Fast Analytic Placement using Minimum Cost Flow

Ameya R. Agnihotri (Magma Design Automation & SUNY Binghamton, USA)

Prof. Patrick H. Madden (SUNY Binghamton, USA)

Main contributions

- New global placement algorithm, Vaastu :
  - Continuous + discrete optimization -
    - Nonlinear optimization + network flows.
  - Network flow algorithms are based on the more accurate half-perimeter wire length (HPWL) model.
    - Uncommon in current placers.

Motivation

- Nature of the placement problem has been changing.
  - Need for strong and fast placement algorithms.
- IBM released new designs at ISPD'05/06 for academic research.
  - Representative of today's circuits.
    - Large & nontrivial constraints.
- At least 10 research groups publishing work.
  - Active research going on.

Current designs – challenges

- Millions of modules
  - Vast solution space.
- Large number of fixed obstacles/macros
  - Movable modules have to avoid obstacles.
- Lot of free space a.k.a white space
  - Increases solution space, further.
  - Distributed differently in different designs.

Analytical Placer : Log-sum-exponent function

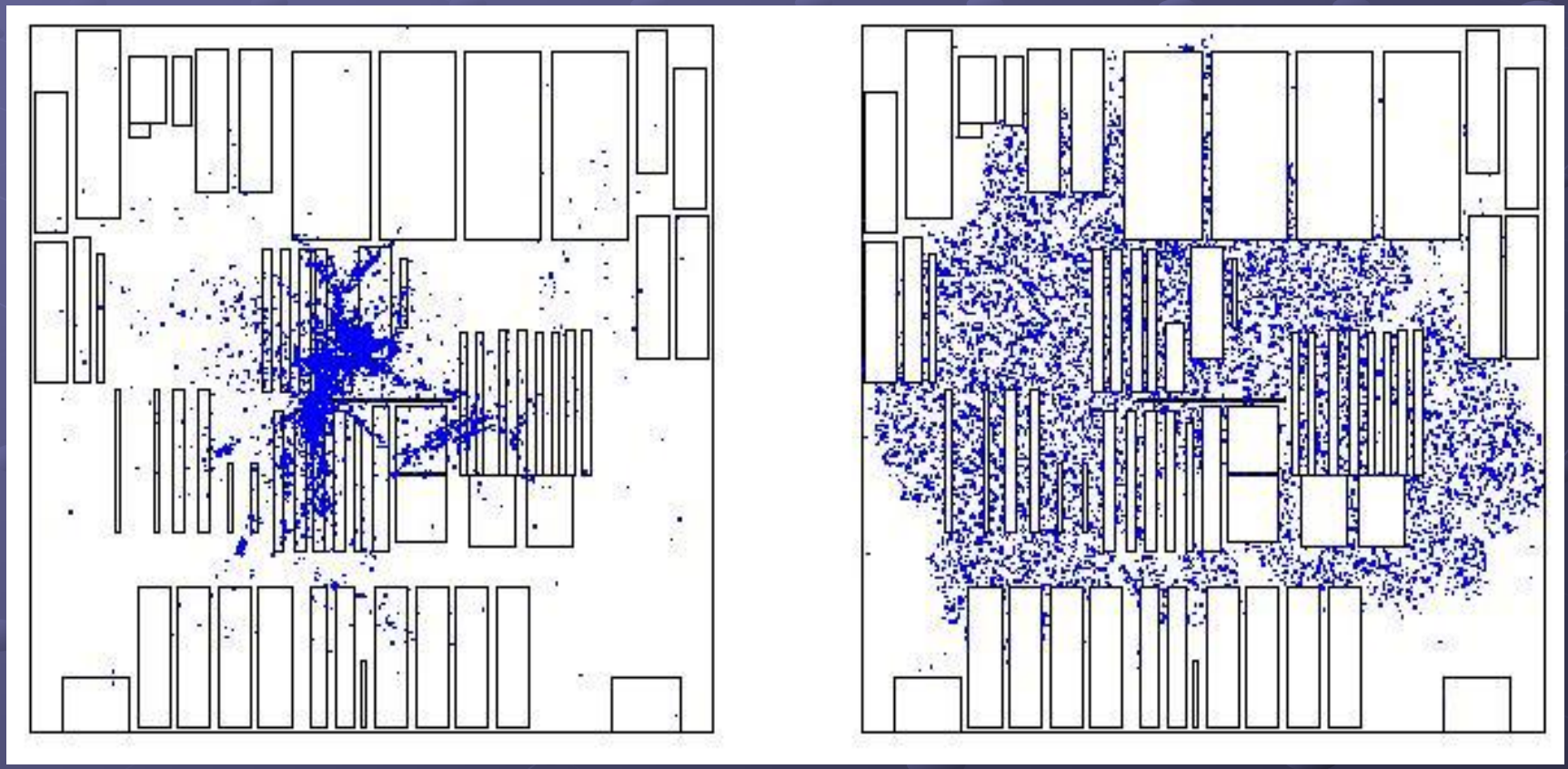
$$wl(P) = \alpha \sum_{v \in \mathcal{N}} \left(log \sum_{m_i \in v} e^{(x_i^v/\alpha)} + log \sum_{m_i \in v} e^{(-x_i^v/\alpha)} + \right.$$

