DeFer: Deferred Decision Making Enabled Fixed-Outline Floorplanning Algorithm

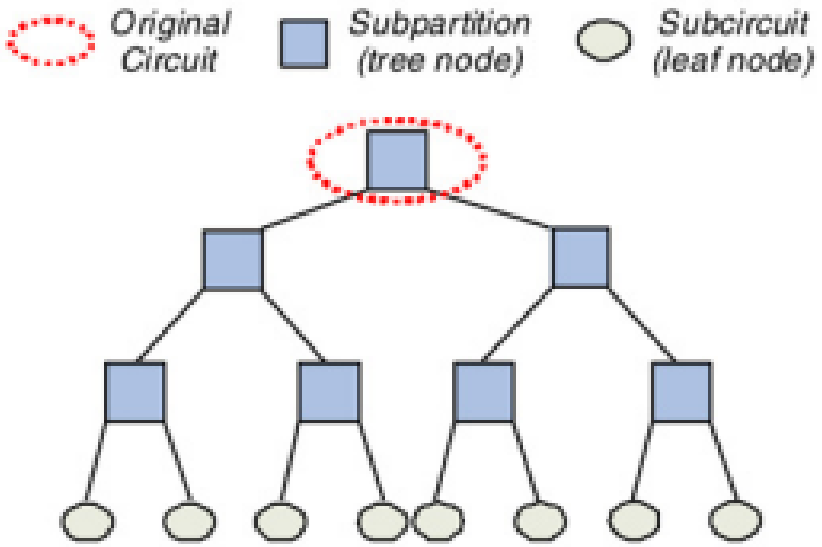
Elmir Dzaka

Department of Electrical and Computer Engineering

University of Utah, Salt Lake City, UT

Background – Deferred Decision Making (DDM)

- Refers to collecting evidence about two or more alternatives
- Decide when to stop and make a final choice
- Compacts a slicing floorplan when applied to a floor-planning algorithm



Why Should We Care?

- A single slicing tree can be used as opposed to the simulated annealing method
  - Drastically increases performance
  - Used to generate a final non-slicing floorplan
- How can we develop an algorithm that can generate a final non-slicing floorplan?

Formal Problem Formulation

- Input: *n* hard/soft module blocks with areas $A_1, \ldots, A_n$
- Constraints: a fixed-outline floorplan with area A
- Output: An optimized non-slicing floorplan with coordinates (x_i, y_i), width w_i and height h_i for each block such that $h_i w_i = A_i$ and $r_i \leq h_i/w_i \leq s_i$

