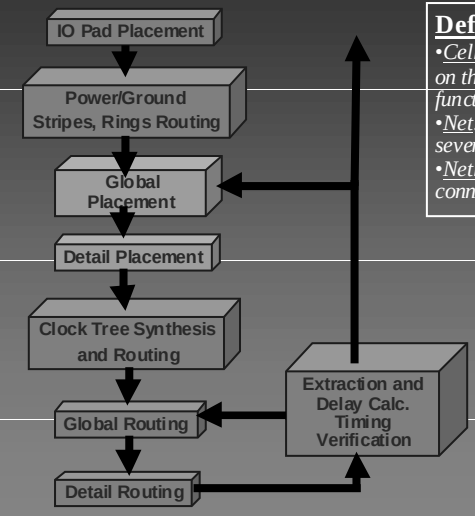
Analytical Minimization of Half-Perimeter Wirelength

Andrew Kennings, Cypress Semicondutor
Igor Markov, UCLA
imarkov@cs.ucla.edu

Outline

- VLSI Circuit Placement
- Analytical Placement
  - reduction to graphs
  - quadratic and Linear wirelength minimization
  - β-regularization
- Our contribution
  - p-regularization
  - experimental validation
- Conclusions and ongoing work

VLSI Design Flow and Physical Design Stage



Definitions:

- *Cell: a circuit component to be placed on the chip area. In placement, the functionality of the component is ignored.*
- *Net: an electronic wire to connect several cells.*
- *Netlist: a set of nets which contains the connectivity information of the circuit.*

© Andrew B. Kahng
Majid Sarrafzadeh

VLSI Circuit Placement

- Very large circuits, limited CPU time
- Need to place many times
  - to analyze various high-level "what if" scenarios
- Interconnect delay dominates device delay
- Improving wirelength ***still*** improves area
  - often used as first step in timing-driven placement
- Wirelength represented by convex functions
  - allows one to apply mathematical techniques

Global Placement

- Context
  - timing- and routability-driven placement of 10^5 cells and up
- Formalization
  - weighted hypergraph represents signal nets
  - cell shapes ignored; cells can overlap
  - constrained vertex locations, e.g., I/O pads
  - minimize objective function of unknown vertex locations
- Cell areas must be “distributed uniformly”

Hypergraph Placement

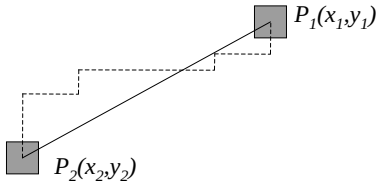
- Half-perimeter Bounding Box WL (HPWL)
  - most common convex wirelength objective
  - $\Sigma_{\text{ for all nets}}$ (*x-span* + *y-span*)
    - $x\text{-span} = max(x_1,x_2,\ldots,x_n) - min(x_1,x_2,\ldots,x_n)$
  - closely under-estimates Steiner tree wirelength
    - exact for 2-pin and 3-pin nets, ok for 4-pin
- The formulation *requires* fixed vertices
- Greedy minimization quickly stalls
  - vertices need to be moved in groups

Analytical Minimizations

- ***Decompose multi-pin nets into cliques/stars***
  - cannot handle large nets directly
  - half-perimeter WL differs for nets with 3+ pins
  - $\Rightarrow$ solution quality losses in placement
- ***Quadratic wirelength***
  - easy to minimize (need to solve two linear systems)
- ***Linear wirelength***
  - same as HPWL for 2-pin nets
  - can be minimized by LP or network flows (too slow)
  - better addresses congestion
- ***Signal delay represented with non-linear terms***
  - hard to handle by quadratic or linear minimization

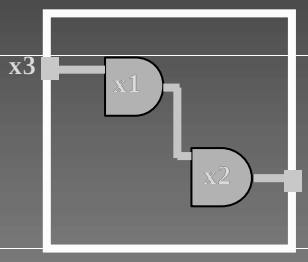
Objective Functions

- Total weighted "wirelength" of all edges
- Given an edge from $P_1(x_1,y_1)$ to $P_2(x_2,y_2)$
  - Linear (Manhattan) wirelength is $w_{12}(|x_1-x_2|+|y_1-y_2|)$
  - Squared (Euclidean) wirelength is $w_{12}((x_1-x_2)^2+(y_1-y_2)^2)$



Quadratic Placement

☞ Analytical Technique



$Min\ [(x1-x3)^2 + (x1-x2)^2 + (x2-x4)^2] : F$

$\delta F/\delta x1 = 0;$

$\delta F/\delta x2 = 0;$

$Ax = B$

$$A = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \quad B = \begin{bmatrix} x3 \\ x4 \end{bmatrix} \quad x = \begin{bmatrix} x1 \\ x2 \end{bmatrix}$$