$$\left. log \sum_{m_i \in v} e^{(y_i^v/\alpha)} + log \sum_{m_i \in v} e^{(-y_i^v/\alpha)} \right)$$

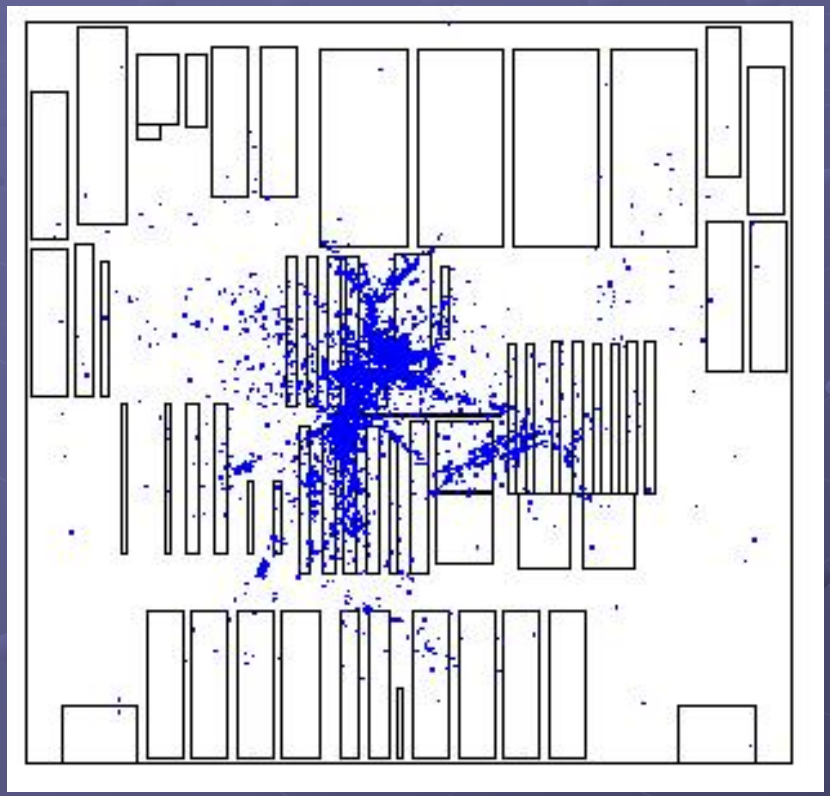
- Non-linear : Approximates linear HPWL.
- Synopsys patent.
- Placers APlace, mPL, NTUPlace3 also use it.
- Function minimizer – Analytic Solver (AS).

Analytical Placement : what's the challenge?

- Placement 1 : minima of log-sum-exponent fn.



A Novel Way to Look at the Problem



- Modules : Supply themselves.
- Regions : Demand for modules.
- Supply-demand network – Bipartite graph.
- Assignment problem – find an assignment of modules to regions.

