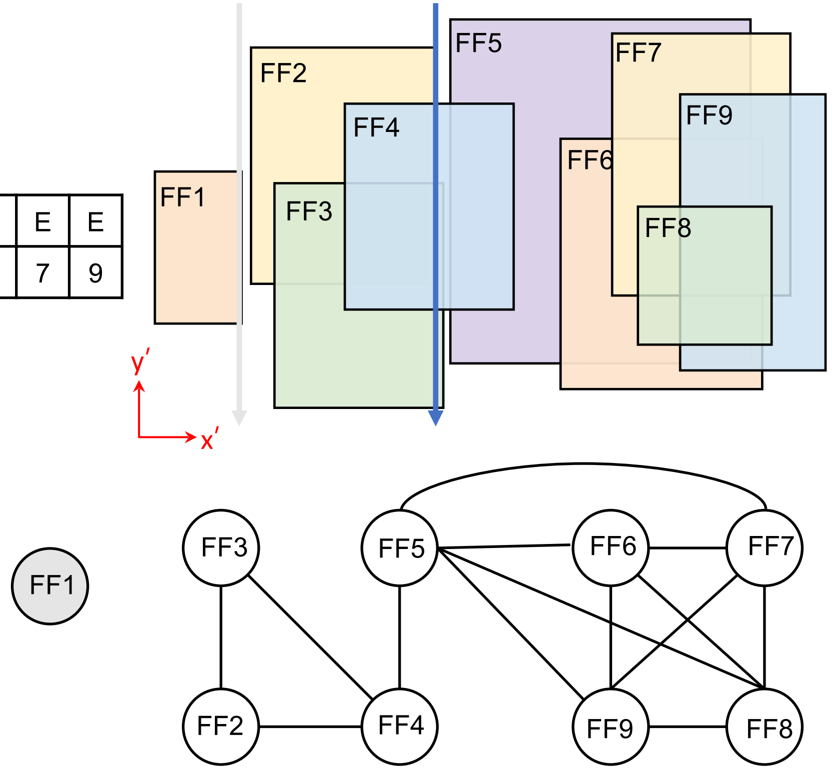
X'

Type	S	E	S	S	S	E	E	S	E	S	S	S	S	E	E	E	E	E
FF#	1	1	2	3	4	2	3	5	4	6	7	8	9	5	6	8	7	9

Decision point

Partial Y'

Type	S	S	S	E	E	E
FF#	3	4	2	3	4	2



MBFF Generation

INTEGRA

- Indentify a decision point, essential FFs and related FFs.
  - Single FFs are removed.
- Extract the maximal cliques from partial Y'.
- Partition the clique and generate MBFFs.
- Place MBFFs at density-free locations.

Remove single FF | Related FFs | Essential FFs

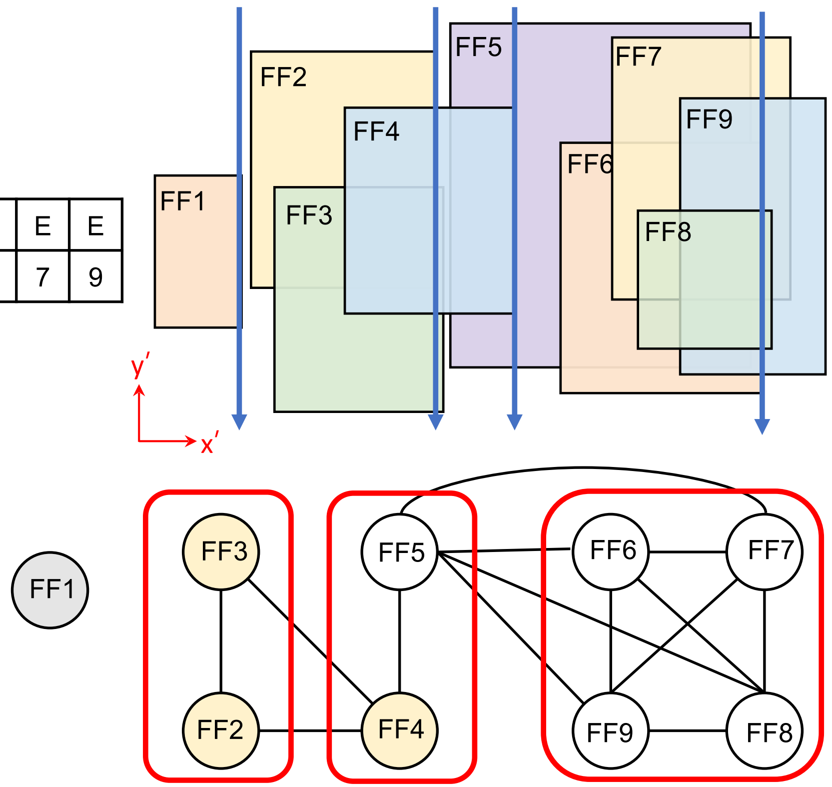
X'

Type	S	E	S	S	S	E	E	S	E	S	S	S	S	E	E	E	E	E
FF#	1	1	2	3	4	2	3	5	4	6	7	8	9	5	6	8	7	9

Decision point

Partial Y'

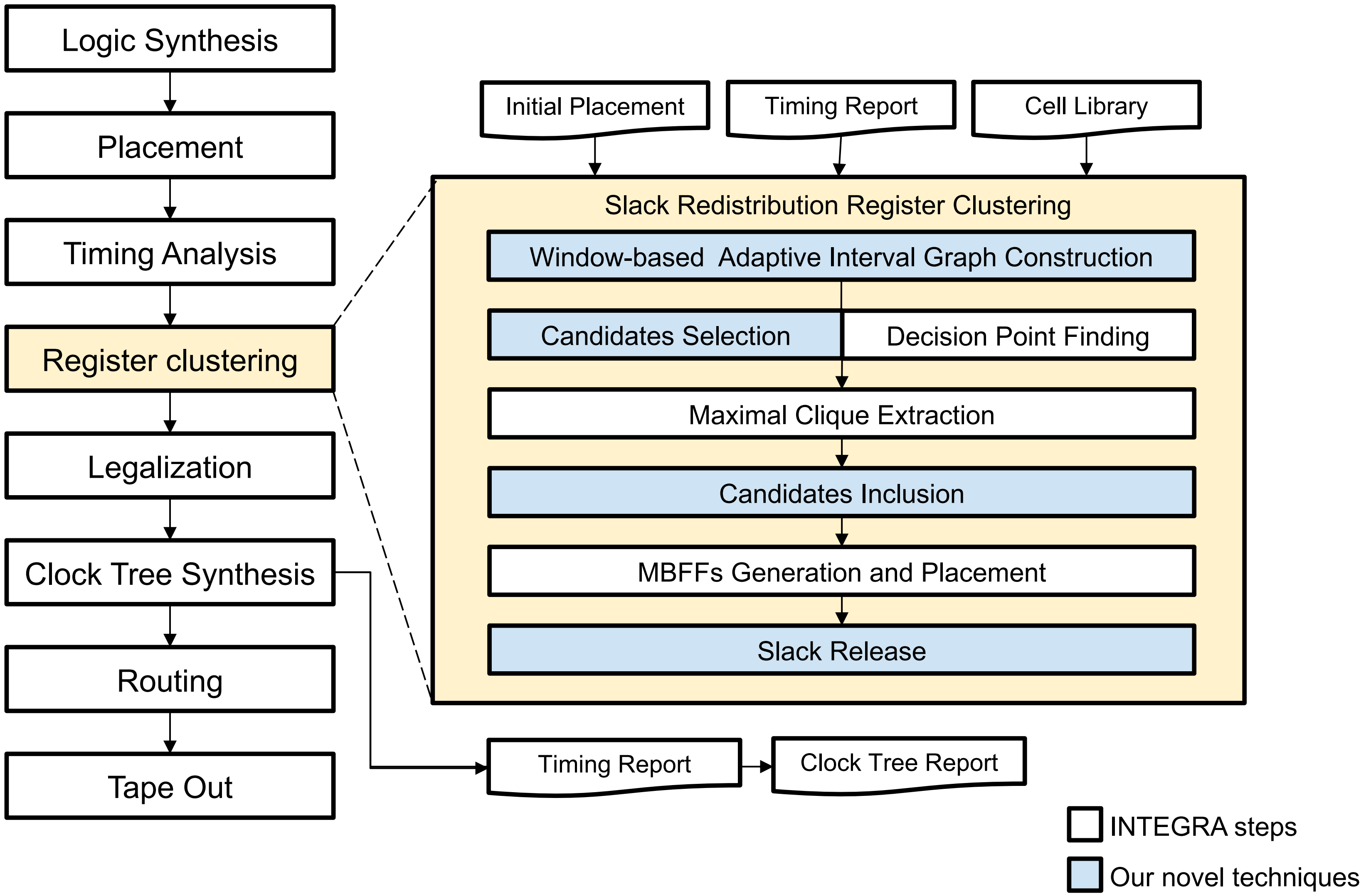
Type	S	S	S	S	E	E	E	E
FF#	6	7	8	9	6	8	7	9



