FastPlace3.0: A Fast Multilevel Quadratic Placement Algorithm with Placement congestion Control

Natarajan Viswanathan
Min Pan
Chris Chu

Iowa State University

ASP-DAC 2007

Work supported by SRC under Task ID: 1206
and NSF under grant CCF-0540998

Motivation

- Current designs have millions of modules to placed
- Placement needs to be run repeatedly during various stages of the physical synthesis design flow
- Mixed-size placement - varied sizes of modules complicates the placement step
- IP blocks etc. in the form of placement blockages / fixed-macros
- A high amount of free space for routing, buffer insertion, gate sizing etc.

Need efficient techniques to handle mixed-size placement, placement blockages, placement density constraints

Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- Experimental Results
- Conclusions

Review of FastPlace (1.0 and 2.0)

Stage 1: Global Placement

1. Cell Shifting for mixed-size designs
2. Iterative Local Refinement
3. Hybrid Net Model

Stage 2: Legalization

1. Legalize and then fix movable macros
2. Legalize standard cells

Stage 3: Detailed Placement

1. Global Swap
2. Vertical Swap
3. Local Re-ordering
4. Single-segment Clustering

Techniques in FastPlace 1.0

Techniques in FastPlace 2.0

Overview of FastPlace 3.0

- FastPlace 3.0: A Fast Multilevel Quadratic Placement Algorithm with Placement Congestion Control.
  - Mixed-size placement
  - Multilevel global placement
    - Two-phase clustering (Netlist and Physical Clustering)
  - Improved Iterative Local Refinement technique to handle
    - Placement blockages
    - Placement density constraints
  - Placement density aware standard-cell legalization

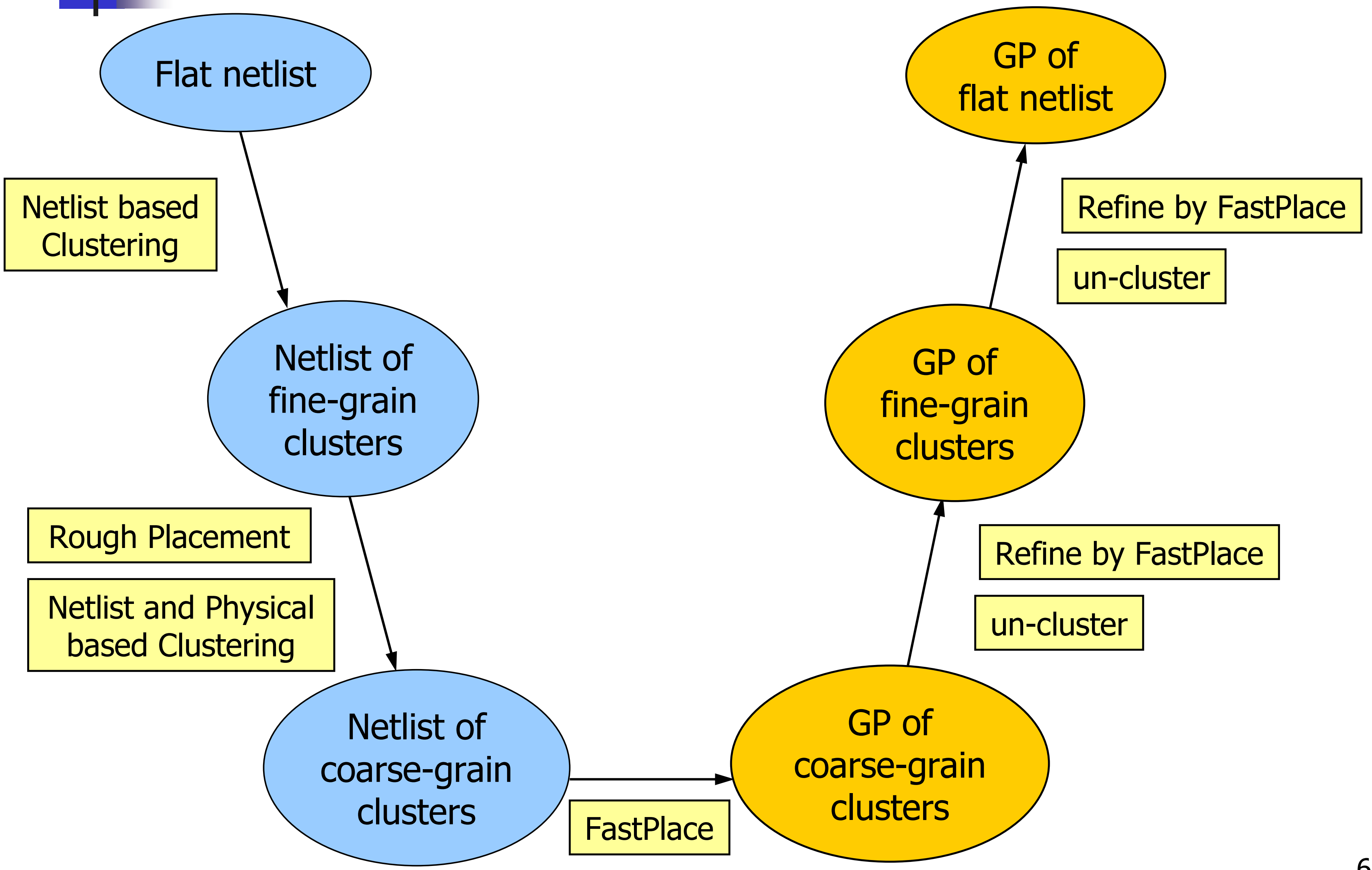
- ISPD-2005 Placement Contest Benchmarks

mPL6:	5.12x faster	2% higher HPWL
Capo10.2:	11.52x faster	9% better HPWL
APlace2.0:	16.92x faster	3% better HPWL

Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- Experimental Results
- Conclusions

Multilevel GP: Framework



Clustering for Placement

- Incorporate two-phases of clustering
  - Total 4x reduction in #objects

- Use Best-Choice Clustering with lazy-update speed-up

- Strict control on the area of clusters during both phases

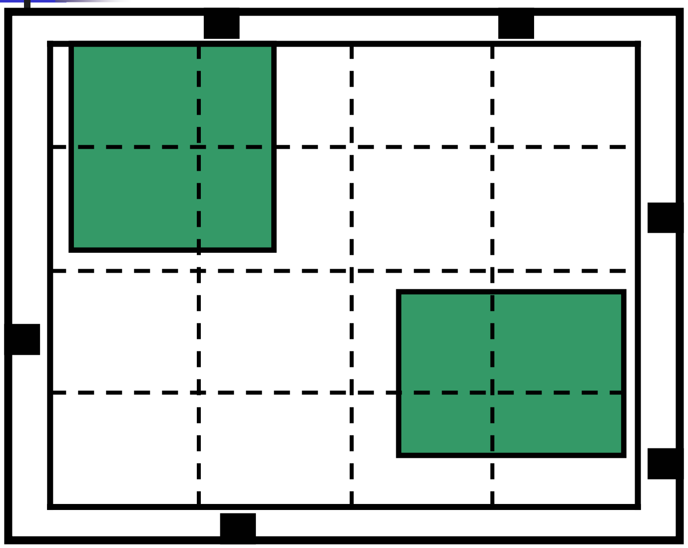
- Phase 1: Netlist-based Fine-Grain Clustering
  - clustering score $s(j,k) = \dfrac{\sum_{v \in N} w_v}{(a_j + a_k)}$ N = set of nets connecting j and k; $w_v = \dfrac{1}{|d|}$; d = degree of net v
  - only 2-3 cells of original netlist per cluster
  - 2x reduction in #objects

- Phase 2: Netlist and Physical Coarse-Grain Clustering
  - clustering score (netlist weight + satisfy distance threshold)
  - 2x reduction in #objects

Iterative Local Refinement (ILR)

- Greedy technique to simultaneously spread the cells and reduce wirelength
- Divide the placement region into bins
- Consider moving an object to the 8 neighboring bins
- Compute a score for each direction based on
  - Half-perimeter Wirelength (HPWL) reduction
  - Bin density at the source and target bins
  - Placement blockage score at the source and target bins
- Move in the direction with highest positive score
  (Do not move if no positive score)