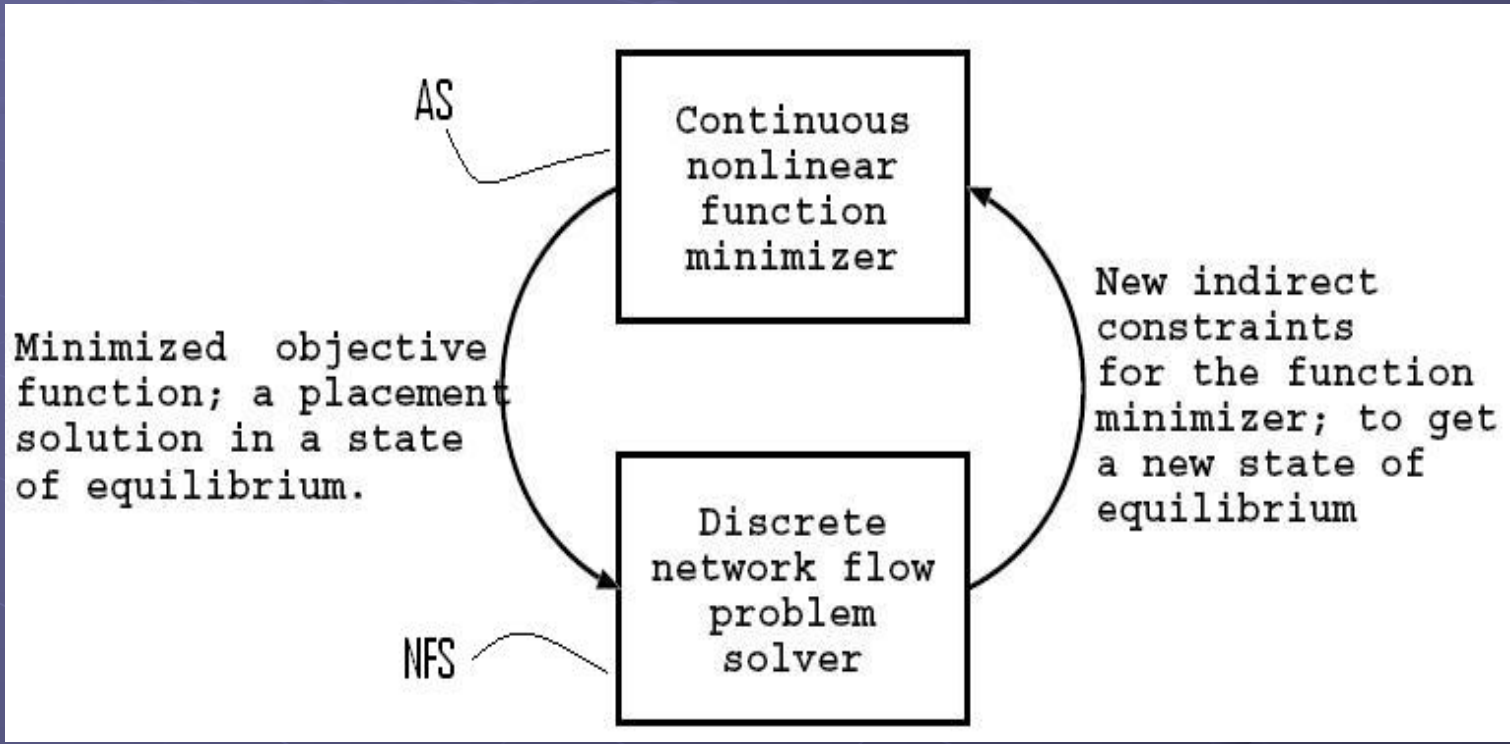
What else?

- Minimize the cost of assignment.
  - Maintain low wire length.
- Minimum cost flow (MCF) problem.
- Solved by → Network flow solver (NFS).
  - Extract a NF instance from a placement.
  - Solve it : find an assignment.

Advantage

- We are treating the problem as partly discrete.
- Modules can spread right through big obstacles.
  - We can guarantee that there is space across them.
  - We know this wont worsen HPWL.