Outline

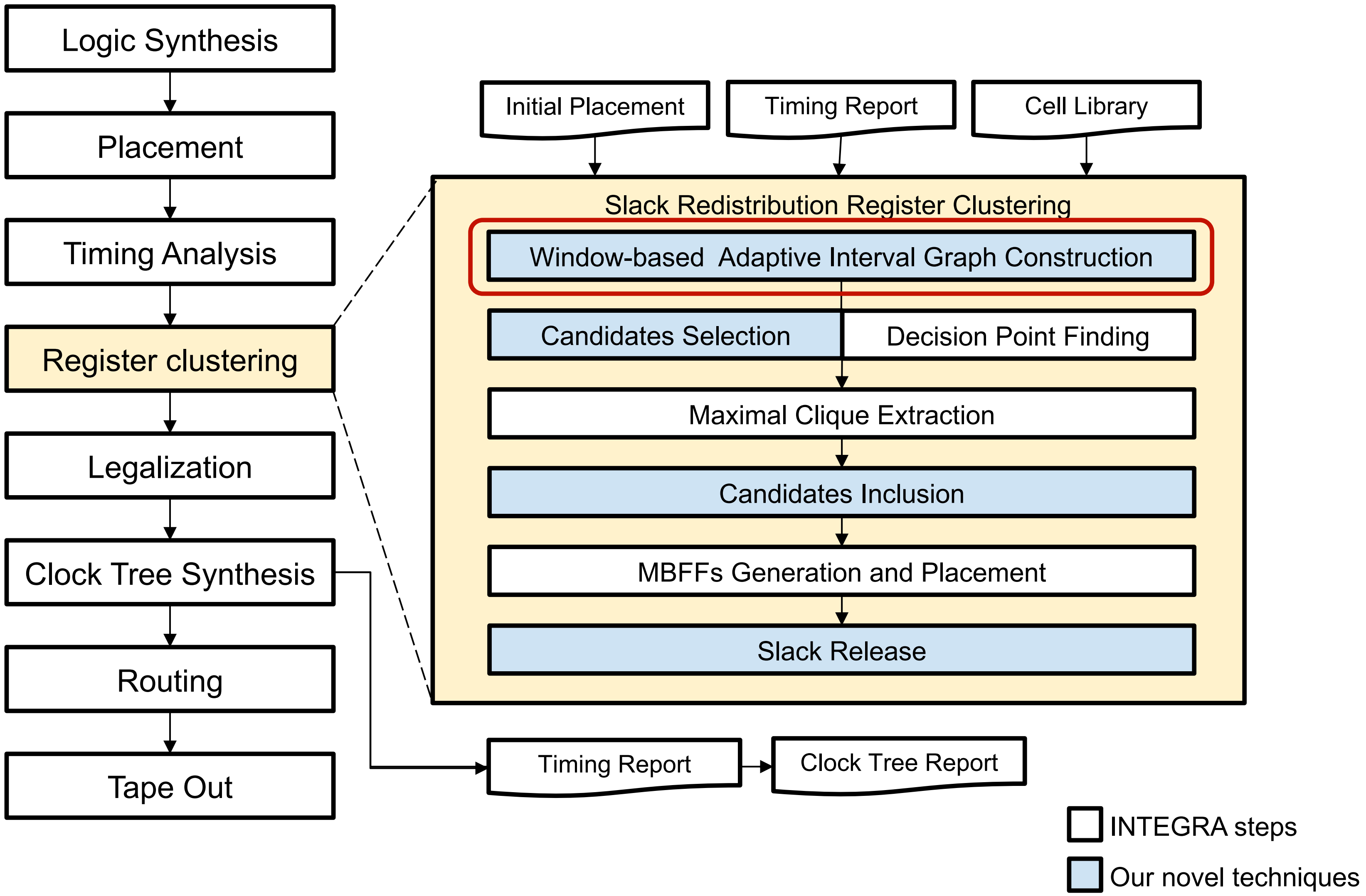
- Introduction
- Preliminaries
- **Algorithm**
- Experimental Results
- Conclusion

Framework



Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

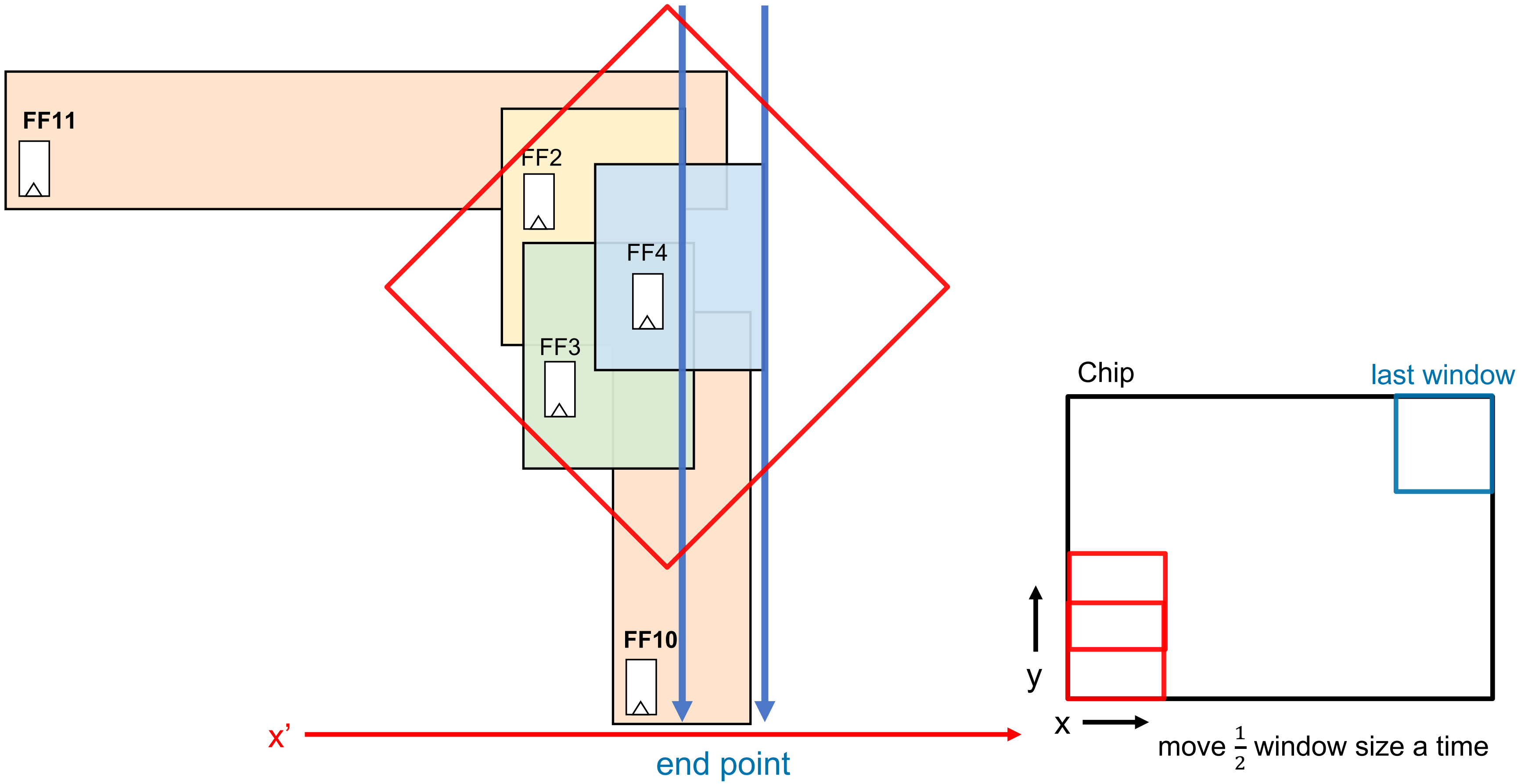
Framework



Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

Window-based X' Sequence Generation

- To ensure the small displacement of FFs.
  - Only the FFs in the inverstigated window are constructed in current X' sequence.



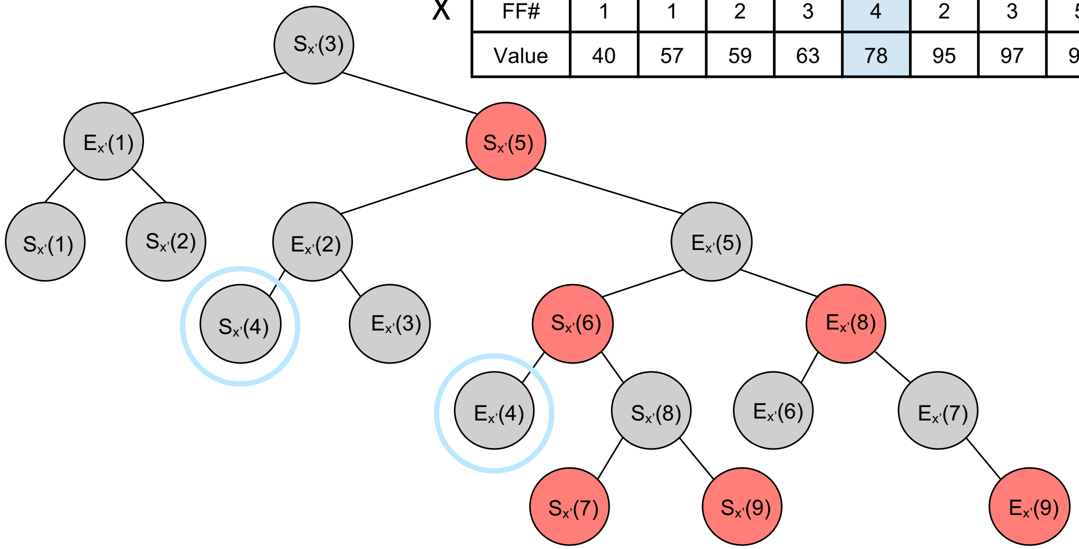
Adaptive Interval Graph Construction

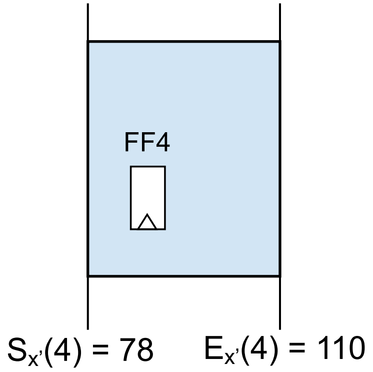
- Sequences should be adaptive to feasible region changes.
  - Maintain the X’ and partial Y’ sequence as red-black tree.

- Nodes in sequence should possess value information.
  - Value: the coordinate in corresponding interval graph.