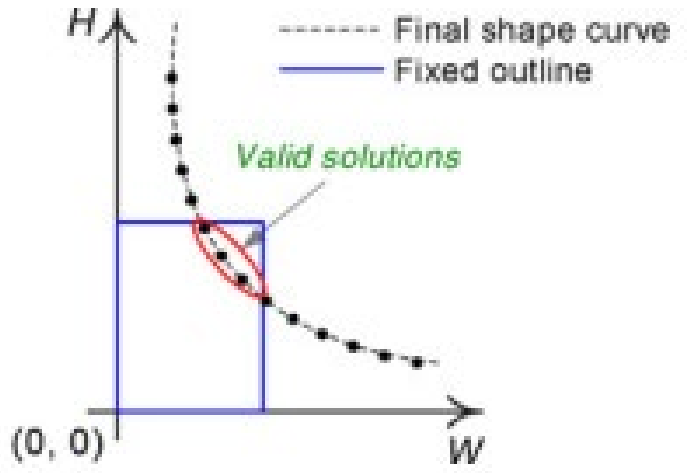
DeFer

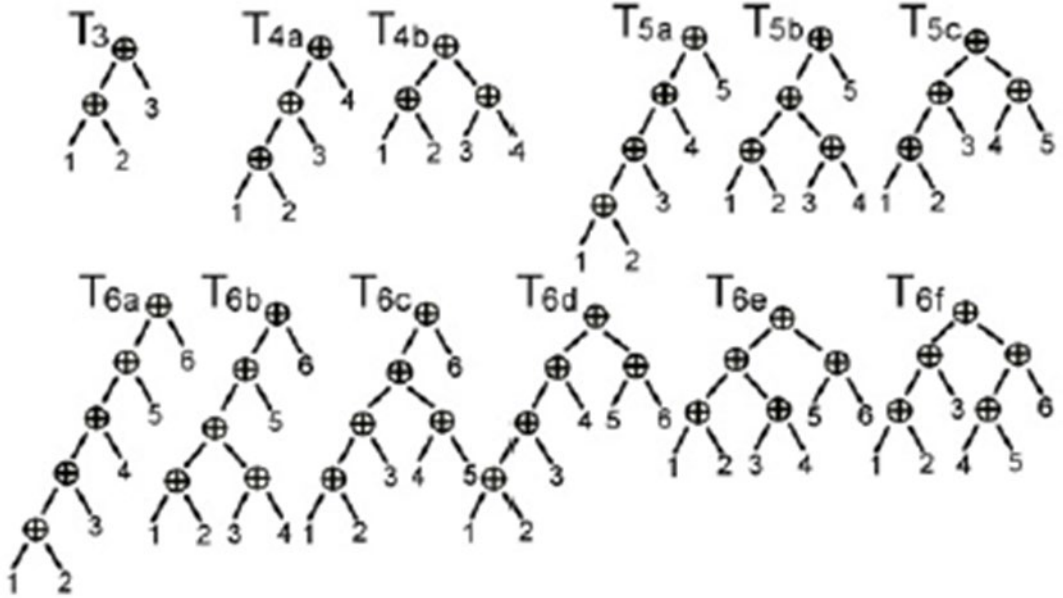
- Algorithm that chooses the final non-slicing floorplan from a single slicing tree
- Fast, scalable, and can handle both hard and soft modules
- Uses a Generalized Slicing Tree along with Enumerative Packing and block swapping/mirroring to make a decision

Algorithm Flow of *DeFer*
Begin
Step 1): Top-down recursive min-cut bisectioning
Step 2): Bottom-up recursive shape curve combination
Step 3): Top-down tracing selected points
Step 4): Top-down wirelength refinement by swapping
Step 5): Slicing floorplan compaction
Step 6): Greedy wirelength-driven shifting
End

Purpose of a Generalized Slicing Tree

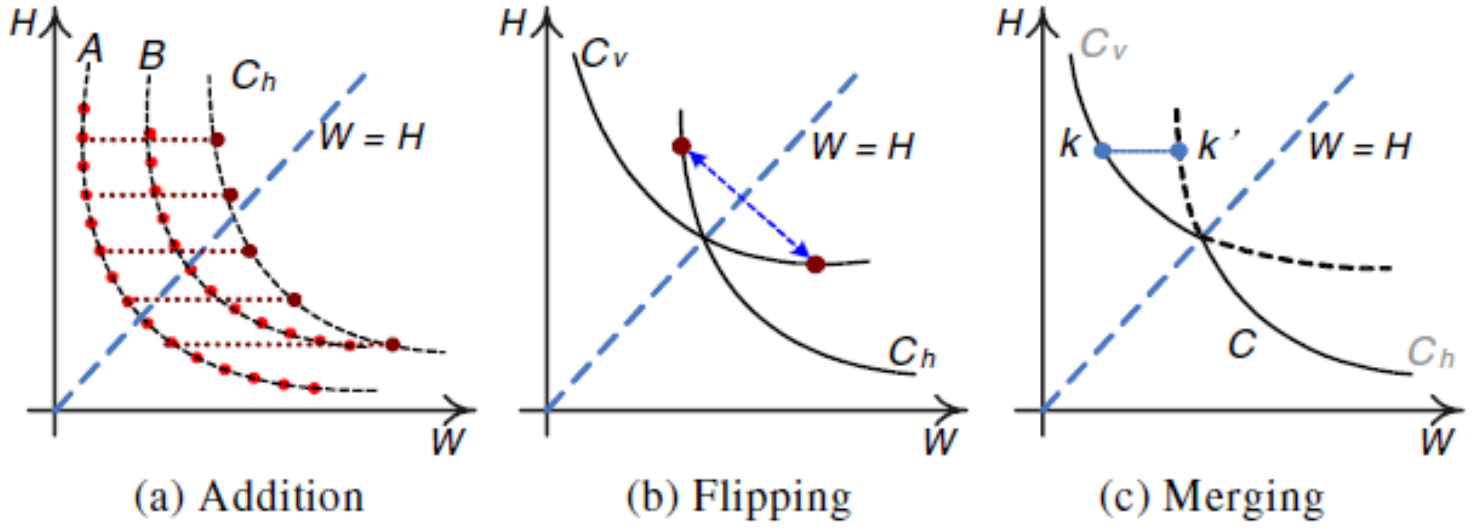
- A Generalized Slicing Tree can be enumerated to reach all slicing layouts
  - Using the enumerative packing (EP) technique while enumerating builds up one shape curve
  - A shape curve captures all slicing layouts among the modules of a sub floorplan