Approach : Skeleton



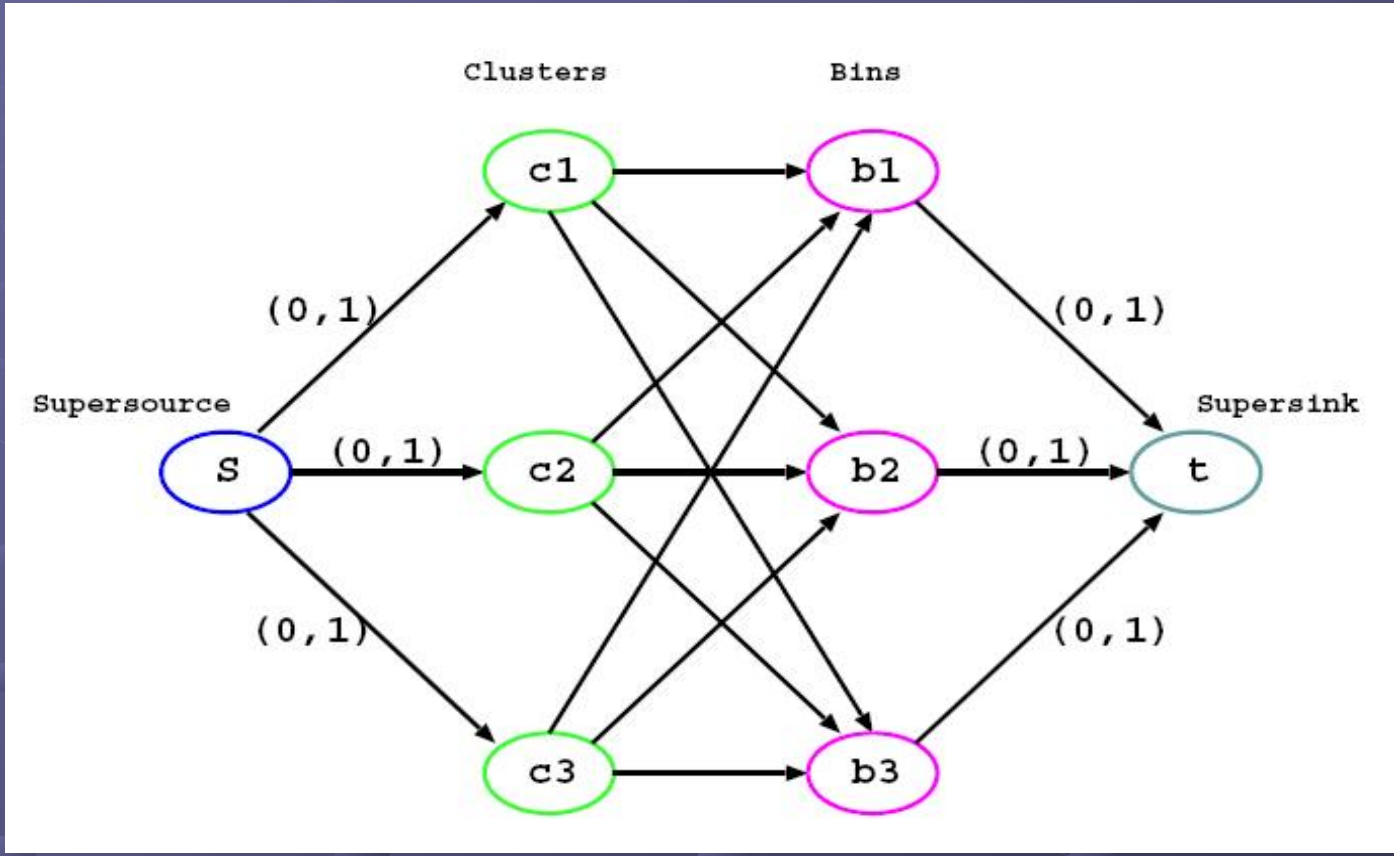
- Analytic Solver (AS) & Network Flow Solver (NFS).
- Guarantee the availability of space in the direction of spreading.

Next few slides →
Details of Network Flow Solver (NFS)

Physical Clustering & bin formation

- Clustering – necessity.
  - Otherwise : impractical to use Network Flows.
- Use ONLY geometric locations of modules to form clusters.
  - Clusters – supply nodes in graph
- Divide placement region into bins.
  - Bins – demand nodes in graph.

Minimum Cost Flow (MCF) Model



- For each edge between a cluster & a bin :
  - Cost = w(c,b), where c is the cluster, b is the bin.

Cost of assignment

- How to calculate cost(c,b) or w(c,b)?

- Let N_c denote the set of nets of cluster c.
- If c is moved to the center of b :
  - What is the gain in HPWL?
  - HP == Half-perimeter wire length.