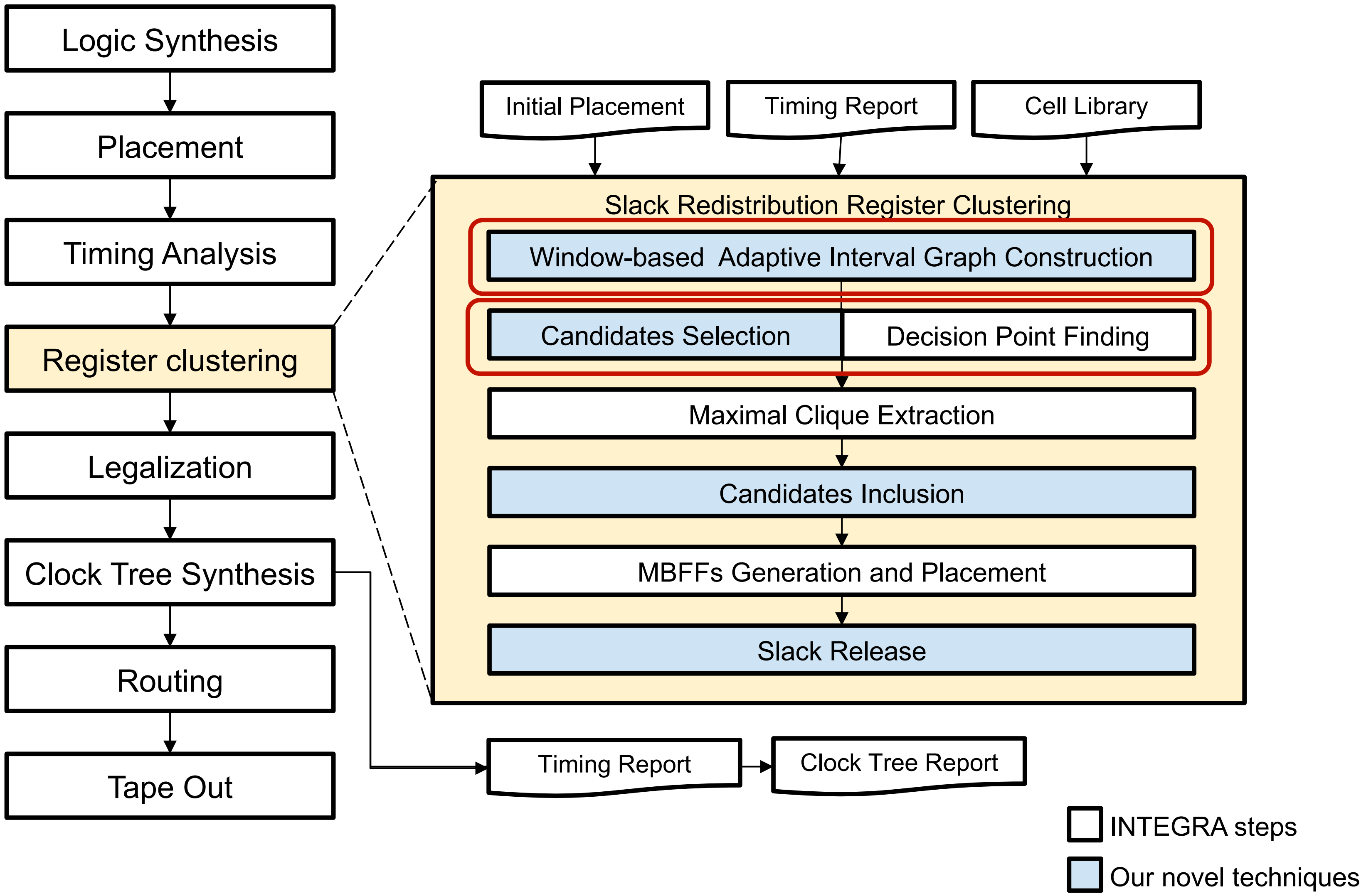
X′

Type	S	E	S	S	S	E	E	S	E	S	S	S	S	E	E	E	E	E
FF#	1	1	2	3	4	2	3	5	4	6	7	8	9	5	6	8	7	9
Value	40	57	59	63	78	95	97	98	110	120	130	135	143	157	159	161	165	172





Framework



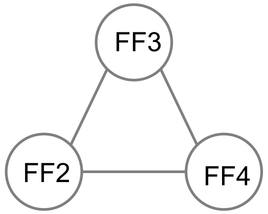
Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

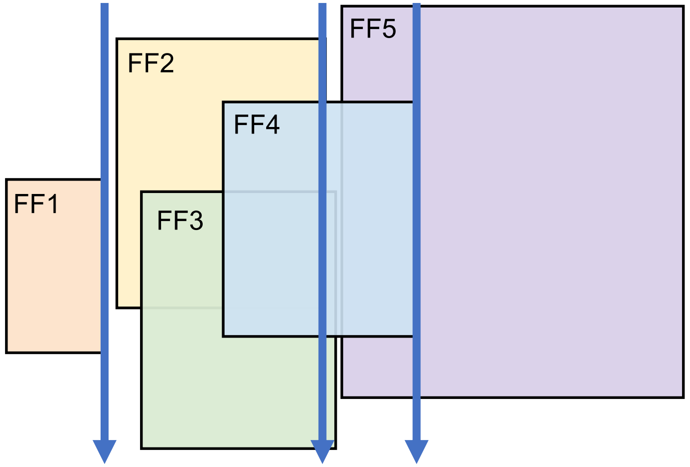
Candidates Selection

- To identify candidates with a high potential to cluster with related FFs.
  - Candidates are the FFs between the previous decision point and the next decision point but not related FFs.
  - Isolated flip-flops and visited but unclustered flip-flops are remained in X'.

Partial Y'

Type	S	S	S	E	E	E
FF#	3	4	2	3	4	2





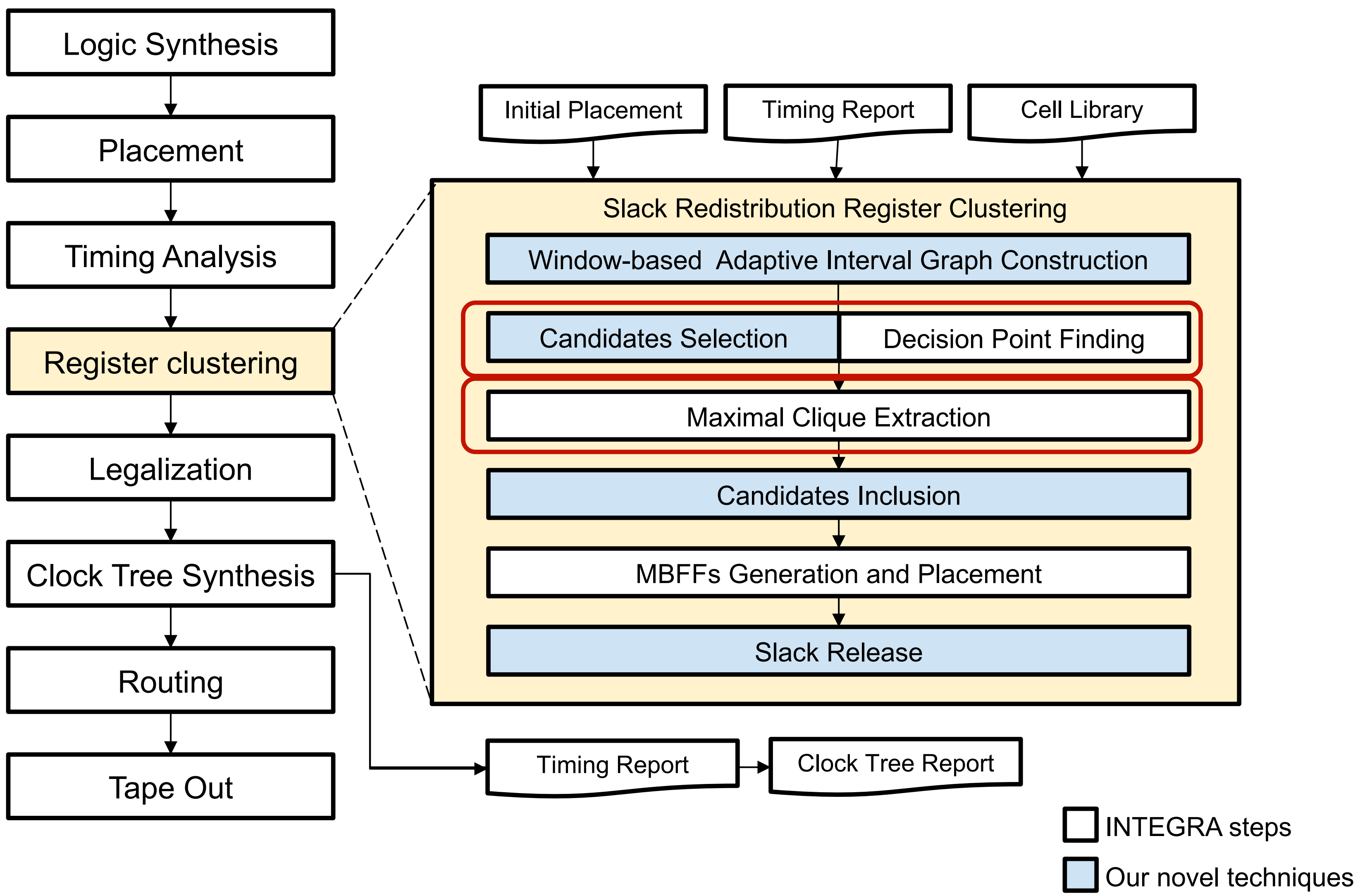
Candidate (FF 1) | Related FFs (FF 2, 3, 4) | Essential FF (FF 2) | Candidate (FF 5)

X'

Type	S	E	S	S	S	E	E	S	E	S	S	S	S	E	E	E	E	E
FF#	1	1	2	3	4	2	3	5	4	6	7	8	9	5	6	8	7	9
Value	40	57	59	63	78	95	97	98	110	120	130	135	143	157	159	161	165	172

Previous | Current | Next | Decision Point

Framework


Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

Candidates Inclusion

- Try to form a maximal clique of perfect size by including candidate FFs.
  - The perfect size of a maximal clique means that it can directly form an MBFF
  - For a clique with imperfect sizes, expand the clique until it reaches the perfect size.