Contour Map for Fixed-Macros



1	1	0	0
1	1	0	0
0	0	1	1
0	0	1	1

Initial Matrix

0.025	0.05	0.025
0.05	0.7	0.05
0.025	0.05	0.025

Smoothing Filter

1	1	0	0
1	1	0	0
0	0	1	1
0	0	1	1

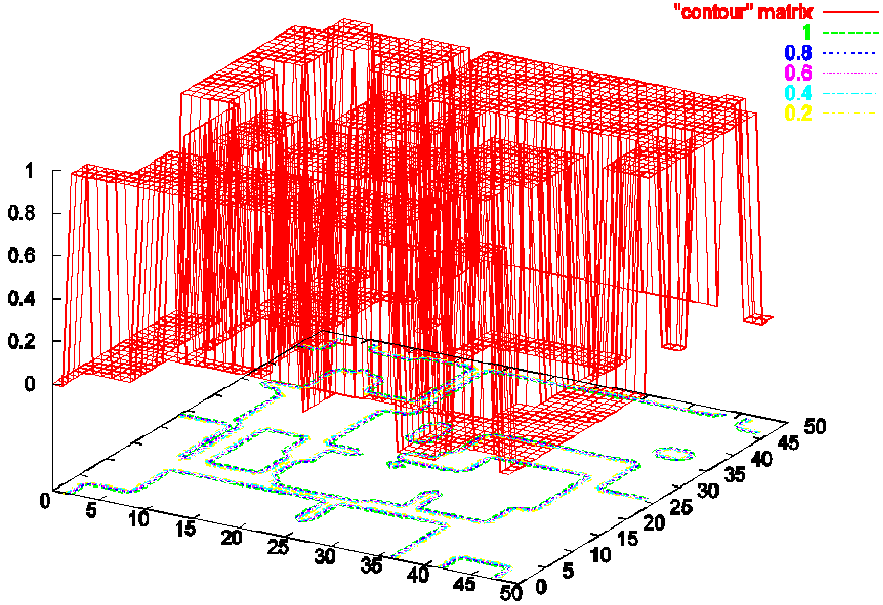
Contour Map for Fixed-Macros

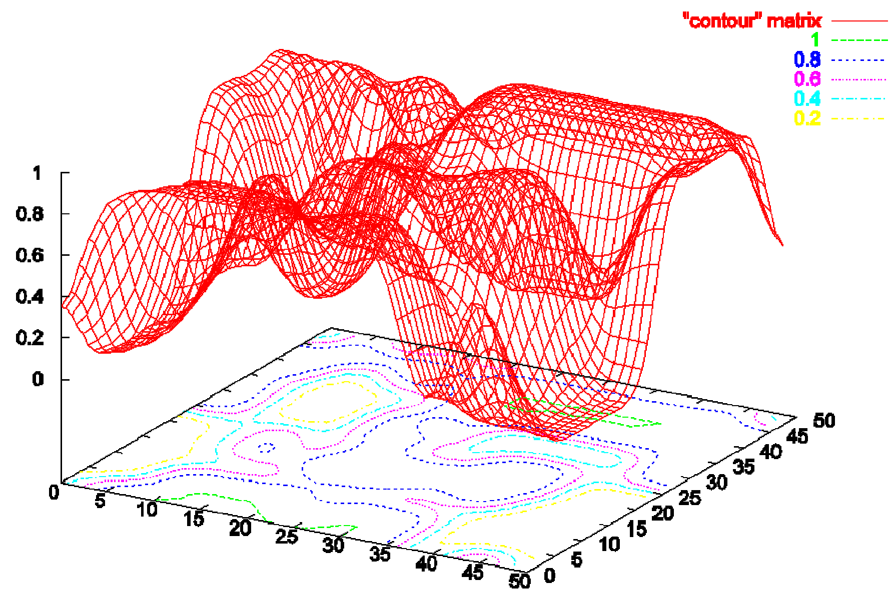
1.00	0.92	0.08	0.00
0.92	0.85	0.15	0.08
0.08	0.15	0.85	0.92
0.00	0.08	0.92	1.00

1 iteration

0.90	0.72	0.28	0.10
0.72	0.61	0.39	0.28
0.28	0.39	0.61	0.72
0.10	0.28	0.72	0.90

5 iterations





Score: Move cell i from bin m → n

α:	Weight for the wirelength component
β(m):	Weight of the utilization component for bin m
β(n):	Weight of the utilization component for bin n
γ:	Weight for the contour component
$wl_i(m)$:	Wirelength score for cell i in bin m
$wl_i(n)$:	Wirelength score for cell i in bin n
U(m):	Utilization function for bin m
U(n):	Utilization function for bin n
C(m):	Contour height at bin m
C(n):	Contour height at bin n

$$
\begin{aligned}
s_i(m,n) = \alpha(wl_i(m) - wl_i(n)) \quad &+ \\
(\beta_m U(m) - \beta_n U(n)) &+ \\
\gamma(C(m) - C(n))
\end{aligned}
$$