$$gain(e) = gain(c,b) = \sum_{v \in \mathcal{N}_c} (HP(v) - HP^b(v))$$

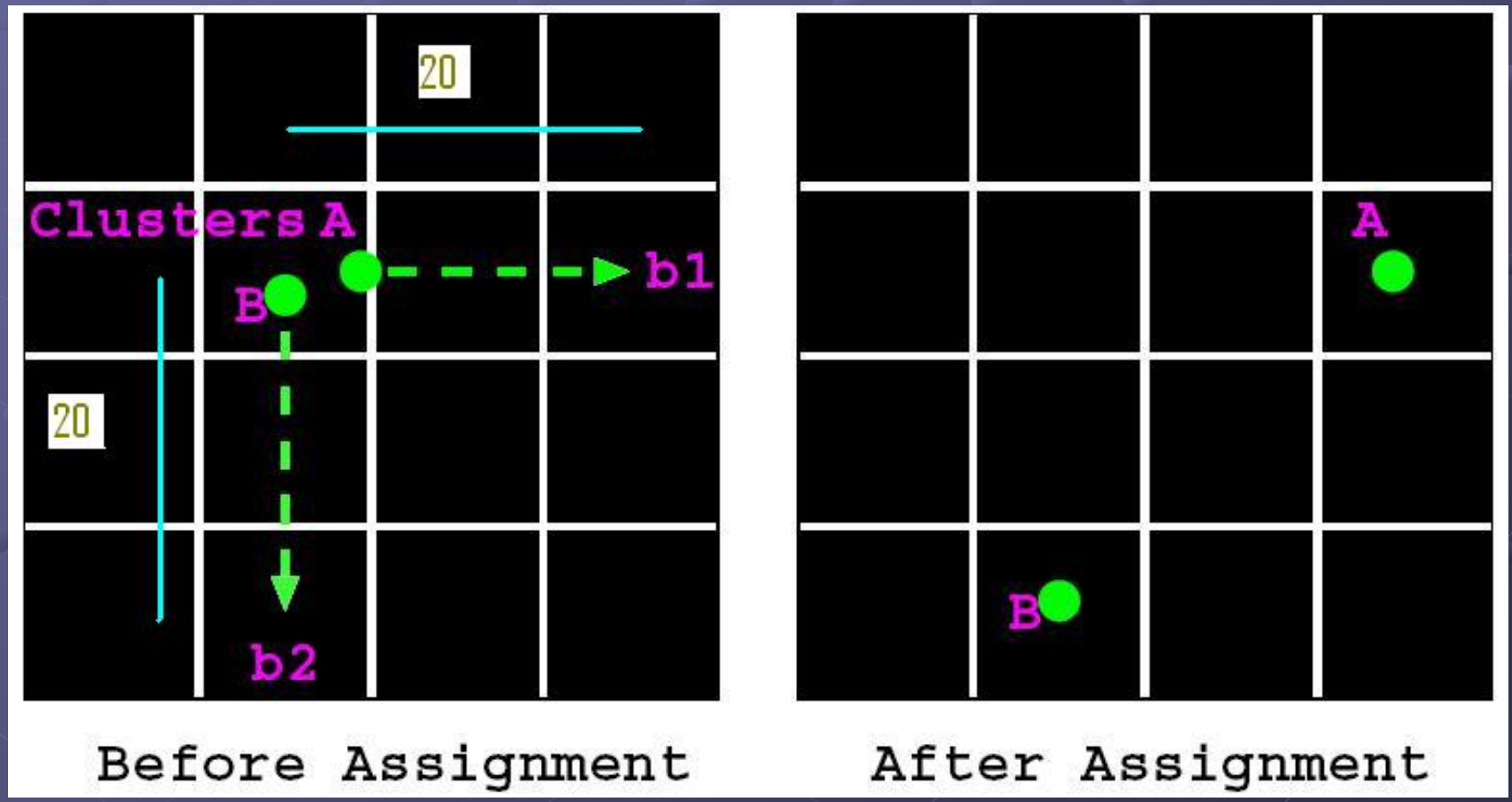
Cost of assignment – contd..

- Need to maximize the gain in HPWL for the entire assignment.
- If cost(c,b) is expressed as below, then :
  - Minimizing cost == maximizing gain.
- Minimum cost flow comes into picture.

$$w_e = w(c,b) = -gain(c,b)$$

Cost function inaccuracy

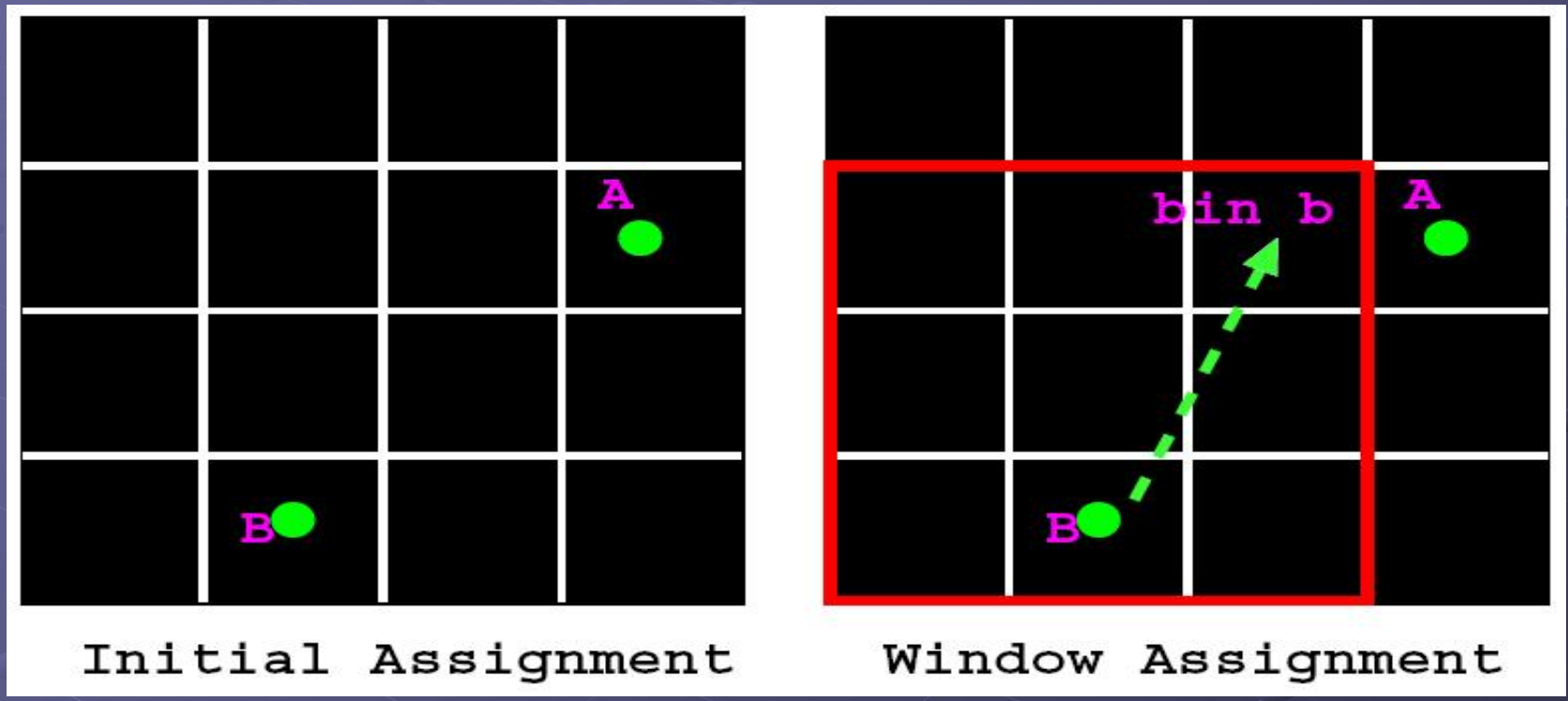
- MCF cost function is inaccurate. Why?