Pseudo power library

Size	Normalized Power Per Bit	Normalized Area Per Bit	Driving Strength	Power(μW)	Area(μm^2)
1	1.00	1.00	Low	100	16.245
			High	170	17.382
2	0.70	0.96	Low	140	31.190
			High	238	33.374
4	0.60	0.72	Low	240	46.786
			High	408	50.061
8	0.50	0.65	Low	400	83.174
			High	680	88.997

Perfect size

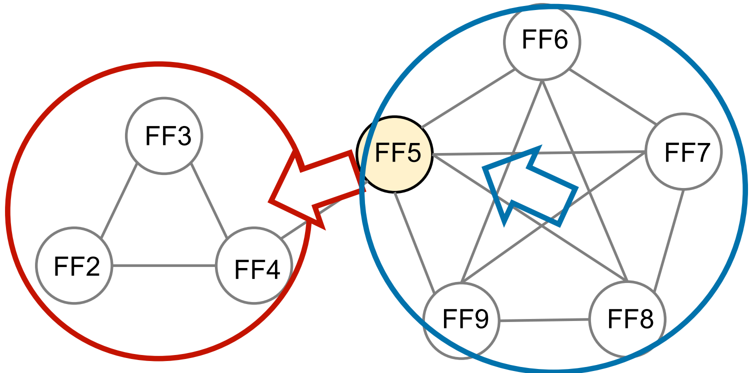
Inclusion Force

- The inclusion force exists between a clique and a FF, which reflects the force between them.
  - Focus on power reduction since the feasible region has already taken timing into account.

- A larger inclusion force means include the FF requires less slack borrowing and has a smaller impact on the original maximal clique to which the FF belongs.

- The inclusion force consists of an attractive factor and a repulsive factor.

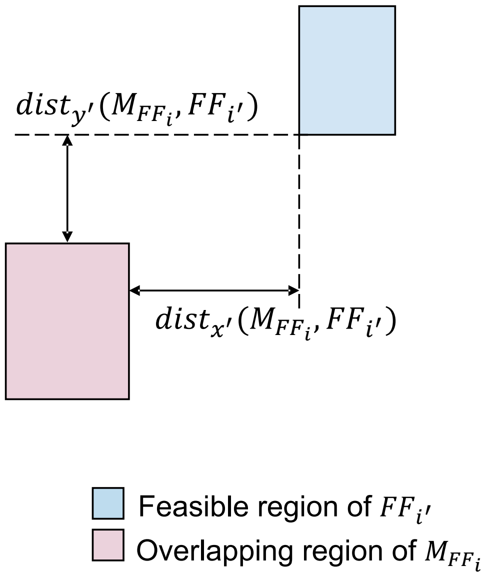
$$F_{inclusion} = F_{att} \times F_{rep}$$

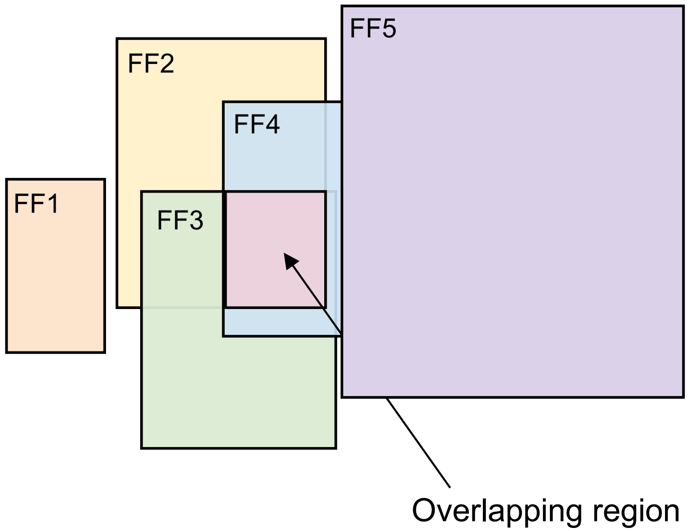


Attractive Factor

- Considering the proximity of a candidate FF's FF_j feasible region to the overlapping region of the maximal clique M_{FF_i}.

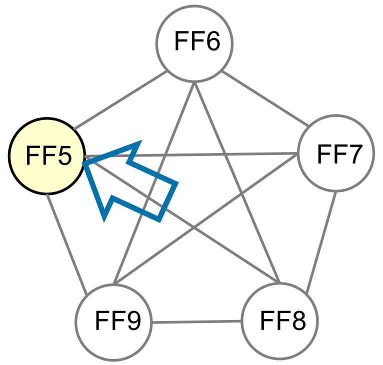
$$F_{\text{att}}\left(M_{FF_i}, FF_j\right) = \frac{1}{\max(\text{dist}_{x'}(M_{FF_i}, FF_j), \text{dist}_{y'}(M_{FF_i}, FF_j)\ )}$$





Repulsive Factor

- Influenced by the size of the maximal clique to which the candidate FF belongs and the degree of power saving.
  - $\overline{S_{M_{FF_j}}}$: closet but "not" larger than M_{FF_j}'s perfect size.
  - $\underline{S_{M_{FF_j}}}$: closet but smaller than M_{FF_j}'s perfect size.



- $F_{\text{rep}}(M_{FF_i}, FF_j)$

$$= \begin{cases} \dfrac{\overline{S_{M_{FF_j}}} - S_{M_{FF_j}}}{S_{M_{FF_j}} - \underline{S_{M_{FF_j}}}} \times \underline{S_{M_{FF_j}}},\ if\ S_{\max} \geq S_{M_{FF_j}} \geq S_{\min} \\ (S_{\max} - S_{\min}) \times \left(S_{M_{FF_j}} - S_{\max}\right) \times S_{\max},\ if\ S_{M_{FF_j}} > S_{\max} \\ (S_{\max} - S_{\min}) \times \left(S_{\min} - S_{M_{FF_j}}\right) \times S_{\max},\ if\ S_{M_{FF_j}} < S_{\min} \end{cases}$$

Repulsive Factor

- $F_{\text{rep}}(M_{FF_i}, FF_j)$

$$
= \begin{cases}
\dfrac{\overline{S_{MFF_j}} - S_{MFF_j}}{S_{MFF_j} - \underline{S_{MFF_j}}} \times \underline{S_{MFF_j}}, \; if \; S_{\max} \geq S_{MFF_j} \geq S_{\min} \\
(S_{\max} - S_{\min}) \times \left(S_{MFF_j} - S_{\max} \right) \times S_{\max}, \; if \; S_{MFF_j} > S_{\max} \\
(S_{\max} - S_{\min}) \times \left(S_{\min} - S_{MFF_j} \right) \times S_{\max}, \; if \; S_{MFF_j} < S_{\min}
\end{cases}
$$

Pseudo power library

Size	Normalized Power Per Bit	Normalized Area Per Bit	Driving Strength	Power(μW)	Area(μm^2)
1	1.00	1.00	Low	100	16.245
			High	170	17.382
2	0.70	0.96	Low	140	31.190
			High	238	33.374
4	0.60	0.72	Low	240	46.786
			High	408	50.061
8	0.50	0.65	Low	400	83.174
			High	680	88.997

A clique with n FFs

Minimum perfect size ← 2

5 6 7

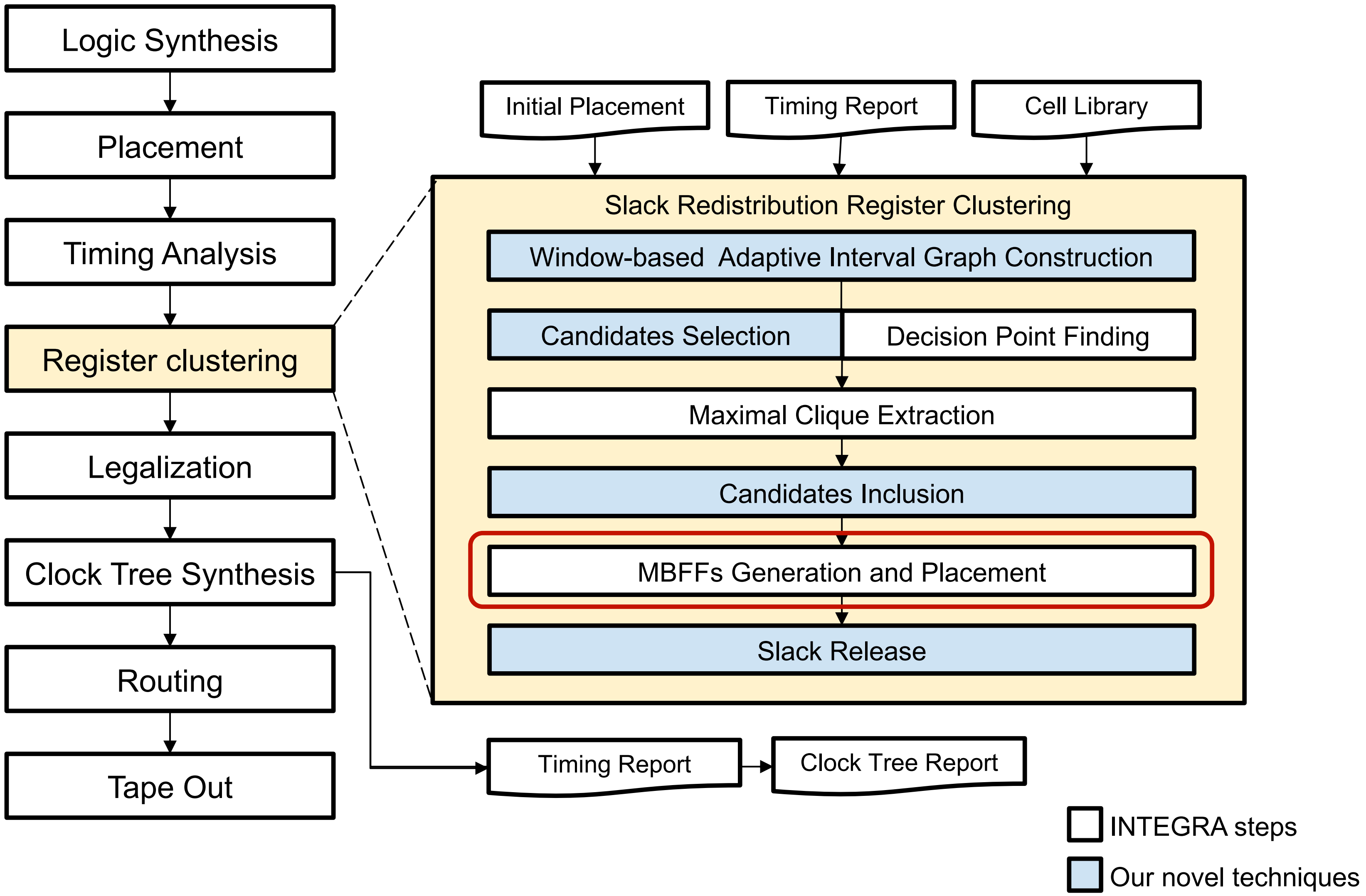
Maximum perfect size ← 8

Perfect size

Candidates Inclusion Process

- First, generate the maximal feasible region for each candidates.
  - Remove the candidates who cannot include even borrowing all slack from the connected paths simultaneously.
    - Find maximal clique and sort those FFs waste execution time.
- Calculate the inclusion force for each candidate FF.
- Sort candidate FFs by inclusion force and try to include FFs by this order.
  - After successfully include one FFs, the overlapping region need to be updated.
- Finish when reach a maximal clique of perfect size.