Shape Curve

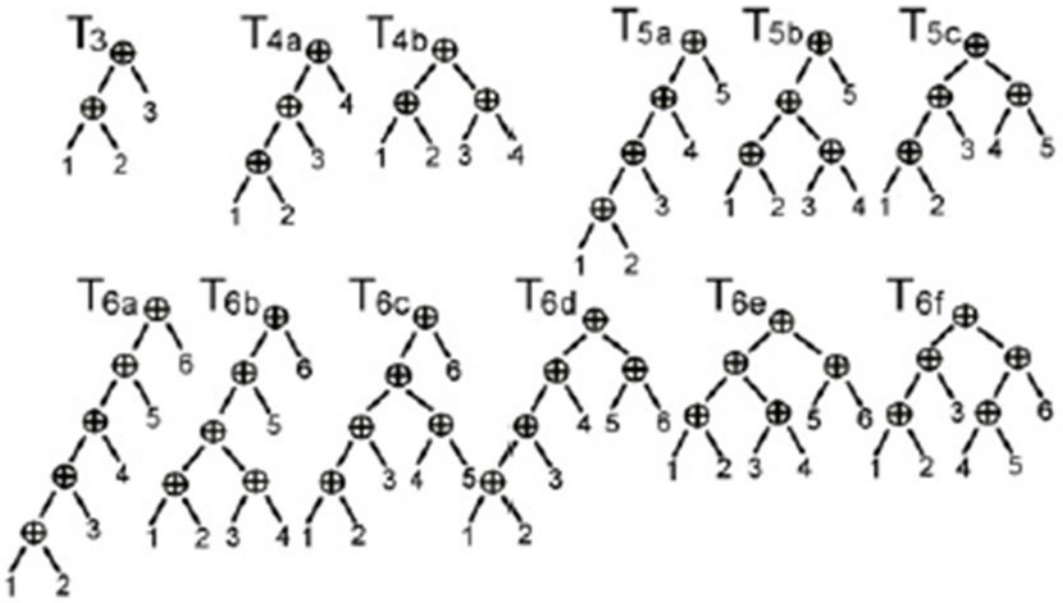
- Generate a parent curve C by finding child curves A and B
- Three steps:
  - Addition
  - Flipping
  - Merging

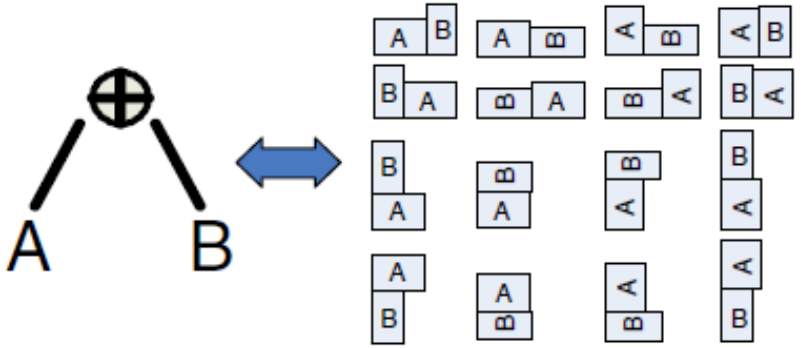


(a) Addition (b) Flipping (c) Merging

Enumerative Packing (Greedy)

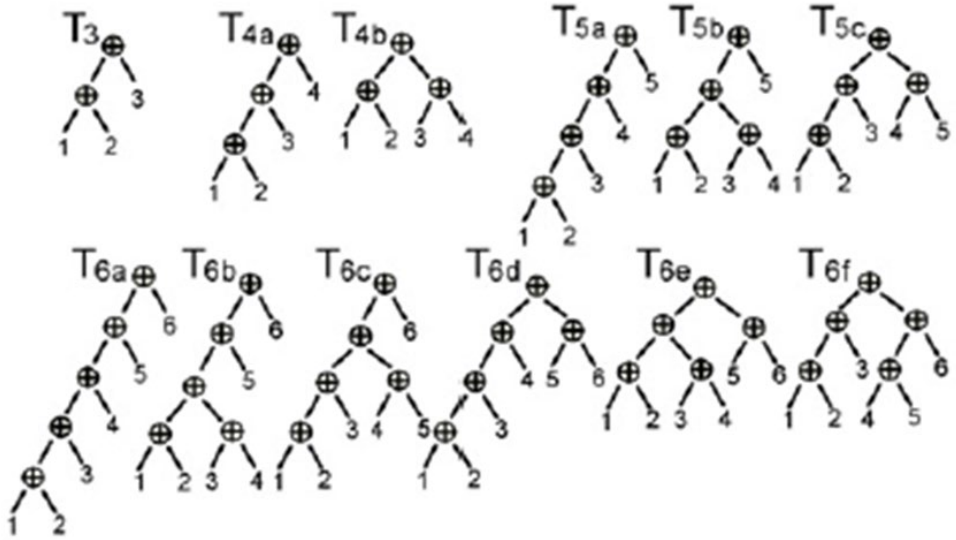
- Map each subcircuit to a leaf node
- Prune redundant orders to find non-redundant permutations
  - N!/2