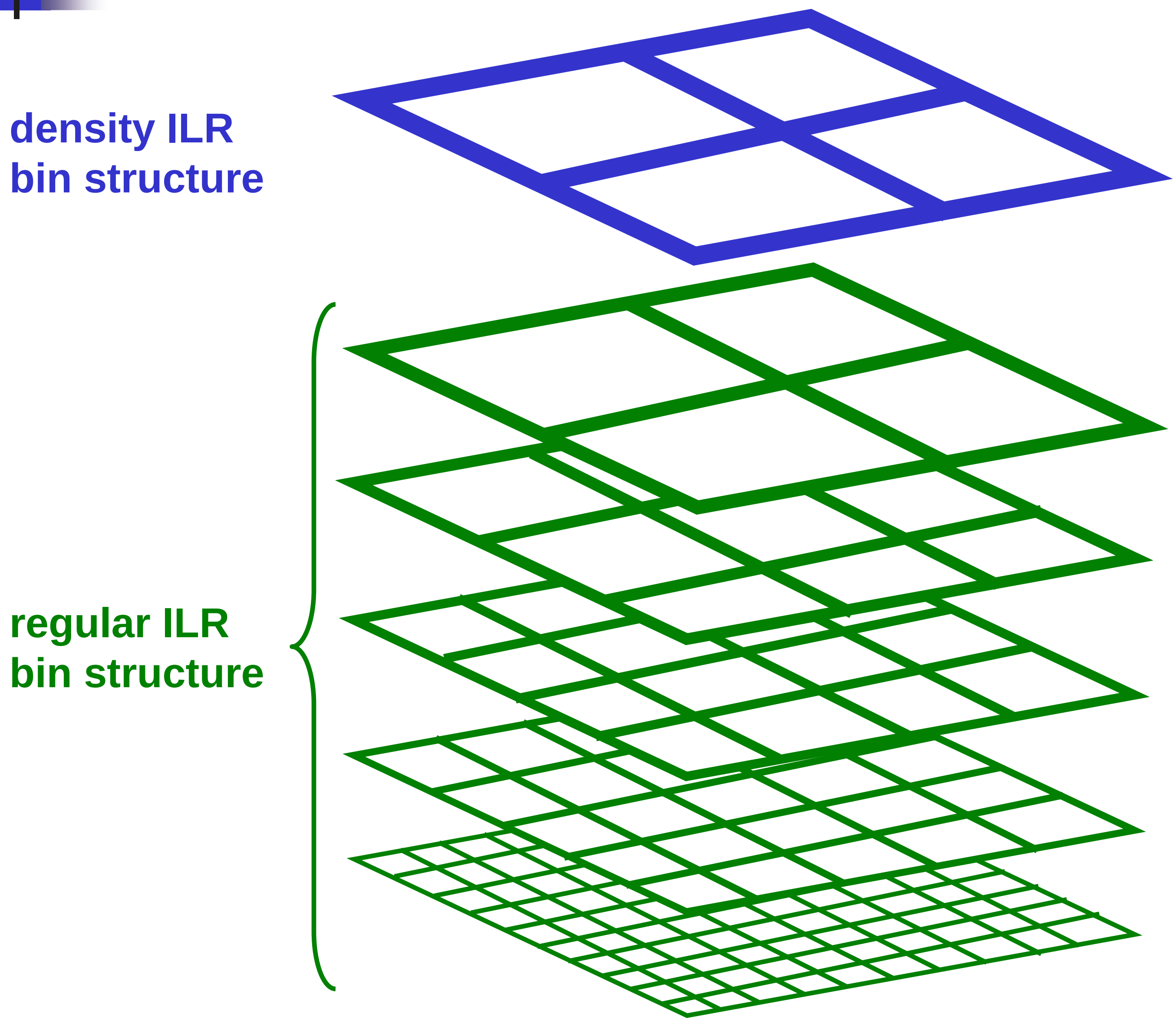
Placement Density Control

- A popular way to control routing congestion
  - Designers divide the placement region into bins
  - Specify placement density constraint for each bin
  - Run the placer until the density constraints have been satisfied
- FastPlace 3.0 uses Iterative Local Refinement (ILR) to distribute cells among bins to satisfy the density constraints
  - Similar idea used in GP, Legalization, and DP

Congestion Aware ILR

- Consists of two steps
  - d-ILR: density-bin based ILR
  - r-ILR: regular ILR
- Density-bin constructed based on user input
  - for eg: for the ISPD 06 placement contest: height and width for the d-ILR bin structure was 10x row height
- Flow:
  - First run d-ILR (using d-ILR bin structure)
  - Run r-ILR (using progressively smaller bins)

ILR Bin-Structure



Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- Experimental Results
- Conclusions

Macro Block Legalization

- Formulated as a fixed-outline floorplanning problem to resolve overlaps with minimum perturbation
- Sequence-pair (SP) to represent a floorplan

Minimum **P**erturbation **F**loorplan **R**ealization Problem

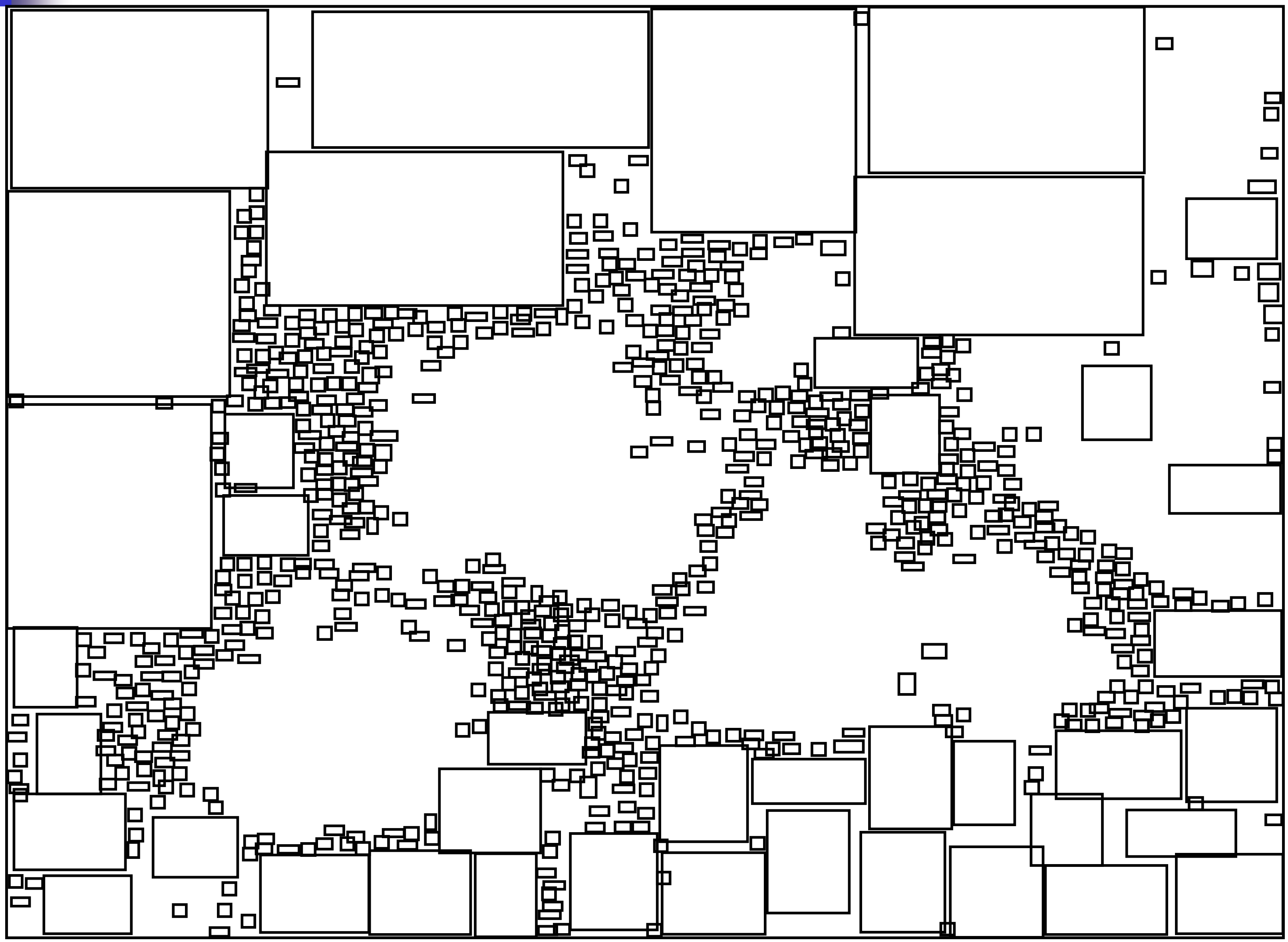
Given: n macros with target coordinates (x_i^*, y_i^*) for $i = 1, \ldots, n$ and a sequence-pair (p, q)