Sliding window MCF

- Solution? → Sliding window MCF.
- Improve the global MCF assignment.
  - Select a small window of bins.
  - Run MCF on clusters & bins in the window.
  - Slide window.
- Why will this help?

Sliding window MCF – contd..



- Cluster A is stationary – outside the window.
- Cluster B is free to move to bin b.

Sliding window MCF – contd..

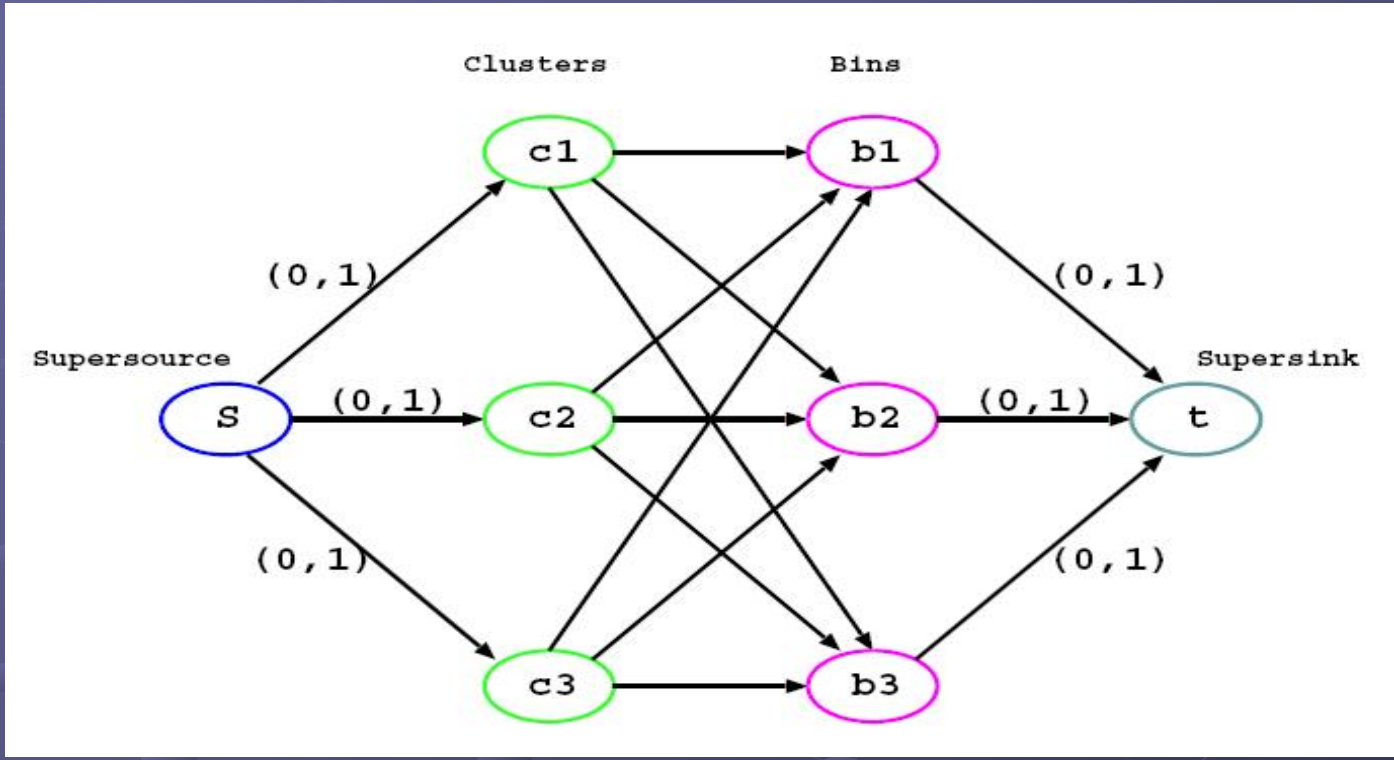
- Smaller window == higher accuracy.
- Smaller window == lesser scope for optimization.
- We use 8x8, 6x6, 4x4, 2x2 windows in each iteration.