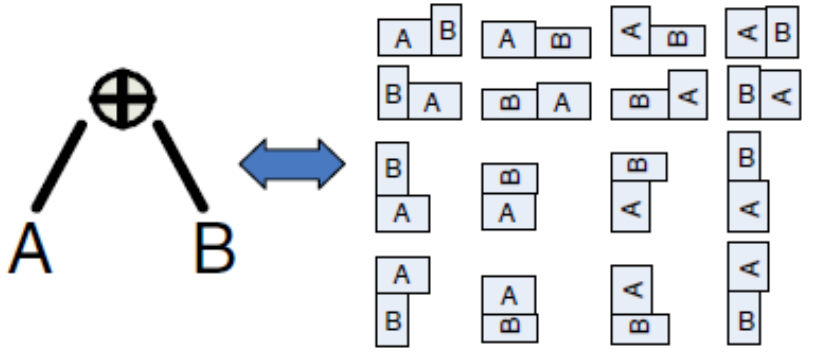




Enumerative Packing (Dynamic Programming)

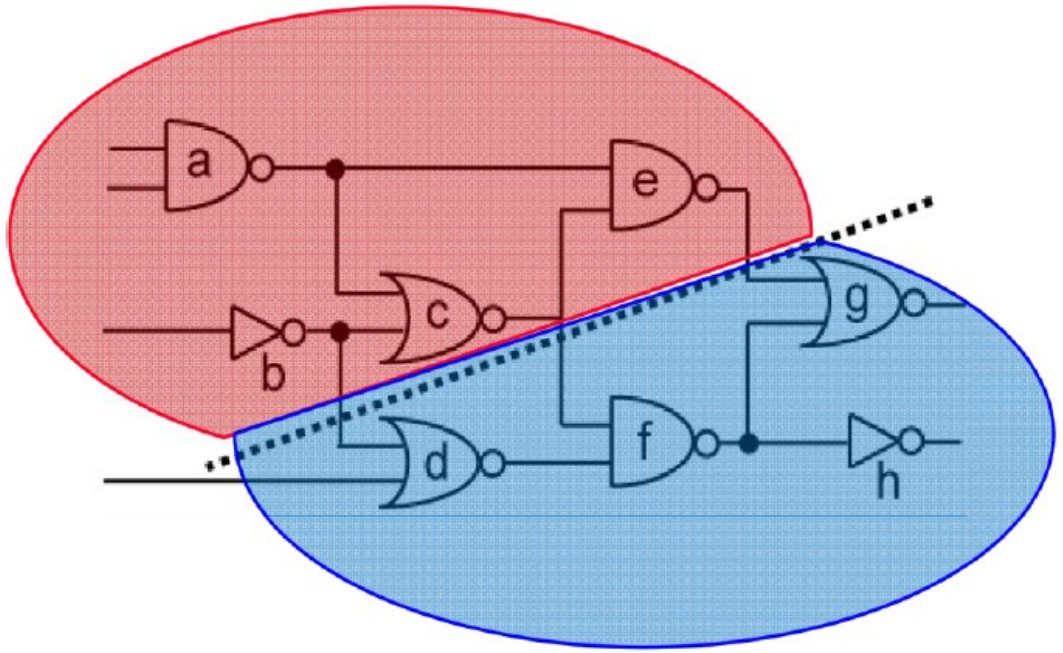
- $S(M) = \text{MERGE}_{(A \subset B, B = M - A)} (S(A) \oplus S(B))$.
  - S(M) = Shape Curve
  - MERGE = Merge operation
  - $(S(A) \oplus S(B))$ = shape curve formula