Determine: Legalized Coordinates (x_i, y_i) s.t. $\sum_{i=1}^{n} |x_i - x_i^*| + |y_i - y_i^*|$ is minimized.

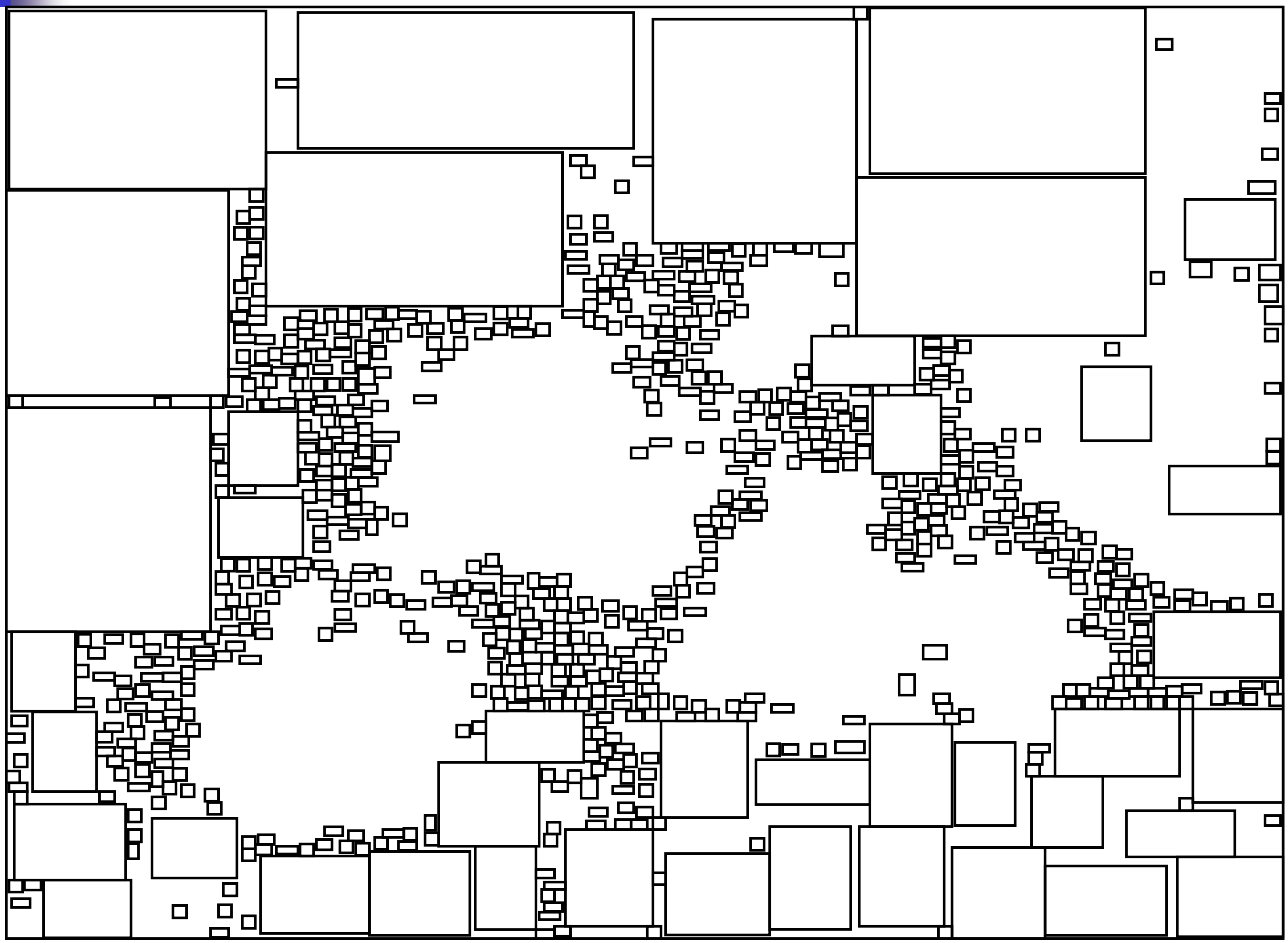
For Details:

Natarajan Viswanathan, Min Pan, and Chris Chu, "FastPlace 2.0: An Efficient Analytical Placer for Mixed-Mode Designs", pp 195 – 200, ASP-DAC 2006.