Placement Algorithm – so far

- Run AS → Find minima of log-sum-exp.
- Initialize num_clusters; //small
- Until {size of cluster is not small}
  - Run NFS;
  - Prepare spreading forces for AS;
  - Run AS; // find new minima
  - Increase num_clusters;

Next few slides → Speeding-up Minimum Cost Flow (MCF)

MCF speed-up



- If every c is mapped against every b, number of edges will be too high, roughly $O(|V|^2)$.
- Impractical to use MCF for bigger designs.

MCF speed-up : Radius

- Map a cluster c with k closest bins.
  - k == Radius, small user defined parameter.

- Drawback – Flow feasibility:
  - There may not be a feasible flow.
  - Check flow feasibility using Maximum flow.
  - If feasible → solve MCF.
  - If not → increase radius.
  - Max flow – much faster than – MCF.

Further speed-up : Max Flow

- In last few iterations, graph size increases.. Even with the idea of radius.
  - As we are increasing num_clusters;

- Need further speed-up.
  - Switch to Maximum Flow (MF) instead of MCF (for the global problem).
    - Significantly reduces run time.
  - NOTE : Assignment will not be a minimum cost assignment anymore.
    - Use sliding window MCF to improve the assignment.

Next few slides → Interaction between AS & NFS

Anchoring

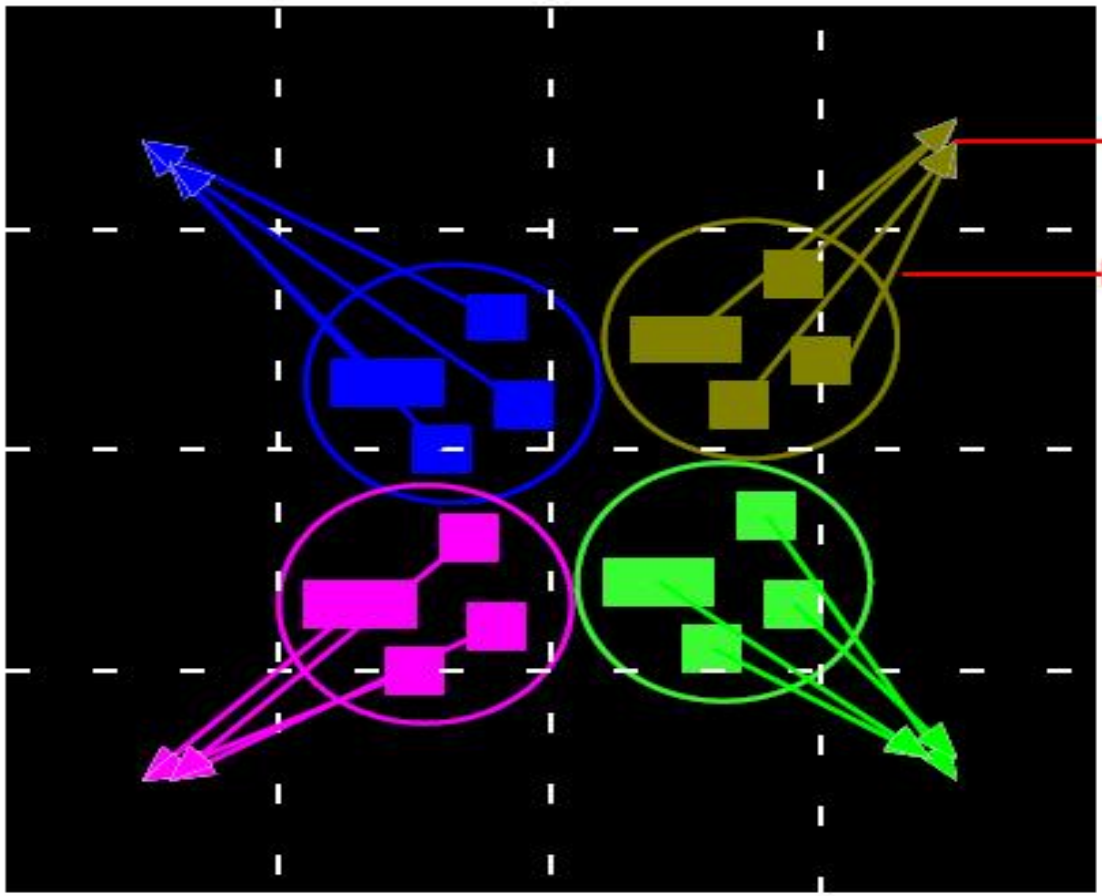
- How to use the assignment given by NFS to spread modules?