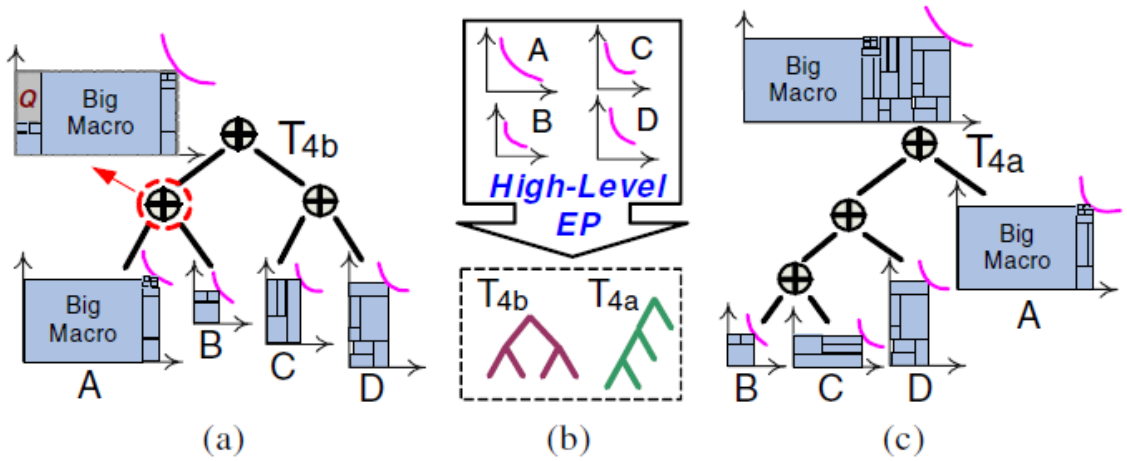
Step One: Partitioning

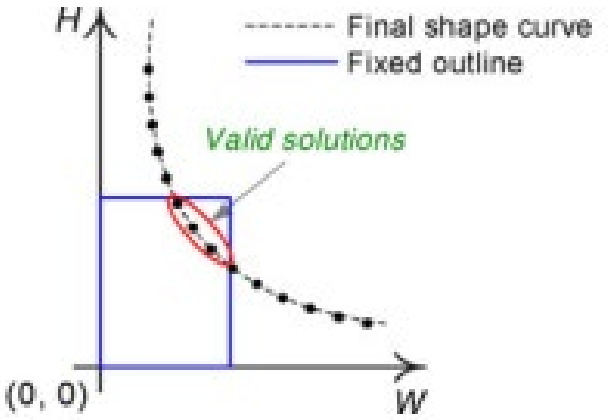
- Divide the original circuit into multiple sub-circuits
  - Minimize interconnections among them
  - Done using hMetris
- Build a high-level slicing tree with each sub-circuit being a leaf node



Step Two: Combining

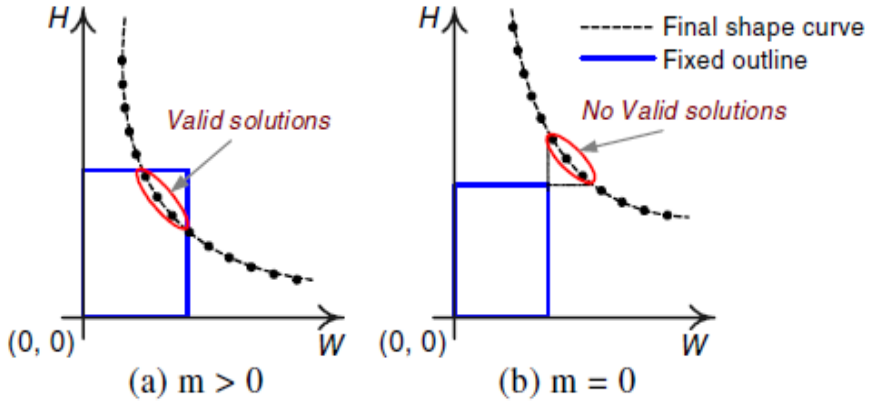
- Defer the decision of each subcircuit by applying the EP technique
  - Explores all slicing packing layouts within the subcircuit
  - The final shape curve at the root *maintains all explored slicing floorplan layouts*
  - WAP (Whitespace-Aware Pruning) prunes redundant points