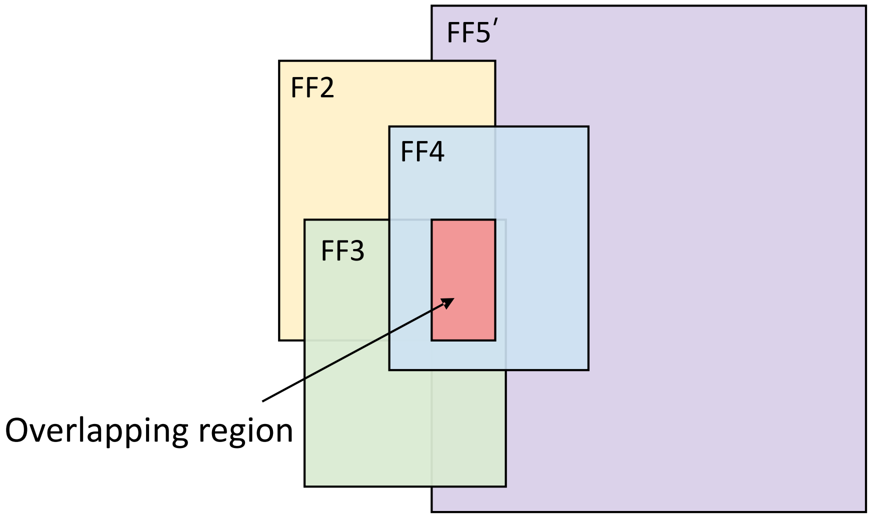
Framework

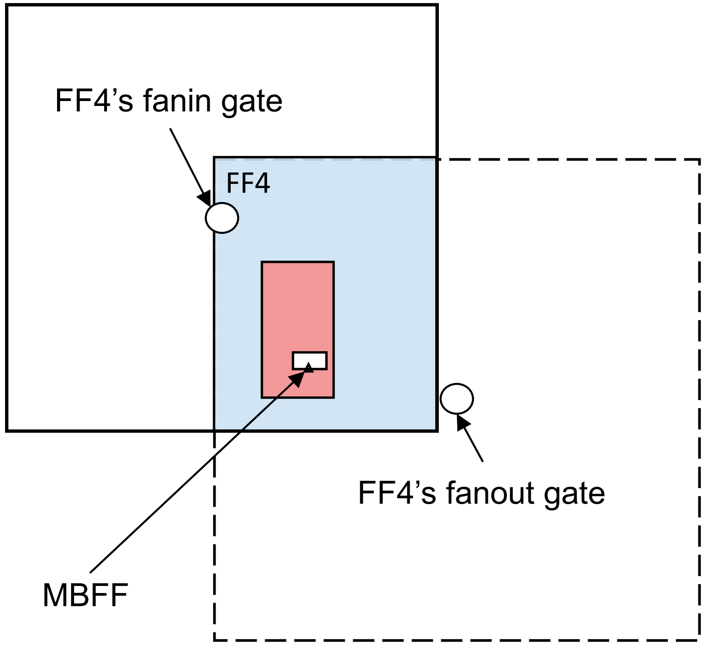


Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

Slack Release

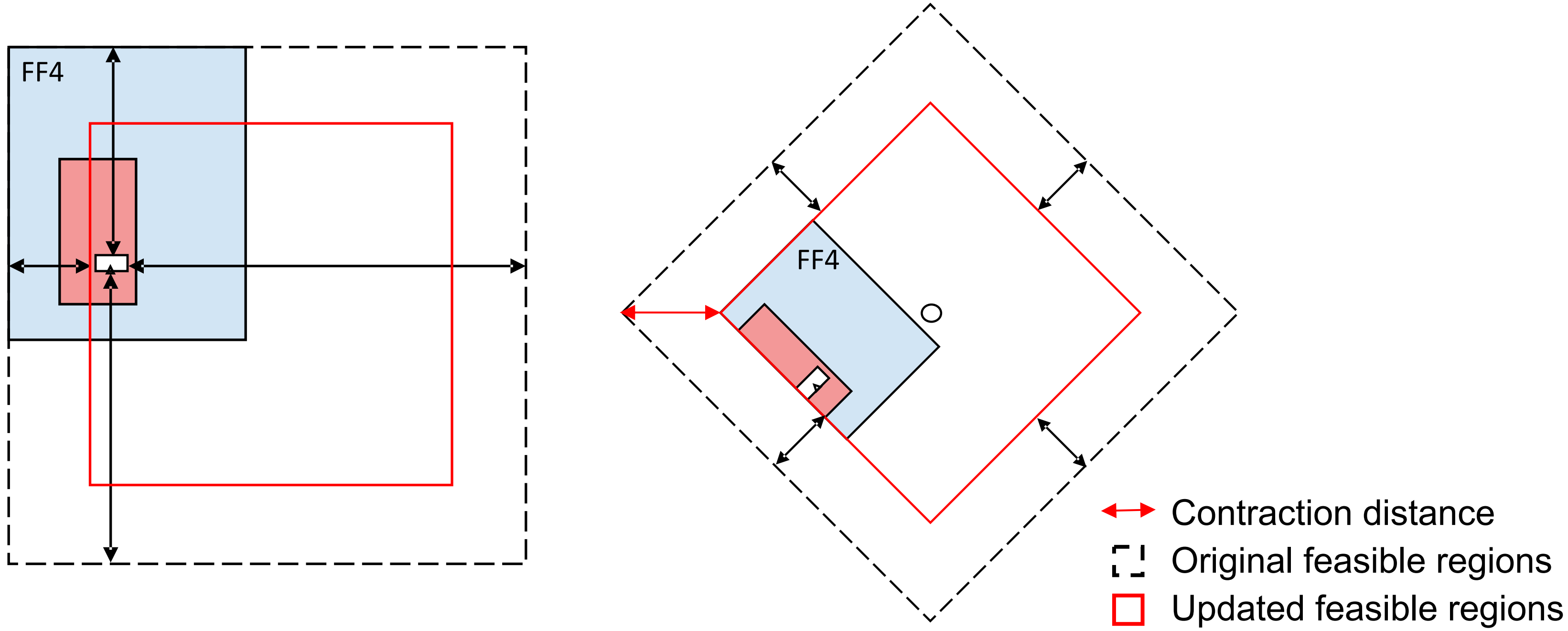
- The clustered FFs can release their extra slack to the connected and unclustered FFs.
  - Recall that a feasible region is constructed based on several diamond (square) regions.
  - The regions are contracted to just enclose the location of the formed MBFF.