Macros Before Legalization (ibm10)



Macros After Legalization (ibm10)



Density Aware Standard-cell Legalization

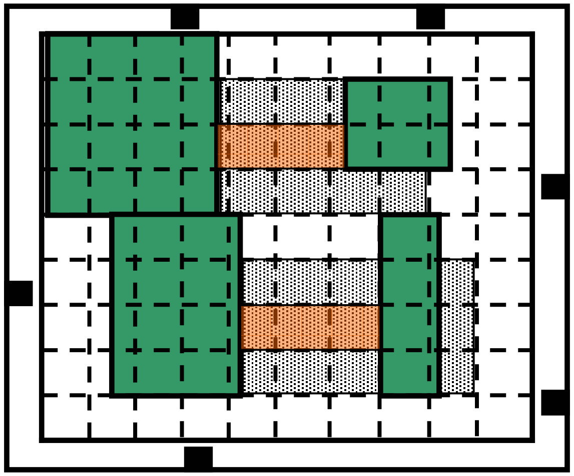
- Satisfy segment capacities
  - Selective Bin-based Standard-cell Movement
  - Segment-based Cell Rippling
  - Selective Segment-based Cell Movement (spiral)

- Legalize cells within segment

- Set-up
  - Fix movable macros if any
  - Bin the placement region
  - Fragment rows into segments based on placement blockages.
  - Find bin utilization and segment capacities

Selective Bin-based Standard-cell Movement

- Selectively turn on bin-based move around over-utilized segments
- Construct move map based on segment capacities and placement blockages
  - M(m) = 1 if bin m is around a segment with density > target_density
  - M(m) = 0 otherwise or if bin m overlaps with placement blockage



- Use similar scoring function as ILR
- Advantages:
  - Very less perturbation of the global placement
  - Distributes cells evenly within a segment (helps density constraint)
  - Fast

Segment-based cell Movement

- Segment Based Rippling
  - Consider over-utilized and neighboring segments
  - Ripple cells out of segments
  - Can increase the utilization of neighboring segments
  - Consider wirelength and utilization for score
  - In most cases decreases wirelength while satisfying segment capacities

- Selective Segment-based Cell Movement
  - Move cells to closest segment (radial search)
  - Used as last resort

Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- Experimental Results
- Conclusions

Detailed Placement Overview

Perform ***Single-Segment Clustering***

Repeat

    Perform ***Global Swap***

    Perform ***Vertical Swap***

    Perform ***Local Re-ordering***

Until no significant improvement

Repeat

    Perform ***Single-Segment Clustering***

Until no significant improvement

For Details:

Min Pan, Natarajan Viswanathan, and Chris Chu, "An Efficient and Effective Detailed Placement Algorithm", pp 48 – 55, ICCAD 2005.

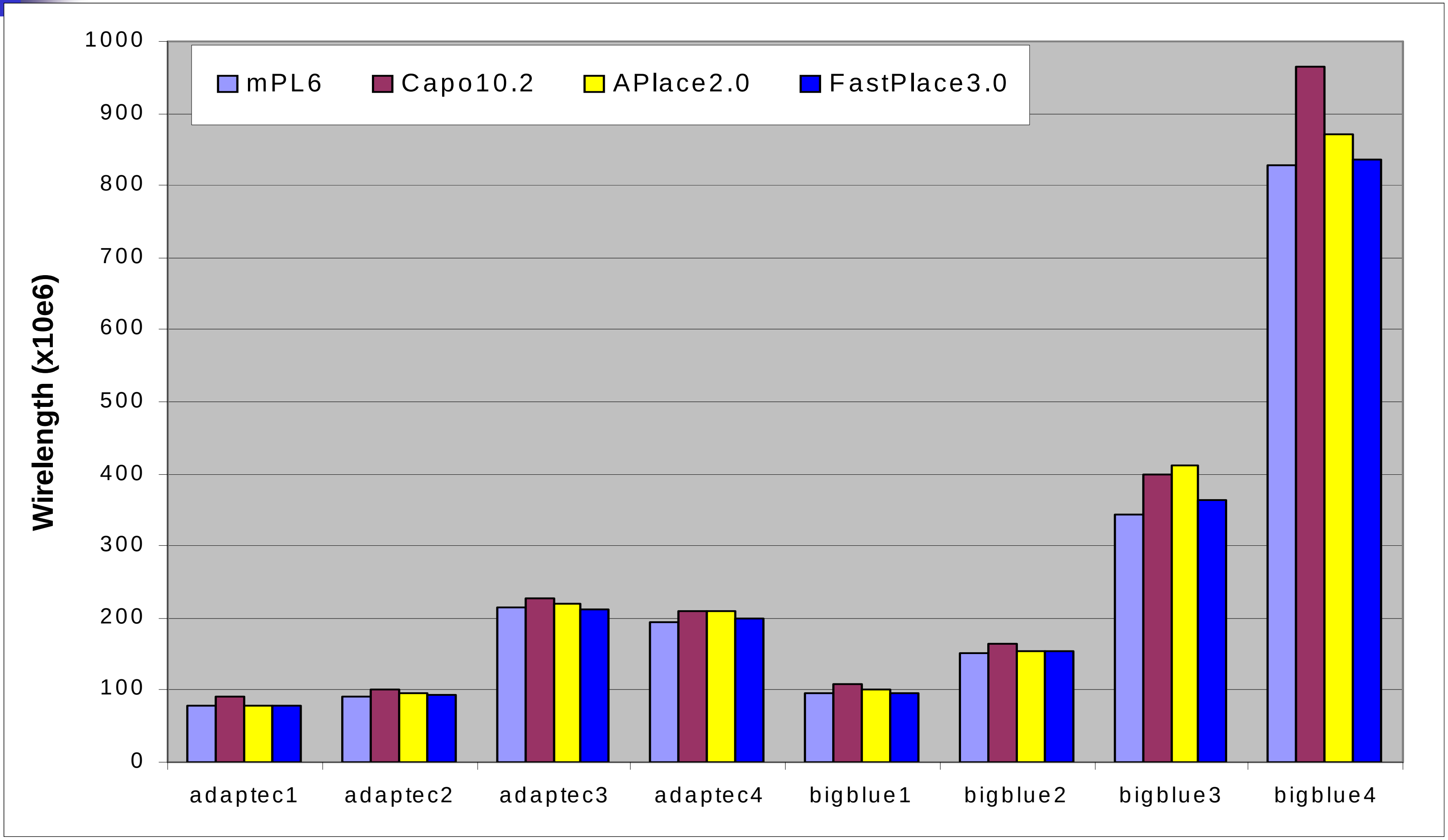
Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- **Experimental Results**
- Conclusions