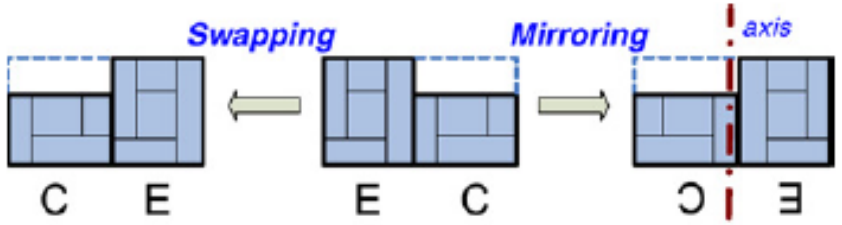
Step Three: Back-Tracing

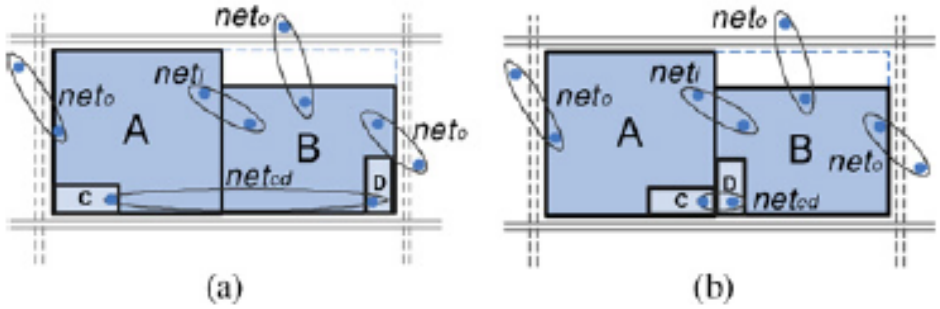
- Final shape curve is generated from candidate points of subcircuits
  - Backtrace from top-down to generate candidate points
  - Every point is generated by adding two points from two child curves



Step Four: Swapping

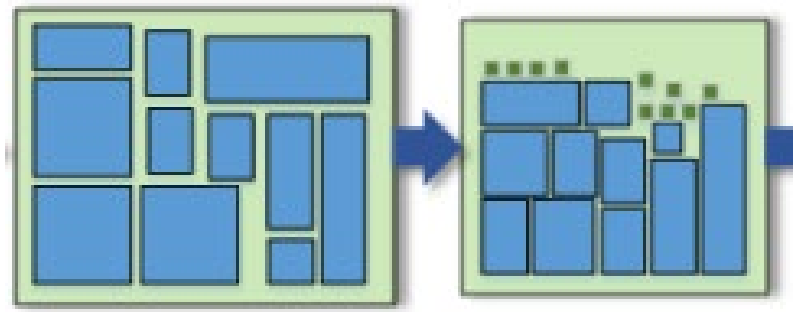
- Make decisions on the subfloorplan order
  - Dimensions stay the same, but wire interconnections are optimized
- Greedily swap every two child subfloorplans
  - Rough Swapping
  - Detailed Swapping
  - Mirroring





Step Five: Compacting and Shifting

- Compact all modules to the center of the fixed outline
  - Puts modules nearer to each other such that the wirelength is further reduced
- Candidate floorplan with the best wirelength is the final output solution
- Greedily shift modules if modules are over-compacted



2. Initial floorplan 3. Compact floorplan

Contributions

- A fast and scalable fixed-outline floorplanner
  - Done by using a Generalized Slicing Tree
- Enumerative Packing (EP)
  - Defer decisions on all sub-circuits to get a final shape curve
- Rough Swapping and Mirroring

Experimental Results

- Results performed on a Linux machine with Intel Duo[9] 1.86 GHz CPU and 2GB memory
- Wirelength measured by HPWL
- DeFer compared with best publicly-available floor-planners

Experimental Results: Hard-Block Benchmarks

TABLE II

COMPARISON ON GSRC HARD-BLOCK BENCHMARKS [22] (γ = 10%)

Circuit		*n*100			*n*200			*n*300			**Normallized**
Aspect Ratio		1	2	3	1	2	3	1	2	3	
Suc%	*Parquet 4.5*	42%	43%	33%	26%	19%	17%	16%	16%	14%	**0.25**
	FSA	100%	0%	0%	100%	0%	0%	0%	0%	0%	**0.22**
	IMF	100%	100%	100%	100%	100%	100%	100%	100%	100%	**1.00**
	IARFP	99%	100%	99%	100%	99%	63%	100%	100%	46%	**0.90**
	PATOMA	0%	0%	0%	0%	100%	0%	100%	100%	100%	**0.44**
	Capo 10.5	17%	17%	15%	0%	0%	2%	0%	1%	0%	**0.06**
	DeFer	100%	100%	100%	100%	100%	100%	100%	100%	100%	**1**
HPWL	*Parquet 4.5*	248 652	269 191	289 963	467 627	506 946	544 621	686 588	725 833	781 556	**1.27**
	FSA	243 823	–	–	414 777	–	–	–	–	–	**1.14**
	IMF	250 680	251 418	257 935	438 467	454 231	482 651	584 578	617 510	666 245	**1.14**
	IARFP	220 269	230 553	247 283	386 537	409 208	433 631	535 850	567 496	600 438	**1.03**
	PATOMA	–	–	–	–	483 110	–	653 711	697 740	680 671	**1.25**
	Capo 10.5	227 046	241 789	261 334	–	–	444 079	–	566 998	–	**1.05**
	DeFer	208 650	229 603	248 567	372 546	402 155	431 552	498 909	538 515	577 209	**1**
Time (s)	*Parquet 4.5*	10.85	10.58	10.27	44.43	44.47	41.96	95.02	87.03	86.31	**181.49**
	FSA	39.78	–	–	202.13	–	–	–	–	–	**557.74**
	IMF	7.65	10.82	9.29	41.21	43.59	38.71	74.74	71.48	71.72	**157.91**
	IARFP	4.44	4.50	4.52	16.51	15.48	14.22	29.30	29.48	30.03	**64.33**
	PATOMA	–	–	–	–	0.25	–	0.36	0.34	0.48	**1.15**
	Capo 10.5	122.64	125.18	160.07	–	–	3054	–	8661	–	**222.39**
	DeFer	0.13	0.11	0.11	0.25	0.23	0.22	0.35	0.33	0.33	**1**
#Valid Point/#Total Point		3/617	4/621	3/621	3/670	2/672	2/672	6/869	5/869	4/869	