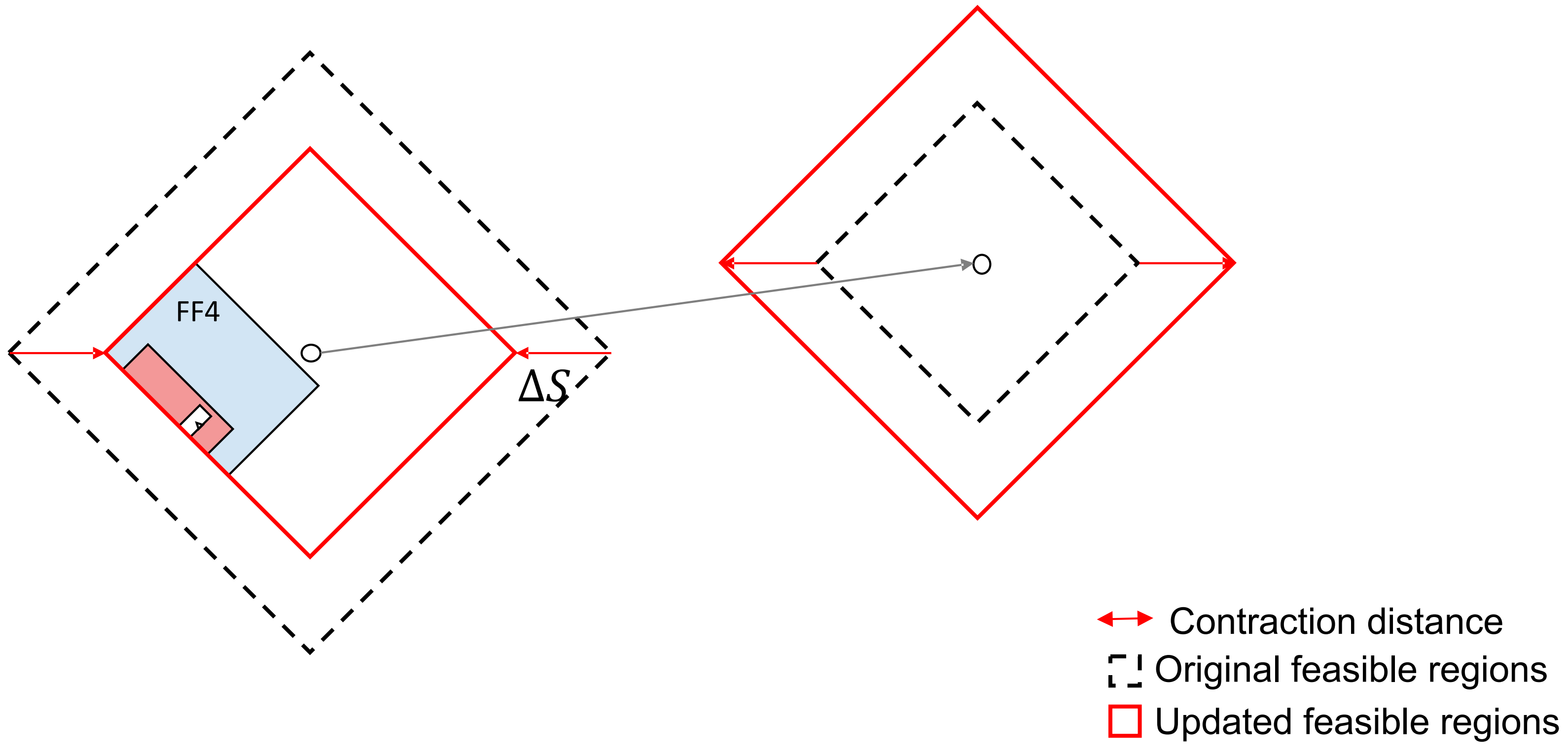
Slack Release

- The regions are contracted to just enclose the location of the formed MBFF.
  - That is, finding the minimum distance between the four sides of the square region and the location of the MBFF
  - Each diamond region should retain its diamond shape and center after contraction and expansion.

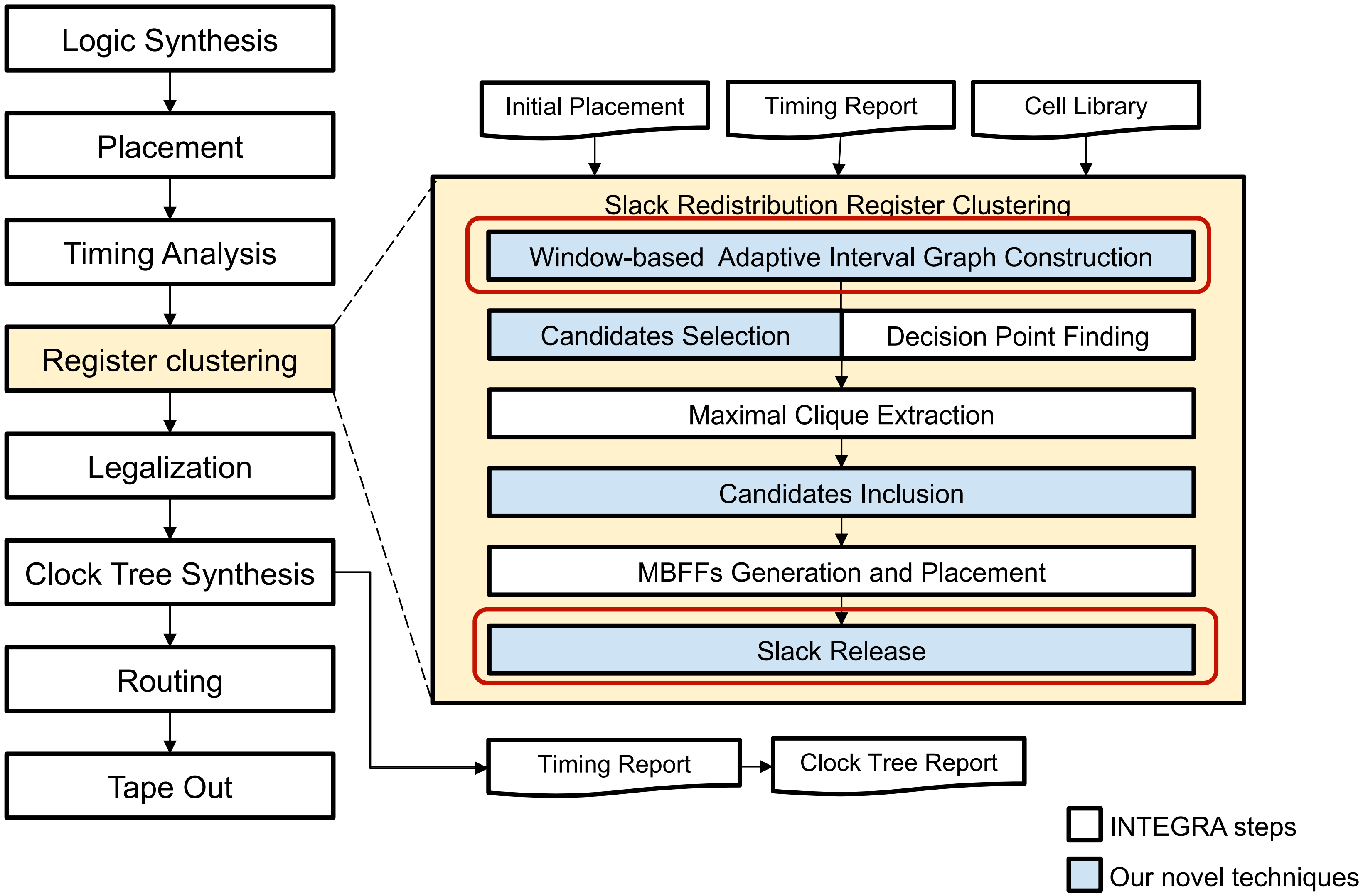


Slack Release

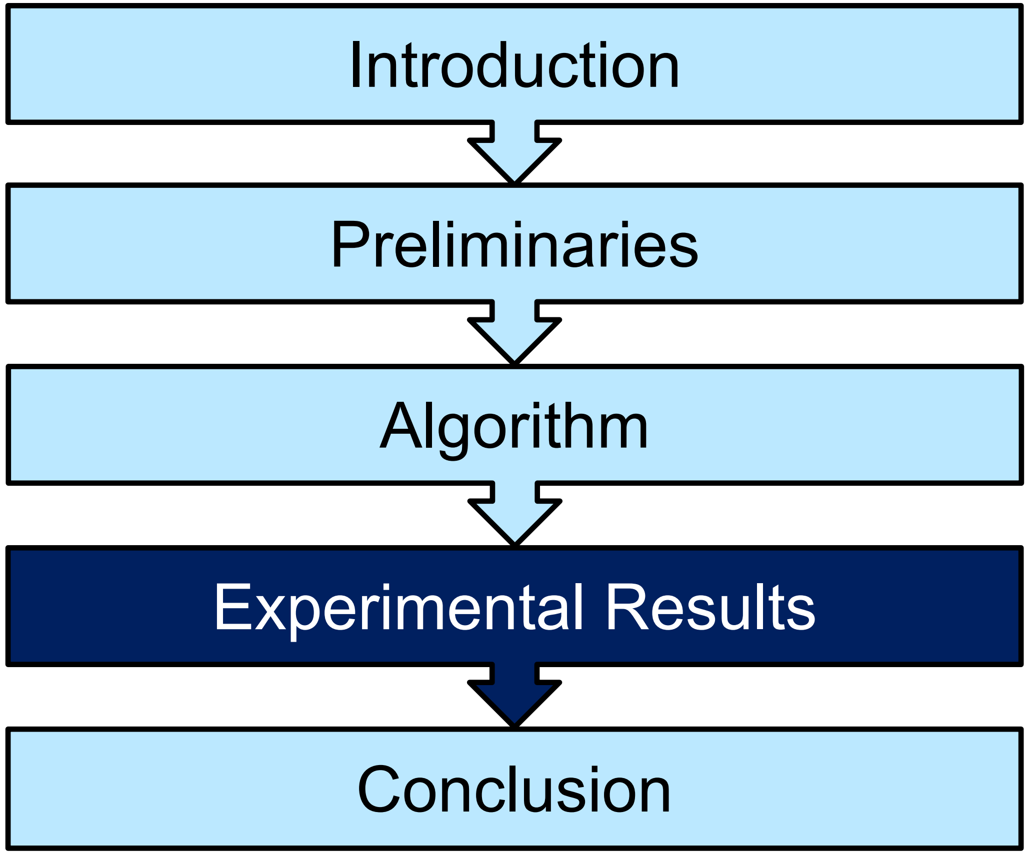
- The distance is converted back to slack then distributed to the connected and unclustered FFs.



Framework


Logic Synthesis
Placement
Timing Analysis
Register clustering
Legalization
Clock Tree Synthesis
Routing
Tape Out
Initial Placement
Timing Report
Cell Library
Slack Redistribution Register Clustering
Window-based Adaptive Interval Graph Construction
Candidates Selection
Decision Point Finding
Maximal Clique Extraction
Candidates Inclusion
MBFFs Generation and Placement
Slack Release
Timing Report
Clock Tree Report
INTEGRA steps
Our novel techniques

Outline


Introduction
Preliminaries
Algorithm
Experimental Results
Conclusion

Experimental Settings

- Language
  - C++
- Platform
  - An Intel Xeon E5-2650 v2 2.6 GHz CPU with 192GB memory
- Benchmark
  - ICCAD 2015 Incremental Timing-driven Placement benchmark suite.
  - Half FFs that are more timing-critical are changed to high-driving strength.
- Evaluation of timing and clock wirelength/buffers
  - Cadence Innovus
- Evaluation of power
  - Pseudo clock sink power and area table provided by [1]

[1] M.-Y. Liu, Y.-C. Lai, W.-K. Mak, and T.-C. Wang, "Generation of mixed-driving multi-bit flip-flops for power optimization," in 2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9, 2022.

Benchmark & Power Table

- Timing-driven placed designs
  - ICCAD 2015 Incremental Timing-driven Placement benchmark suite.