Experimental Setup

- ISPD-2005 placement contest benchmarks
- ISPD-2006 placement contest benchmarks
  (with placement density target constraint)
- 210k – 2.48M movable objects
- All experiments are on a 2.5GHz AMD Opteron 250 machine with 8 GB RAM

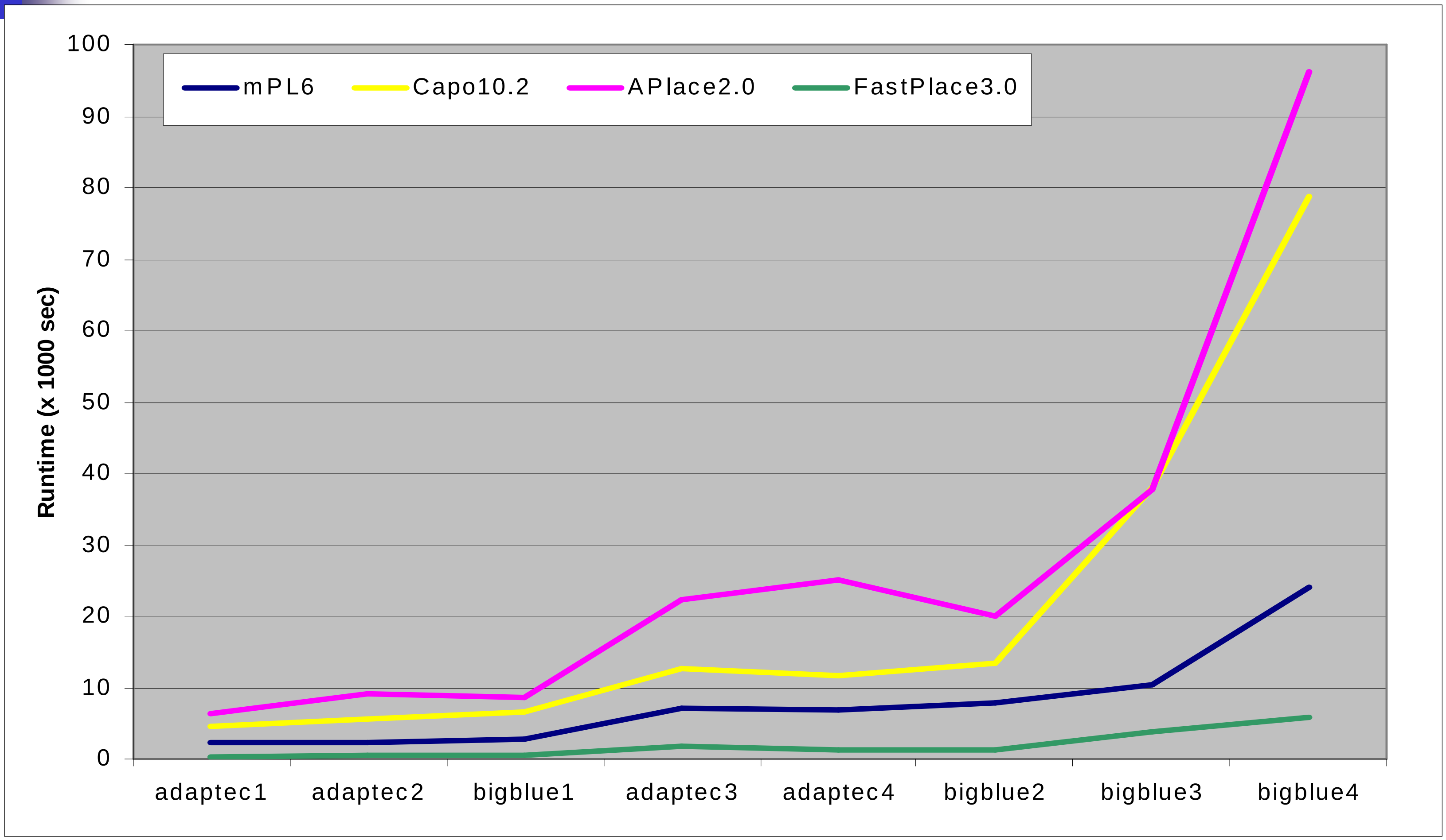
ISPD-05 Benchmarks (HPWL)