Experimental Results: Soft-Block Benchmarks

TABLE III

COMPARISON ON GSRC SOFT-BLOCK BENCHMARKS [22] ($\gamma = 1\%$)

Circuit		$n100$			$n200$			$n300$			**Normallized**
Aspect Ratio		1	2	3	1	2	3	1	2	3	
Suc%	*Parquet 4.5*	0%	0%	0%	0%	0%	0%	0%	0%	0%	**0**
	Capo 10.5	0%	0%	0%	0%	0%	0%	0%	0%	0%	**0**
	PATOMA	100%	100%	100%	100%	100%	100%	100%	100%	100%	**1.00**
	DeFer	100%	100%	100%	100%	100%	100%	100%	100%	100%	**1**
HPWL	*Parquet 4.5*	–	–	–	–	–	–	–	–	–	**—**
	Capo 10.5	–	–	–	–	–	–	–	–	–	**—**
	PATOMA	215 455	213 561	230 759	383 330	367 565	404 574	524 774	486 351	518 204	**1.01**
	DeFer	196 457	217 686	235 702	354 885	380 470	410 464	476 508	514 764	551 610	**1**
Time (s)	*Parquet 4.5*	–	–	–	–	–	–	–	–	–	**—**
	Capo 10.5	–	–	–	–	–	–	–	–	–	**—**
	PATOMA	0.39	0.40	0.38	0.92	0.93	0.83	1.28	1.28	1.37	**3.50**
	DeFer	0.09	0.09	0.09	0.18	0.19	0.19	0.78	0.96	0.97	**1**
#Valid Point/#Total Point		28/20 392	30/20 469	30/20 469	16/25 513	18/25 493	17/25 493	9/30 613	10/30 598	10/30 603	

Experimental Results: Macro Analysis

TABLE IV
COMPARISON ON HB BENCHMARKS [24] (γ = 10%)