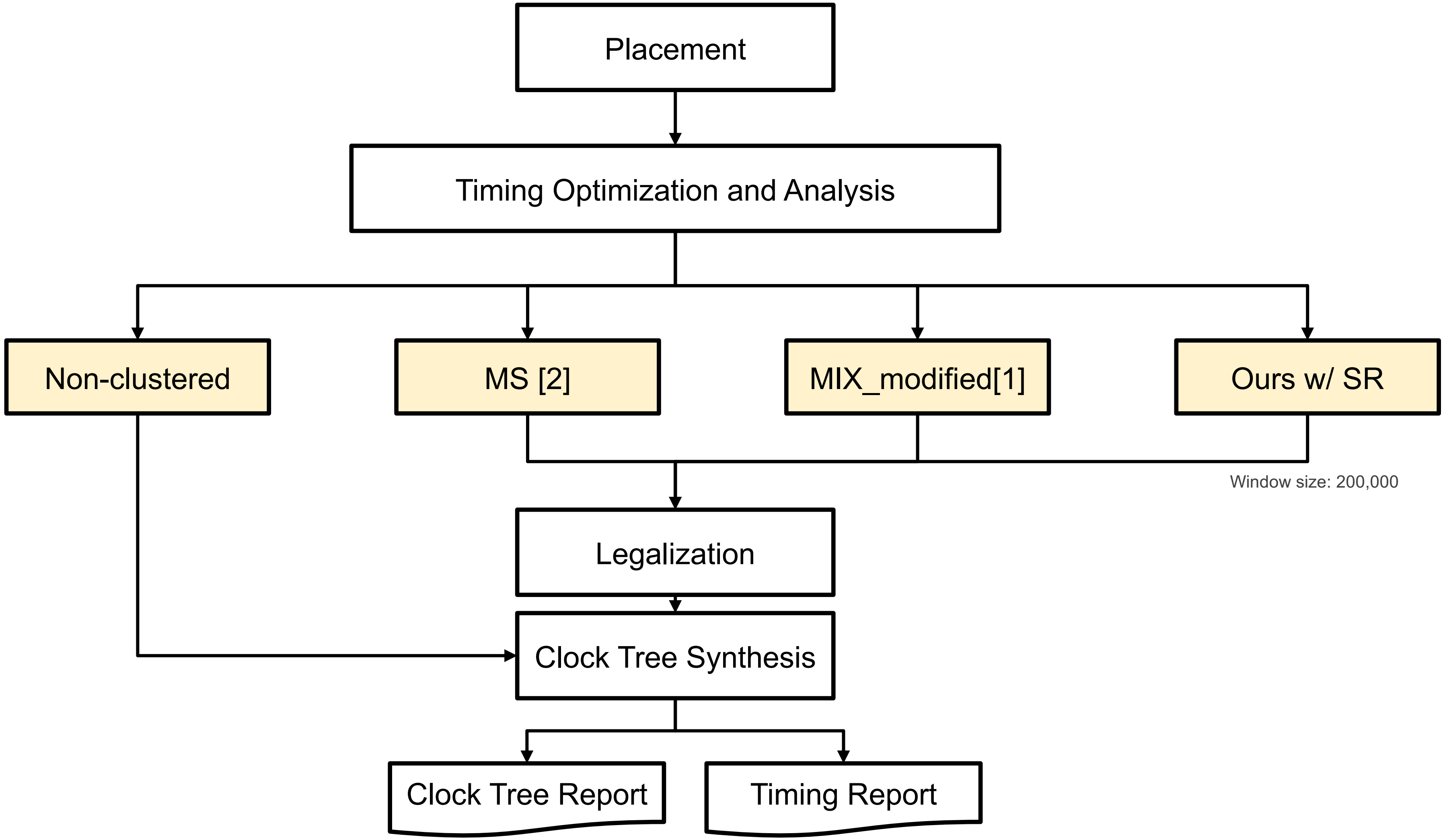
Circuit	# of Cells	# of FFs	Clock Period (ps)
Superblue1	1209716	137560	9500
Superblue3	1213253	163107	10000
Superblue4	795645	167731	6000
Superblue5	1086888	110941	9000
Superblue7	1931639	262176	5500
Superblue10	1876103	231747	10000
Superblue16	981559	142543	5500
Superblue18	768068	101758	7000

- Pseudo power library [1]

Size	Normalized Power Per Bit	Normalized Area Per Bit	Driving Strength	Power(μW)	Area(μm^2)
1	1.00	1.00	Low	100	16.245
			High	170	17.382
2	0.70	0.96	Low	140	31.190
			High	238	33.374
4	0.60	0.72	Low	240	46.786
			High	408	50.061
8	0.50	0.65	Low	400	83.174
			High	680	88.997

[1] M.-Y. Liu, Y.-C. Lai, W.-K. Mak, and T.-C. Wang, "Generation of mixed-driving multi-bit flip-flops for power optimization," in 2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9, 2022.

Experimental Flow



[2] Y.-C. Chang, T.-W. Lin, I. H.-R. Jiang, and G.-J. Nam, "Graceful register clustering by effective mean shift algorithm for power and timing balancing," in Proceedings of the 2019 International Symposium on Physical Design, pp.11–18, 2019.

[1] M.-Y. Liu, Y.-C. Lai, W.-K. Mak, and T.-C. Wang, "Generation of mixed-driving multi-bit flip-flops for power optimization," in 2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9, 2022.

Performance Comparisons

- The best performance in terms of WNS, power and runtime.
  - Although there is a 0.79% TNS degradation compared to MS, we achieve a 1.81% improvement in power.

Table 4: Timing, Power, MBFF Size and Runtime Comparison.

Circuit	Method	WNS (ns)	WNS Difference	TNS (ns)	TNS Ratio	Routed WL (um)	Routed WL Ratio	Buffers No.	Buffers Ratio	Clock Sink FF Power Ratio	MBFF Size 8	MBFF Size 4	MBFF Size 2	MBFF Size 1	Runtime (s)
Superblue1	NC	-8.103	0.000	-1304.5	0.00%	1063091.97	100.00%	3408	100.00%	100.00%					
	MS	-8.230	-0.127	-1417.0	8.62%	585660.05	55.09%	1268	37.21%	54.99%	12192	5854	6276	4056	9.44
	MIX_modified	-8.358	-0.255	-1505.0	15.37%	667101.72	62.75%	1535	45.04%	65.05%	13540	2819	1670	26493	393.76
	Ours	-8.215	-0.112	-1410.7	8.14%	572681.23	53.87%	1231	36.12%	53.74%	13108	5468	5968	183	7.72
Superblue3	NC	-14.001	0.000	-2316.5	0.00%	1069539.04	100.00%	3580	100.00%	100.00%					
	MS	-14.218	-0.217	-2403.2	3.74%	598915.84	56.00%	1405	39.25%	55.25%	14002	7625	7977	4637	11.35
	MIX_modified	-14.627	-0.626	-2508.6	8.29%	716060.67	66.95%	1886	52.68%	67.16%	14759	3088	1812	40614	438.86
	Ours	-14.153	-0.152	-2375.4	2.54%	560480.58	52.40%	1292	36.09%	52.91%	16740	4900	5338	196	9.94
Superblue4	NC	-7.992	0.000	-7345.5	0.00%	1013895.91	100.00%	3733	100.00%	100.00%					
	MS	-8.089	-0.097	-7694.3	4.75%	574376.20	56.65%	1367	36.62%	55.00%	14795	7287	7705	4813	12.08
	MIX_modified	-14.773	-6.781	-7908.1	7.66%	645928.14	63.71%	1696	45.43%	64.56%	16649	3899	2315	27729	467.65
	Ours	-8.010	-0.018	-7332.9	-0.17%	542716.56	53.53%	1273	34.10%	52.89%	17204	5165	5289	142	6.37
Superblue5	NC	-14.695	0.000	-8167.1	0.00%	918192.48	100.00%	2973	100.00%	100.00%					
	MS	-15.211	-0.516	-8194.5	0.34%	553203.92	60.25%	1406	47.29%	55.43%	9359	5327	5800	3161	7.56
	MIX_modified	-15.716	-1.021	-8675.5	6.22%	634599.86	69.11%	1676	56.37%	67.05%	10546	2586	1436	23059	298.46
	Ours	-15.113	-0.418	-8302.4	1.66%	542740.68	59.11%	1421	47.80%	54.19%	10274	4606	5508	338	9.51
Superblue7	NC	-4.408	0.000	-756.8	0.00%	1719102.67	100.00%	5619	100.00%	100.00%					
	MS	-4.615	-0.207	-801.7	5.93%	941079.29	54.74%	2100	37.37%	55.21%	22740	11853	12653	7538	18.91
	MIX_modified	-4.510	-0.102	-907.6	19.93%	1098879.62	63.92%	2825	50.28%	66.64%	24463	4107	2712	64032	748.85
	Ours	-4.588	-0.180	-792.9	4.77%	890514.34	51.80%	1971	35.08%	53.86%	24536	10778	12281	523	24.29
Superblue10	NC	-40.855	0.000	-93697.3	0.00%	1593716.94	100.00%	5209	100.00%	100.00%					
	MS	-42.204	-1.349	-97615.6	4.18%	910477.62	57.13%	2147	41.22%	55.22%	20135	10413	11118	6779	16.19
	MIX_modified	-43.883	-3.028	-101000.0	7.79%	1036498.63	65.04%	2609	50.09%	65.78%	22864	4928	2606	44870	671.27
	Ours	-42.008	-1.153	-98312.5	4.93%	856849.51	53.76%	2018	38.74%	53.02%	23651	6986	7977	355	26.66
Superblue16	NC	-5.781	0.000	-1470.2	0.00%	943833.92	100.00%	3034	100.00%	100.00%					
	MS	-5.850	-0.069	-1557.9	5.97%	499360.29	52.91%	1028	33.88%	55.57%	11828	7213	7515	4037	9.53
	MIX_modified	-6.306	-0.525	-1978.6	34.58%	561250.95	59.47%	1293	42.62%	65.23%	14715	2889	2247	22132	365.84
	Ours	-5.830	-0.049	-1605.7	9.22%	472084.90	50.02%	970	31.97%	53.58%	13771	5643	5564	98	5.66
Superblue18	NC	-7.269	0.000	-1179.7	0.00%	633668.95	100.00%	2029	100.00%	100.00%					
	MS	-7.242	0.027	-1114.1	-5.56%	344257.63	54.33%	792	39.03%	54.83%	9114	4266	4451	2880	7.03
	MIX_modified	-7.532	-0.263	-1578.6	33.81%	396922.66	62.64%	960	47.31%	64.12%	10030	2546	1226	17336	293.92
	Ours	-7.195	0.074	-1180.3	0.05%	324213.09	51.16%	714	35.19%	52.89%	10301	3372	3228	127	3.41
Average	NC		0.000		0.00%		100.00%		100.00%	100.00%					
	MS		-0.310		3.10%		56.01%		39.27%	55.19%	0.88	1.29	1.28	23.31	1.27
	MIX_modified		-1.575		16.71%		64.20%		48.73%	65.70%	0.99	0.60	0.33	153.33	50.86
	Ours		-0.251		3.89%		53.21%		36.89%	53.38%	1.00	1.00	1.00	1.00	1.00