- Do not move clusters to the assigned bins.
- Instead :
  - Create an Anchor for each module at the center of the assigned bin.
  - Create a Pseudonet between each module and its anchor.
  - This is called Anchoring.
    - Many analytic placers have related ideas.
    - mFAR adds ``Fixed-points'' – similar in spirit.

Anchoring – net weights

- Pseudonet – fake net.
  - Weight < weight of original nets (fixed → 1).
  - Reason – AS should not prefer optimizing the pseudonets over original nets.

Anchoring – example



Anchoring – contd..

- Initially, assignment found by NFS is not so good (in terms of HPWL).
  - Due to large overlap.
  - Keep weight of pseudonets low.
- As modules spread..
  - Assignments get better.
  - Increase weight on pseudonets.

Limitation of Network Flows

- Limitation of NFS :
  - Can not run till the bottom level :
    - 1 node in graph = 1 module
  - Graph size will be huge.

- We need to go to bottom level →
  - With clustering →
    - All modules in a cluster have same anchor location
    - Modules in a cluster may clump at the center of bin.

Cut line shifting (CLS)

- In last few iterations, use CLS (Li etal ICCAD'04, ASPDAC'05).
  - Very fast. Completely geometric.
  - Need to be careful :
    - Only use it, when modules have spread reasonably.
    - Else, severe WL increase.
- Details not discussed here.

Global Placement Algorithm

- Run AS → Find minima of log-sum-exp.
- Initialize num_clusters, pseudonet_weight, radius;
- Until {size of cluster is not small}
  - Run NFS/CLS;
  - Prepare spreading forces for AS;
  - Run AS; // find new minima/placement
  - Increase num_clusters, pseudonet_weight;
  - Decrease radius;

Legalization & Detailed Placement

- After global placement, we use FastDP algorithm (Pan etal, ICCAD'05) to legalize and further optimize the placements.
  - Prof. Chris Chu's group.

Experimental Results

ISPD'05 placement benchmarks

Benchmark	#movable	#fixed	#nets	Util.(%)
adaptec1	211k	543	221k	57.34
adaptec2	254k	566	266k	44.32
adaptec3	451k	723	467k	33.52
adaptec4	495k	1329	516k	27.14
bigblue1	278k	560	284k	44.67
bigblue2	535k	23084	577k	37.78
bigblue3	1096k	1293	1123k	56.48
bigblue4	2169k	8170	2230k	44.29

TABLE I
BENCHMARK CHARACTERISTICS

HPWL comparison – Subset of Results

Circuit	Vaastu	APlace ISPD05	KraftWerk ICCAD06	NTUPlace3 ICCAD06
adaptec2	92.97	87.31	95.91	89.85
adaptec4	192.06	187.65	202.9	193.74
bigblue1	98.09	94.64	101.19	97.28
bigblue2	153.43	143.82	157.53	152.20
bigblue3	370.72	357.89	346.01	348.48
bigblue4	828.25	833.21	869.75	829.16
Average	**1.037**	**1.00**	**1.059**	**1.019**

- Better than all the other published results.

Results Summary

- HPWL
  - Best reported WL on the biggest ISPD'05 design (2million modules).
- Run time –
  - Ours – 9.8 hours – for all 8 designs.
  - APlace (ISPD'05) – 113 hours – for 6/8 designs.
  - Much faster than APlace, competitive with NTUPlace3 and latest Kraftwerk.

Summary

- Fast & strong placement algorithm for today's circuits – Vaastu.

- Novel features :
  - Treat the problem as [partly discrete + partly continuous].
    - Nonlinear optimization + network flows
  - Network flow – based on the more accurate HPWL model.
  - Effective methods for speeding the algorithm presented.
  - Fast, state of the art results.

Thank you!

- Thanks to :
  - Prof. Chris Chu's group for providing FastDP.
  - Prof. Cheng-Kok Koh and Dr. Chen Li for useful discussions related to the Analytic Solver.
  - Dr. Gi-Joon Nam for organizing ISPD contests, and to the groups who participated.

- Latest results to be reported at ISPD'07 Placement Contest.