Circuit	#Soft./#Hard. /#Net.	Aspect Ratio	*PATOMA* [14] Suc%	*PATOMA* [14] WL (e+06)	*PATOMA* [14] Time (s)	*Capo 10.5* [5] Suc%	*Capo 10.5* [5] WL (e+06)	*Capo 10.5* [5] Time (s)	*DeFer* Suc%	*DeFer* WL (e+06)	*DeFer* Time (s)	#Valid Point /#Total Point
ibm01	665	1	100%	2.84	7.04	0%	–	183	100%	2.66	1.44	16/1571
	/246	2	0%	–	–	0%	–	977	100%	2.70	1.28	11/1482
	/4236	3	100%	5.60	1.66	0%	–	696	100%	2.82	1.30	12/1490
ibm02	1200	1	0%	–	–	0%	–	456	85%	6.55	14.48	6/2348
	/271	2	0%	–	–	–	–	*> 2 days*	100%	6.21	3.33	7/1161
	/7652	3	0%	–	–	0%	–	3726	100%	6.29	3.52	10/1144
ibm03	999	1	100%	12.59	5.42	100%	10.70	566	100%	8.77	3.60	59/2684
	/290	2	100%	12.94	5.58	100%	12.01	1874	100%	8.89	3.49	40/2503
	/7956	3	0%	–	–	0%	–	2028	100%	8.99	3.59	44/2630
ibm04	1289	1	0%	–	–	0%	–	2752	100%	8.94	3.04	4/1492
	/295	2	0%	–	–	100%	17.77	5253	100%	8.96	3.12	9/1514
	/10 055	3	0%	–	–	100%	16.32	2262	100%	9.64	6.31	12/2685
ibm05	564	1	100%	12.27	14.21	0%	–	458	100%	12.61	3.55	46/3369
	/0	2	100%	12.60	13.68	0%	–	358	100%	12.73	3.52	46/3371
	/7887	3	100%	13.19	13.85	0%	–	411	100%	13.45	3.53	46/3371
ibm06	571	1	0%	–	–	0%	–	235	100%	7.87	3.66	53/2187
	/178	2	0%	–	–	0%	–	592	100%	7.76	3.66	41/2235
	/7211	3	0%	–	—	0%	–	2831	100%	8.91	3.60	36/2196
ibm07	829	1	0%	–	–	0%	–	1094	100%	13.81	3.87	12/1527
	/291	2	100%	24.64	7.85	0%	–	1270	100%	13.91	4.48	22/1625
	/11 109	3	100%	24.34	8.68	0%	–	2274	100%	14.32	4.26	18/1590
ibm08	968	1	0%	–	–	0%	–	2527	100%	13.95	5.44	15/1333
	/301	2	0%	–	–	0%	–	1110	100%	14.16	5.40	17/1290
	/11 536	3	0%	–	–	0%	–	1958	100%	14.43	5.55	19/1309
ibm09	860	1	0%	–	–	0%	–	2273	100%	12.85	2.60	3/1495
	/253	2	0%	–	–	0%	–	2670	100%	12.57	3.77	17/1486
	/11 008	3	0%	–	–	100%	34.48	6652	100%	12.98	3.54	14/1486
ibm10	809	1	100%	48.47	21.71	0%	–	2353	100%	33.25	11.63	9/2576
	/786	2	0%	–	–	*Crashed*	*Crashed*	*Crashed*	100%	34.23	18.00	14/2897
	/16 334	3	0%	–	–	100%	53.64	2014	100%	36.59	16.52	9/2725
ibm11	1124	1	100%	20.87	33.87	0%	–	8070	100%	21.99	4.84	12/2218
	/373	2	0%	–	–	0%	–	4732	100%	22.13	4.96	8/2207
	/16 985	3	0%	–	–	0%	–	2245	100%	22.83	4.67	7/2174
ibm12	582	1	0%	–	–	0%	–	3085	100%	29.72	10.95	20/2909
	/651	2	0%	–	–	0%	–	864	100%	31.53	7.71	18/3011
	/11 873	3	0%	–	–	0%	–	19 952	100%	32.16	4.59	8/1957
ibm13	530	1	0%	–	–	0%	–	3401	100%	25.92	6.03	12/2553
	/424	2	100%	43.81	9.84	0%	–	3662	100%	25.46	3.79	10/2048
	/14 202	3	0%	–	–	0%	–	3201	100%	26.47	3.83	8/2095
ibm14	1021	1	100%	71.87	23.59	0%	–	4253	100%	50.83	9.69	30/2976
	/614	2	100%	55.99	35.65	0%	–	10 373	100%	51.67	9.70	34/2971
	/26 675	3	100%	61.65	35.12	0%	–	4976	100%	53.71	9.70	36/2971
ibm15	1019	1	0%	–	–	0%	–	3634	100%	64.18	9.71	25/1651
	/393	2	0%	–	–	0%	–	6827	100%	63.17	9.13	19/1580
	/28 270	3	0%	–	–	0%	–	2902	100%	66.06	9.46	20/1623
ibm16	633	1	0%	–	–	*Crashed*	*Crashed*	*Crashed*	100%	56.88	16.79	18/3823
	/458	2	100%	88.33	16.55	0%	–	8928	100%	58.55	14.55	24/4833
	/21 013	3	100%	98.77	22.94	0%	–	11 675	100%	59.91	12.84	18/4093
ibm17	682	1	100%	102.45	41.75	*Crashed*	*Crashed*	*Crashed*	100%	95.92	10.43	32/3253
	/760	2	100%	96.46	46.63	0%	–	2250	100%	95.48	10.41	27/3252
	/30 556	3	100%	98.18	42.45	*Crashed*	*Crashed*	*Crashed*	100%	100.82	10.42	29/3252
ibm18	658	1	100%	50.28	38.24	0%	–	1083	100%	49.12	7.93	42/3106
	/285	2	100%	49.74	39.15	0%	–	4630	100%	49.29	7.97	41/3128
	/21 191	3	100%	52.26	36.97	0%	–	5262	100%	51.39	7.97	41/3128
Normalized			**0.43**	**1.28**	**3.28**	**0.12**	**1.72**	**789.79**	**1**	**1**	**1**	

Pros and Cons of the Work

- Pros
  - Massive runtime performance increase compared to SA solutions
  - Scalable
  - Works for both hard and soft blocks

- Cons
  - Several greedy algorithms
  - Lots of math involved
  - A final shape curve that doesn’t have a valid solution requires the algorithm running until it does

Summary

- Reviewed the background of Deferred Decision Making
- Analyzed the technical specifications of DeFer
- Compared DeFer to other fixed-outline floorplanning solutions
- Discussed the pros and cons surrounding the algorithm