RELAXED CONSTRAINTS FOR MINCUT PARTITIONING IN TOP-DOWN PLACEMENT

Andrew Caldwell, **Andrew Kahng**, Igor Markov
UCLA, Computer Science Department*
{caldwell,abk,imarkov}@cs.ucla.edu

*Supported by a grant from Cadence Design Systems, Inc.

Outline of the Talk

- **Introduction**
  - Placement via recursive *balanced* bisection
  - Simon and Teng: *exact* mincut bisection provably bad
  - potentially better with *relaxed* balance constraints

- **Experimental Question**
  - Improve VLSI placement by relaxing balance constraints

- **Results**
  - Final wirelength better with relaxed constraints
  - Changes in Rent parameter small,
    no clear correlation to changes in wirelength

- **Conclusions and ongoing research**

Row-Based VLSI Placement

Given:

- Circuit hypergraph
- Discrete legal cell locations (sites) in rows
- Constraints on cell locations, e.g., fixed pads, blockages
- Objective function for cell placement

Find:

- Cell locations minimizing objective subject to constraints

Additional considerations:

- Timing- and routability-driven placement
- Problem sizes of interest: 10^5 and above

Half-Perimeter Bounding Box Objective

- For a net: half-perimeter of bounding box of pin locations
- For circuit hypergraph: sum over all nets
- Compromise between accuracy and efficiency:
  easy to evaluate and approximates routed wirelength
- No known methods to minimize directly
- Typically optimized by recursive *mincut* bisection
- *Net cut* correlated with *half-perimeter wirelength*
- Analytic placement can guide partitioning via graph models

Mincut Partitioning

Advantages:

- Well-studied, fast
- Represents a "divide and conquer" approach
- Difficult to drive partitioners with alternate objectives

Disadvantages:

- "Wrong objective" for placement
- Under-utilizes routing area over the cutlines
- Hard to capture timing constraints

Recursive Bisection vs k-way Partitioning

Recursive bisection:

- Heuristic for multi-way partitioning
- Relies on highly optimized bipartitioning codes
- Easy to implement, good runtimes

Multi-way partitioning:

- Better handles partition balances
- Potentially better solutions

Simon and Teng, 1997

- Analysis of *graphs* only, not *hypergraphs*
- Assume optimal recursive bipartitioning, i.e., all bisections optimal for given constraints
- Compare best possible recursive bisection solutions to best k-way partitioning solutions

Recursive bisection with exact balance constraints

- Can yield cuts $\Omega(n/k)$ from optimal on families of graphs

Recursive bisection with relaxed balance constraints

- Proven to yield cuts within $O(log(k))$ of optimal
- Considerably relaxed constraints required

4-way partitioning example

Simon and Teng build a family of graphs

- 8 sparse expanders $A_1, A_2, A_3, A_4, B_1, B_2, B_3, B_4$
  (any exact bisection has cost $\Omega(n)$)

- Each A_i has $(1/8 + \varepsilon_i)n$ vertices, each B_i has $(1/8 - \varepsilon_i)n$

- $-1/8 < \varepsilon_i < 1/8$ and $\varepsilon_i \neq 0$

- $\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4 = 0$

- $\forall i, j \ (\varepsilon_j + \varepsilon_j) \neq 0$

4-way partitioning example (cont'd)

- Exist for all n
  (follows from result of Erdös, Graham and Szemerèdi)
- Connect each A_i to B_i by one edge (4 edges)
- Add a triple cycle through A_1, A_2, A_3 and A_4 (12 edges)
- Optimal 4-way partition $\{A_i \cup B_i\}_i$ has cut 12
- Optimal bisection first gives $\cup A_i$ and $\cup B_i$
- Next bisection has to partition one of expanders
- Recursive bisection cut is off by $\Omega(n)$

Rent's rule

- Empirical relation observed for subcircuits of "good" layouts
- For *spatially defined* subcircuit connects # cells C to # external terminals T
- Power law: $T = kC^p$
- $0 \leq p \leq 1$ is the *Rent parameter*
- Wirelength estimates in terms of *Rent parameter*: smaller values of p correspond to smaller wirelength
- In *top-down placement* subcircuits are defined by partitions
- *Rent parameter* determined by linear regression

Experimental Questions

- Do Simon and Teng's results apply to circuit hypergraphs ?