Effectiveness of Slack Redistribution

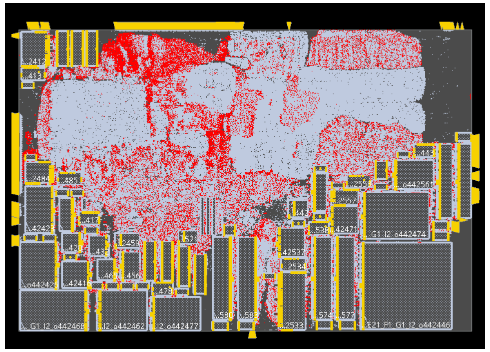
- With (w/SR) slack redistribution, the power consumption further decreases by 2.82% on average
  - TNS and WNS degradation also reduces with almost no runtime overhead.

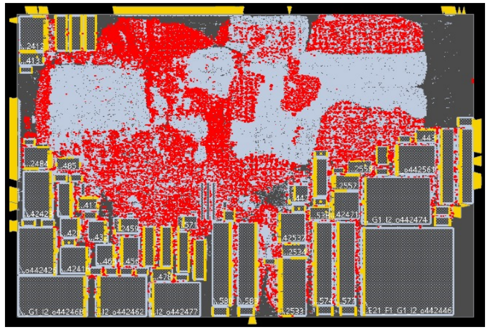
Table 3: Effectiveness of Slack Redistribution (SR).

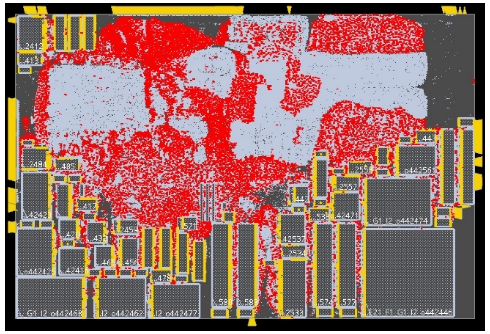
Designs	Method	FF Power Ratio	WNS (ns)	TNS (s)	Runtime (s)
Superblue1	w/o SR	56.44%	-8.258	-1412.1	8.09
	w/ SR	53.74%	-8.215	-1410.7	7.72
Superblue3	w/o SR	55.71%	-14.225	-2405.9	9.93
	w/ SR	52.91%	-14.153	-2375.4	9.94
Superblue4	w/o SR	55.84%	-8.014	-7464.5	6.31
	w/ SR	52.89%	-8.010	-7332.9	6.37
Superblue5	w/o SR	57.01%	-14.651	-8254.9	9.47
	w/ SR	54.19%	-15.113	-8302.4	9.51
Superblue7	w/o SR	56.46%	-4.401	-784.5	24.04
	w/ SR	53.86%	-4.588	-792.9	24.29
Superblue10	w/o SR	55.80%	-43.523	-102000.0	26.62
	w/ SR	53.02%	-42.008	-98312.5	26.66
Superblue16	w/o SR	56.94%	-5.872	-1592.4	5.75
	w/ SR	53.58%	-5.830	-1605.7	5.66
Superblue18	w/o SR	55.42%	-7.300	-1224.3	3.34
	w/ SR	52.89%	-7.195	-1180.3	3.41
Difference/ Ratio	w/o SR	56.20%	-0.393	4.93%	1.00
	w/ SR	53.38%	-0.251	3.89%	1.00

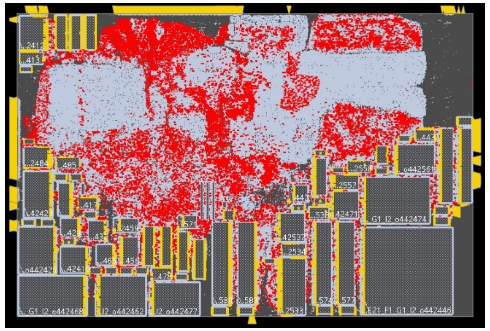
Layout Comparisons – Full

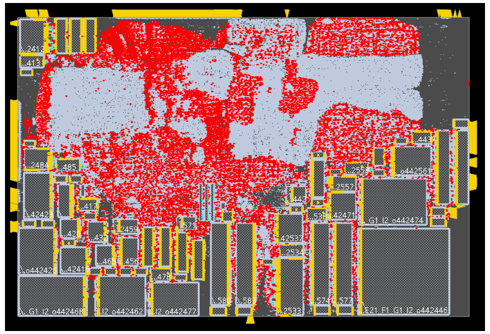
- FFs become more densely packed.


Non-clustered


Ours w/o SR

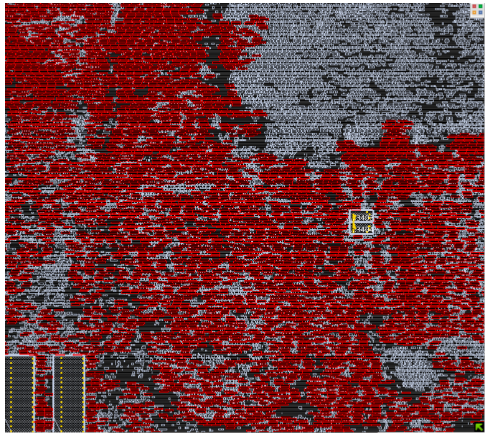

MS


MIX_modified

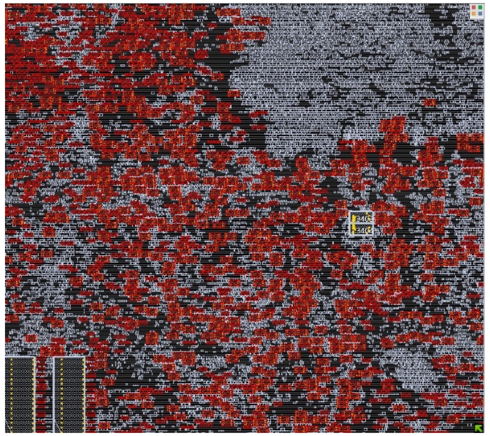

Ours w/ SR

Layout Comparisons - Partial

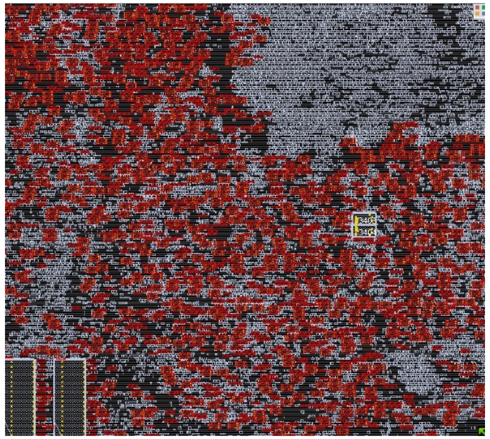
- Groups of flip-flops are replaced by MBFFs that occupy multiple rows and have larger areas.



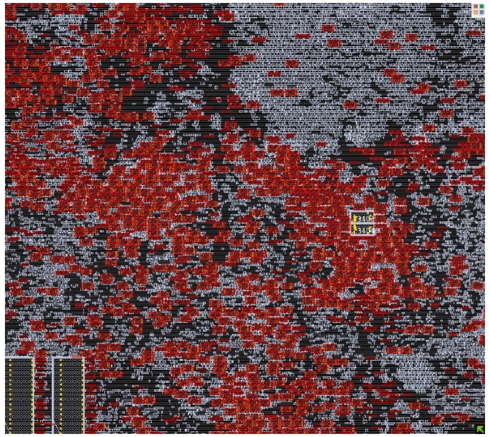
Non-clustered



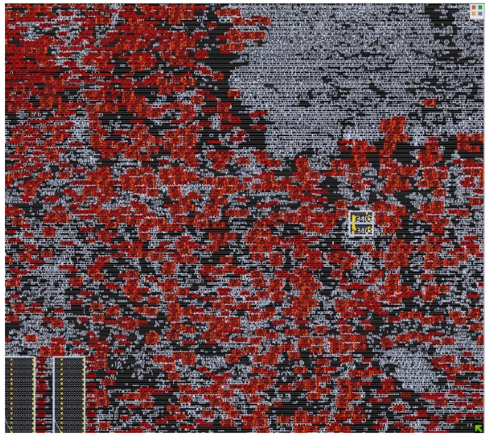
Ours w/o SR



MS



MIX_modified



Ours w/ SR

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- **Conclusion**

Conclusions

- We propose a register clustering algorithm with novel path slack redistribution technique.
  - Use window-based sequence generation to effectively prevent unexpected timing degradation and runtime overhead.
  - Use red-black trees for changing feasible regions.
  - Propose an inclusion force model to create more perfectly sized MBFFs.
  - Release slack to increase the clustering potential to unclustered FFs.

- Experimental results show our algorithm achieve superior performance in terms of clock power reduction, timing balancing and runtime.

Thank you