Average Wirelength Ratio:

mPL6 / FastPlace3.0 :	**0.98**
Capo10.2 / FastPlace3.0 :	**1.09**
APlace2.0 / FastPlace3.0 :	**1.03**

ISPD-05 Benchmarks (Runtime)



Average Runtime Ratio:

mPL6 / FastPlace3.0 : **5.12 X**

Capo10.2 / FastPlace3.0 : **11.52 X**

APlace2.0 / FastPlace3.0 : **16.92 X**

ISPD-05 Benchmarks (contest)

	adaptec2	adaptec4	bigblue1	bigblue2	bigblue3	bigblue4	Avg
APlace	0.94	0.93	0.99	0.93	0.94	1.00	0.955
FastPlace3.0	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
mFAR	0.98	0.95	1.02	1.09	1.00	1.05	1.015
Dragon	1.02	1.00	1.07	1.03	1.00	1.09	1.034
mPL	1.04	1.00	1.03	1.12	0.97	1.09	1.041
Capo	1.07	1.05	1.13	1.11	1.01	1.32	1.115
NTUPlace	1.08	1.03	1.11	1.23	1.08	1.39	1.153
Fengshui	1.32	1.67	1.20	1.84	1.24	1.25	1.420
Kraftwerk	1.69	1.75	1.56	2.08	1.73	1.69	1.749

HPWL Comparison with contest version of placers

No Limit on CPU time for other placers to get the best possible results

ISPD 06 contest benchmarks (1)

	a5	n1	n2	n3	n4	n5	n6	n7	Avg.
Kraftwerk	1.01	1.19	1.00	1.00	1.01	1.04	1.00	1.00	1.03
mPL6	1.00	1.06	1.07	1.17	1.00	1.02	1.00	1.00	1.04
FastPlace3.0	**1.12**	**1.15**	**0.96**	**1.09**	**0.98**	**1.11**	**0.96**	**0.93**	**1.04**
NTUPlace2	1.02	1.00	1.07	1.16	1.03	1.00	1.04	1.07	1.05
mFAR	1.09	1.23	1.09	1.16	1.09	1.13	1.03	1.04	1.11
APlace3	1.26	1.20	1.05	1.13	1.35	1.21	1.06	1.05	1.16
Dragon	1.08	1.21	1.29	1.90	1.05	1.13	1.03	1.23	1.24
DPlace	1.26	1.55	1.77	1.36	1.14	1.35	1.23	1.25	1.36
Capo	1.16	1.57	1.64	1.44	1.22	1.28	1.32	1.46	1.39

Using the ISPD 06 placement contest scoring function
(considering HPWL, Placement Density, Runtime)

ISPD 06 contest benchmarks (2)

Circuit	a5	n1	n2	n3	n4	n5	n6	n7	Avg.
#Objects	843k	330k	441k	494k	646k	1.23M	1.25M	2.51M	
FastPlace3.0 (sec)	**1973**	**609**	**816**	**1619**	**878**	**3156**	**2519**	**3279**	
Kraftwerk	1.67	1.86	1.23	0.56	3.16	2.35	2.12	2.28	1.91 x
mPL6	4.19	3.70	7.47	5.99	6.62	3.91	4.78	8.66	5.66 x
NTUPlace2	5.32	3.55	5.43	4.10	8.51	6.48	5.50	6.55	5.68 x
mFAR	3.48	4.17	3.55	1.83	7.25	3.62	4.82	5.94	4.33 x
APlace3	10.27	7.07	6.78	7.72	17.07	10.39	11.56	16.73	10.95 x
Dragon	1.14	1.62	2.00	0.72	1.69	1.12	1.53	3.02	1.61 x
DPlace	1.46	1.69	7.84	0.64	1.88	1.44	1.60	2.90	2.43 x
Capo	4.93	4.21	6.92	3.75	7.89	6.61	7.34	16.76	7.30 x

Runtime comparison

Outline of the Presentation

- Overview of FastPlace
- Congestion-aware Multilevel Global Placement
  - Clustering for Placement
  - Improved – Iterative Local Refinement (ILR)
- Legalization
  - Macro-block Legalization
  - Density Aware Standard-cell Legalization
- Detailed Placement
- Experimental Results
- Conclusions

Conclusions

- FastPlace 3.0: A Fast Multilevel Quadratic Placement Algorithm with Placement Congestion Control
  - Scalable multilevel global placement algorithm
    - Two-phase clustering (Netlist and Physical Clustering)
    - Improved Iterative Local Refinement technique to handle
      - Placement blockages
      - Placement density constraints
  - Placement density aware standard-cell legalization
  - Fast – Takes only about 1½ hours to place designs with over 2 Million objects (bigblue4, newblue7)
- Linux 32-bit and 64-bit binaries available for download at:

  http://www.public.iastate.edu/~nataraj/FastPlace.html