- In particular, will smaller *net cuts* be achieved, resulting in smaller *half-perimeter wirelength*?

- Will *run time* and/or *Rent parameter* be affected?

Experimental Testbed

- Partitioning-driven top-down placement engine
- 3 industry test cases in LEF/DEF format

Testcase	Core Cells	Nets	I/O Pads
Case 1	2741	2842	545
Case 2	11471	11673	662
Case 3	20392	25634	185

- Rent parameter computation following Hagen et al. '94
- Core rows horizontal
- No macros, no cell heights greater than one row height

Top-Down Placer

- Driven by recursive mincut bisection
- Always cuts in the longest direction
- Vertical cuts
  - target balance 50%/50%
  - tolerance varied from smallest cell size to 40%
  - cut line adjusted after partitioning

Top-Down Placer (cont'd)

- Discrete balance tolerance for bisections *parallel* to rows not supported by traditional partitioning algorithms

- Tight tolerance required for "horizontal" bisections to provide row area for all cells

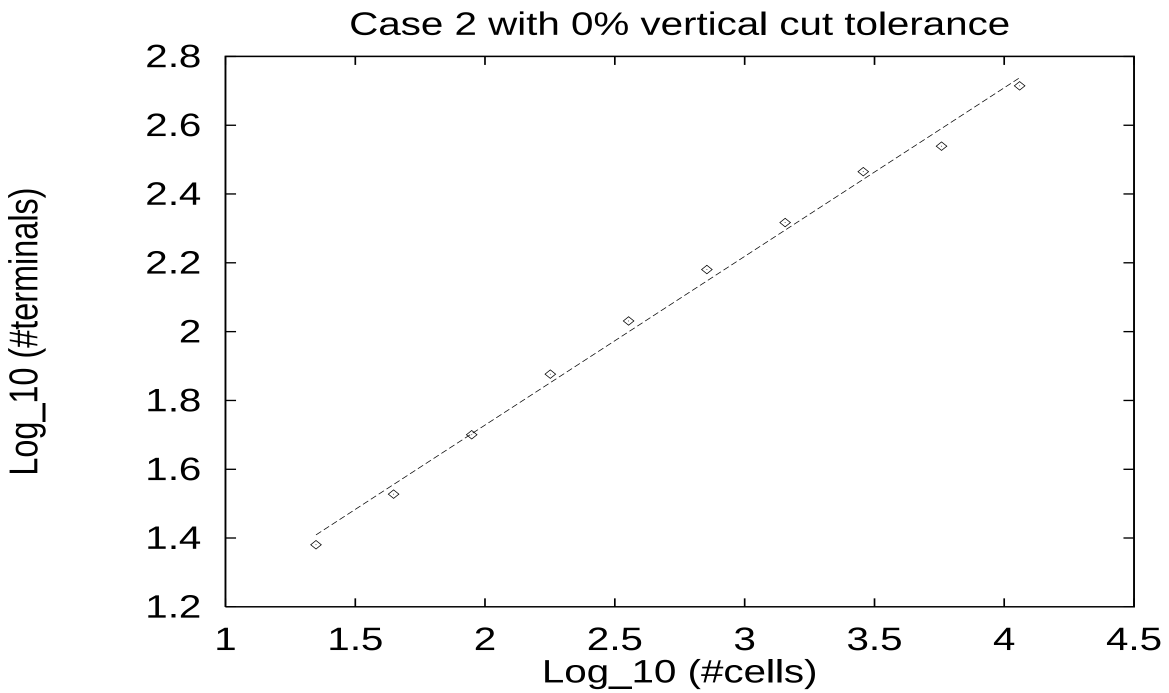
- Horizontal cuts
  - target balance closest to 50%/50% w/o splitting rows
  - minimal possible tolerance (smallest cell size)