© Andrew B. Kahng Majid Sarrafzadeh

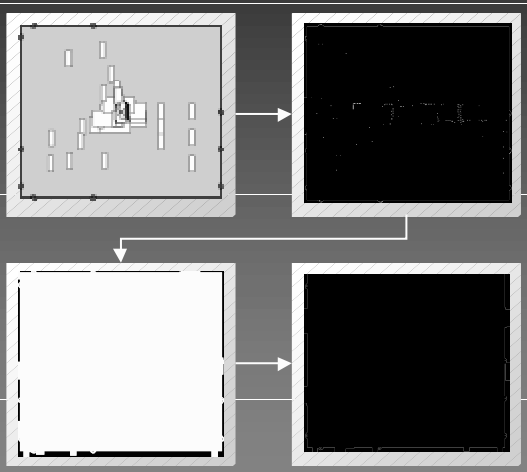
GORDIAN (quadratic + partitioning)

Initial Placement

$min\{\sum (x_i - x_j)^2\}$

$min\{\sum (y_i - y_j)^2\}$

Partition and Replace



© Andrew B. Kahng Majid Sarrafzadeh

GORDIAN-L

- Extension of GORDIAN
  - Sigl, Dohl and Johaness 1991
  - minimizes linear wirelength instead of quadratic WL
  - achieves better results
- *The really tricky part is minimizing linear WL*
  - approximate linear WL with quadratic
  - apply one step of GORDIAN, then change weights

Weiszfeld

- Alpert et al., ISPD `97
- Subsumes GORDIAN-L's linear WL minimization
- Based on a *1937* one-point algorithm by Wieszfeld
- Approximates $|x|$ with $(x^2+\beta)^{1/2}$ (with small $\beta > 0$)
- Has provable and actual linear convergence (i.e., slow)
- Also proposes faster Primal-Newton method
  - provable quadratic convergence
  - faster in practice
- Conclusion: *the really tricky part is approximating WL*
  - **once have a smooth function, can apply a lot of math**

β-regularization For Signal Delay

- Paper by Baldick et al., ASPDAC `98
- Extends *β-regularization*
  - to convex univariate PL functions
    - current journal submission - to multivariate functions
  - to a class of non-linear delay terms (proves convexity)
- Can add delay terms to regularized HPWL terms
- Advocates
  - application of existing leading numerical methods to convex smooth approximations of HPWL and delay

Remaining Problems

- Analytical methods have been inferior to partitioning-based methods as they
  - cannot address nets with 3+ pins
  - are much slower than partitioning-based methods
- Analytical techniques in this work
  - handle multi-pin nets
  - are as fast as several starts of MLFM
  - use standard Newton-based minimization engine

Addressing Multi-pin Nets

- Given a net with *n* cells, we will approximate *x-span* $= max(x_1,x_2,\ldots,x_n) - min(x_1,x_2,\ldots,x_n)$ by a smooth function
- Attempt at using *β-regularization*
  - $max(a,b,c)=max(a,max(b,c))$ (etc. for more args)
  - $max(a,b)=(a+b+|a-b|)/2 \approx (a+b+((a-b)^2+\beta)^{1/2})/2$
  - not a symmetric function of cell locations
  - recursive gradient computation difficult numerically
  - leads to numerical problems

Addressing Multi-pin Nets (cont'd)

- Rewrite *x-span* $= max(x_1,x_2,\ldots,x_n) - min(x_1,x_2,\ldots,x_n)$ as $max(|x_1-x_2|,|x_1-x_3|,\ldots,|x_2-x_3|,..|x_{n-1}-x_n|)$
- The latter expression is L_∞*-norm*
- Can be approximated by *Lp-norms*, with $p \rightarrow \infty$
  - provides a provable over-estimation
  - *Lp-norms* monotonically decrease as $p \rightarrow \infty$
  - each approximation term is strictly convex and infinitely diffentiable except at 0 (can fix by *β-reg.)*
- Our approximation of *x-span* (for even $p>0$) $((x_1-x_2)^p,(x_1-x_3)^p,\ldots,(x_2-x_3)^p,..(x_{n-1}-x_n)^p+\beta)^{1/p}$, $p \rightarrow \infty$

Notes On p-regularization

- *p*-regularization not needed for 2- and 3-pin nets
  - can use *β-regularization* for $|x_i-x_j|$ and $max(|x_1-x_2|,|x_1-x_3|,|x_2-x_3|)=((|x_1-x_2|+|x_1-x_3|+|x_2-x_3|)/2$
- *p-regularization* includes $O(k^2)$-summation for *k-pin* nets (summation over clique edges)
  - we modified it into a "star"-summation for $k>6$
  - the two Σ's are equal for $p=2$ (GODIAN-L paper)
  - quality loss for larger *p* is negligible (we compared)
    - note: there are very relatively few nets with 6+ pins

Notes On p-regularization (cont'd)

- Strict convexity proof is long (not in proc.)
- Values of *p* are bounded from above by
  - exponentiation overhead
  - mantissa overflows (loss of precision)
- $p=2^k$ are numerically convenient
  - *log(p)*-time exponentiation
  - we used $p=16$ (as good quality as for $p=64$)
  - define: sqr(z)=z*z; pow16(z)=sqr(sqr(sqr(sqr(z))));