Mincut Partitioner

- *Multi-level* multi-start partitioner
- Uses *Clip FM* at each level
- *Heavy edge matching* clustering with *matching ratio* 1/6
- *Level growth* ratio 1.3
- Each bisection uses same clustering tree for all starts
- *Starts*: 10 for instances > 200, 5 otherwise

Experiments

- Varied tolerance for vertical bisections, adjusted cut line
- Each configuration ran 20 times, min/max/average reported
- Rent parameter computed via least squares line fit



Results —– Case 1

V-cut tol.	Time Ave	Wirelength $\times 10^7$ Max	Min	Ave	Top-level net-cut Max	Min	Ave	Rent par Ave
0%	137	6.75	6.35	6.57	247	223	240	0.663
5%	151	6.77	6.44	6.60	243	220	232	0.665
10%	150	6.67	6.29	6.44	243	213	226	0.673
15%	151	6.50	6.30	6.42	228	213	222	0.670
17%	154	6.57	6.32	6.39	223	214	219	0.673
19%	**150**	**6.47**	**6.26**	**6.37**	**225**	**209**	**216**	**0.672**
20%	151	6.59	6.28	6.39	222	208	214	0.672
21%	151	6.55	6.25	6.39	223	206	214	0.673
23%	150	6.54	6.26	6.40	216	199	206	0.671
25%	152	6.56	6.33	6.42	208	195	203	0.673
30%	153	6.57	6.37	6.47	195	189	192	0.672
40%	157	6.60	6.40	6.48	181	168	175	0.670

Results — Case 2

V-cut	Time	Wirelength $\times 10^8$			Top-level net-cut			Rent par
tol.	Ave	Max	Min	Ave	Max	Min	Ave	Ave
0%	879	3.02	2.93	2.97	335	296	302	0.489
5%	915	3.08	2.99	3.03	310	291	295	0.492
10%	909	2.96	2.89	2.93	304	292	296	0.495
15%	909	2.97	2.87	2.91	306	292	299	0.496
20%	**926**	**2.98**	**2.86**	**2.90**	**303**	**292**	**295**	**0.496**
25%	921	2.96	2.87	2.91	312	293	298	0.498
30%	930	2.94	2.88	2.92	312	289	297	0.496
40%	1001	2.99	2.90	2.94	311	291	297	0.497

Results — Case 3

V-cut tol.	Time Ave	Wirelength $\times 10^8$ Max	Min	Ave	Top-level net-cut Max	Min	Ave	Rent par Ave
0%	2590	6.76	6.41	6.59	383	298	331	0.422
5%	2528	6.70	6.41	6.57	332	287	298	0.420
10%	2567	6.63	6.31	6.42	307	281	296	0.439
15%	**2662**	**6.61**	**6.28**	**6.41**	**316**	**282**	**290**	**0.441**
20%	2691	6.61	6.31	6.43	325	263	283	0.438
25%	2714	6.62	6.27	6.43	273	252	258	0.434
30%	**2607**	**6.58**	**6.28**	**6.40**	**260**	**255**	**250**	**0.435**
40%	2649	6.58	6.30	6.45	260	253	251	0.449

Discussion

- Average and minimum wirelength affected by tolerance:
  - decrease as tolerance increases to $10 - 15\%$
  - increase afterwards

- Averages do not drift away from minima:
  placer can predictably produce good solutions in few starts

- Changes in CPU time within 10%
  justified by better quality (several percent)

- Changes in Rent parameter small, uncorrelated to changes in wirelength

Conclusions

- Simple, transparent change to existing placers demonstrated

- Reliably reduces final wirelength and intermediate net cuts

- Implementation and run time overhead negligible

- No clear correlation between Rent parameter
  and changes in wirelength

Ongoing research

- Increasing tolerance for horizontal cuts
- Applying multi-way partitioning instead of recursive bisection
  - better handles balances
  - can improve solution quality
- Achieving more wirelength reduction due to relaxed balance tolerances
- Are circuit hypergraphs “well-behaved” compared to Simon and Teng’s graphs?