Our Implementation: BoxPlace

- Used LBFGS software (converted Fortran to C)
  - implements limited-memory Newton method
    - fast
    - no need to explicitely compute Hessian
  - uses More-Thuente line search (from MINPACK)
    - not as precise as convex binary search, but much faster
- Our code is in C++, compiled with Sun CC4.2
- Solaris 2.7, Ultra-10 @300MHz
- Also compiles on Linux with g++ 2.95.2

Experimental Results

- Have not assembled complete placement flow
  - lin. WL minimizer as component of GORDIAN-L
- Compared different linear WL minimizations
  - quadratic
  - Weiszfeld
  - Linear Programming (LPSolve and CPLEX 6.5.1)
  - network-flow-based (Hur and Lillis, DAC `99)
- Recorded CPU time and solution quality
  - conversion time ignored for CPLEX and Hur-Lillis

Improvements to BoxPlace

- Initial solution generator
  - random initial solutions took long time to improve
  - initially we put all movable vertices into one point
  - the point was determined optimally (1-Weber problem)
- Greedy HPWL improvement passes
  - very fast
  - need no regularization (minimize true HPWL)
  - can be applied after Newton-based minimization
  - typically improve HPWL by 1-2%
  - useless before or during Newton-based minimization

Benchmarks

- Circuit hypergraphs with some vertices fixed
- Test1 and Test2 are complete circuits (LEF/DEF)
- Test3 through Test5
  - are produced from ISPD `98 (IBM) benchmarks
  - final placements are used to select “subregions”
  - each subregion gives a smaller placement problem
  - every vertex beyond the “subregion” is fixed

Instance	Modules		Nets	Design size
	Fixed	Free		
Test1	76	200	242	276
Test2	545	2686	2840	3.2K
Test3	2155	6739	7330	12K
Test4	2191	3205	3835	12K
Test5	6545	17380	20902	69K

Optimal Solvers

- All solvers achieve the same solution quality
- We record CPU time
- Hur-Lillis DAC `99 is the fastest optimal solver

HEURISTIC PLACEMENT ALGORITHMS FOR HALF–PERIMETER WIRELENGTH OBJECTIVE											
	BoxPlace from random			BoxPlace from "one–point"				Quadratic		Weiszfeld[1]	
	Initial	Final WL	CPU	Initial	Final WL	+greed x2	CPU	WL	CPU	WL	CPU
Test1	5.73e7	6.38e6	0.4	6.72e6	6.39e6	**6.23e6**	0.2	7.66e6	0.11	6.72e6	0.15
Test2	3.08e8	3.2567	24.4	3.51e7	3.11e7	**3.01e7**	15.0	4.20e7	3.75	3.51e7	7.12
Test3	6.27e6	2.36e6	28.5	2.69e6	2.32e6	**2.21e6**	13.8	2.75e6	19.9	2.67e6	22.4
Test4	5.37e6	1.49e6	13.1	1.91e6	1.44e6	**1.38e6**	5.8	1.61e6	5.4	1.78e6	12.9
Test5	3.75e7	2.55e7	45.1	1.50e7	1.40e7	**1.33e7**	33.8	1.50e7	72.1	1.47e7	87.6

Heuristic Solvers

- Computed Half-Perimeter Wirelength
- BoxPlace: good on larger testcases
  - outperforms Quadratic and Weiszfeld
  - faster than CPLEX and Hur-Lillis
- At most 12% away from optimal soln. quality

OPTIMAL PLACEMENT ALGORITHMS WITH HALF PERIMITER WIRELENGTH									
	LINEAR PROGRAM			OPTIMAL WL(x+y)	LPSolve 2.3w CPU	CPLEX 6.5.1 CPU			HUR-LIILIS [6] CPU
ROWS	COLS	NON 0s				PRIM OPT	NET OPT	TRAN OPT	
Test1	1050	512	2250	5.93e6	1.2	0.54	0.34	0.32	X
Test2	15602	6602	32500	2.75e7	6 MIN	31	9.11	10.74	X
Test3	21276	9046	45563	1.98e6	8 HR	30 MIN	2 MIN	2.2 NIN	6.58
Test4	43486	17598	93860	124e6	1 HR	72	18.44	21.97	48.82
Test5	123648	47438	265750	1.19e7	>3 DAYS	12 HR	12 MIN	15 MIN	3.3 MIN

Conclusions and Future Work

- ***BoxPlace:*** New analytical minimization of HPWL
  - solutions within 12% of optimal quality
  - does not need to represent multi-pin nets by edges
  - outperforms quadratic placement and Weiszfeld
  - faster than CPLEX 6.5.1 and Hur-Lillis
  - as fast as 5-8 starts of a multi-level FM partitioner
    - we used only one start of BoxPlace (<12% subopt. seems okay)
  - **non-linear delay terms can be added** (see ASPDAC '98)
- Future work
  - incorporating BoxPlace into a complete placement flow