TCG-based Multi-bend Bus-driven Floorplanning

Tilen Ma and Evangeline Young
Dept. of Computer Science and Engr.
Chinese University of Hong Kong

Motivations

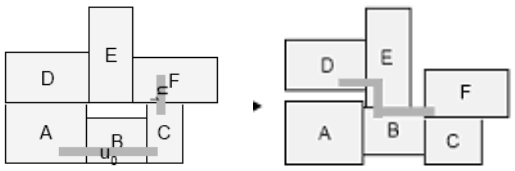
- As technology advances, amount of interconnections between different modules increases.
- Bus routing becomes more and more important.
- Essential to consider bus routing in the floorplanning stage.

Bus-driven Floorplanning (BDF)

- In BDF, we are given:
  - A netlist of modules
  - A set of bus, each of which has a width and a set of modules to pass through
- Our objective is to produce a floorplan such that all buses can go through its modules without overlapping, minimizing the floorplan and bus areas.

Previous Works

- Buses with no bending:
  - Xiang *et al.* at ICCAD 03
  - Chen *et al.* at ISPD 05
- Zero, One or Two bending:
  - Law *et al.* at ISPD 05



Comparison w/Previous Works

- Previous works restrict the bus shapes:
  - Require complex and inefficient shape validation steps.
  - Unable to handle buses with large bus nets.
- Our approach:
  - Without restricting the bus shapes.
  - Reduce the number of vias.

Problem Formulation

- Given:
  - A set M of n rectangular modules $\{m_1,...,m_n\}$ with each m_i associated with an area a_i and an aspect ratio bound $[r_i, s_i]$ where $r_i, s_i \in R^+$.
  - A set B of k buses $\{b_1,...,b_k\}$ with each b_j associated with a width t_j and a bus net N_j, where $t_i \in R^+$ and $N_i \subseteq M$.

Problem Formulation

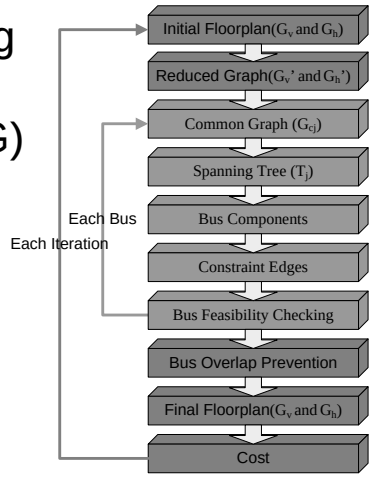
- Constraints:
  - All buses can pass through all the modules in its net.
  - Two layers for bus routing, so no overlap btw. horizontal (vertical) bus components.
  - All bending of the buses must occur at the modules on the corresponding bus nets (for via number minimization).

Problem Formulation

- Objective:
  - Give a floorplan of the circuit with the routes of the buses satisfying the constraints and minimizing a weighted sum of the total chip area and the total bus area.

Methodology

- Simulated annealing with Transitive Closure Graph (TCG) representation is used.



Methodology

- In each iteration of the annealing, for each bus:
  - Construct reduced constraint graphs, G_h' and G_v', from TCG, G_h and G_v.
  - Construct *common graph* from G_h' and G_v'.
  - Find spanning tree T in common graph.
  - Find bus components from T.
  - Add constraint edges to G_h and G_v to enforce alignment between modules.

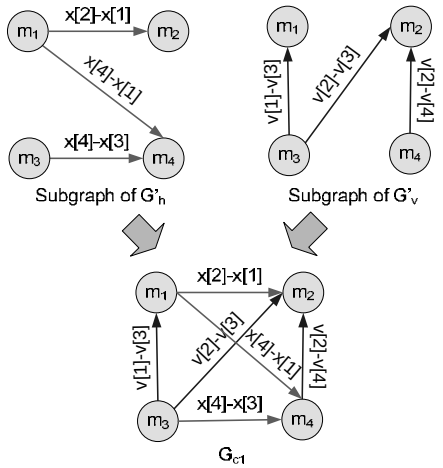
Reduced Constraint Graphs

- Given TCG $G_h(V,E_h)$, construct $G_h'(V',E_h')$ as follows:
  - $V' = \cup N_j$ for all $1 \leqq j \leqq k$
  - $e_{ij} \in E_h'$ iff $e_{ij} \in E_h$ and $m_i, m_j \in V'$
- Weights on the edges are the actual distances between them in a realized floorplan without buses.
- Similarly for the vertical direction.

Common Graphs

- We will construct a common graph $G_{cj}(V_j,E_j)$ for each bus b_j from G_h' and G_v':
  - $V_j = N_j$ (the set of modules on bus b_j)
  - $E_j = \{e(a,b) \mid e(a,b) \in G_v' \cup G_h'$ and $x,y \in N_j\}$
  - Weights on the edges are the weights of the corresponding edges in G_v' or G_h'. Note that an edge between two modules exists in either G_v' or G_h' since G_v' or G_h' are sub-graphs of TCG.

Common Graphs



Finding MST

- A multi-bend bus b_j can be regarded as a spanning tree in G_{cj}.
- Our strategy is to find a minimum spanning tree T_j to reduce the length of the bus.

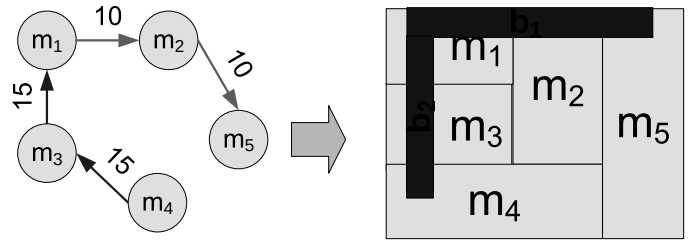
Finding MST

- Two constraints:
  - At most one horizontal (or vertical) bus component of a bus can pass through a module on its bus net.
  - The alignment between modules due to a bus should not violate with the alignment due to those previously routed buses.
- Modified Kruskal's algorithm.
- Cycle detection after finding the MST for each bus.

Finding Bus Components

- Two kinds of edges in T_j
  - Those coming from G_v'.
  - Those coming from G_h'.
- Group a sequence of adjacent edges of the same kind together.
- Each forms a 0-bend bus component.

Finding Bus Components



Spanning Tree

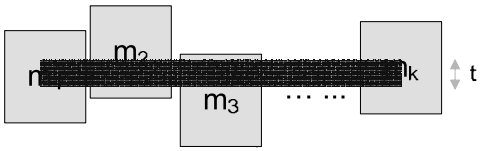
Bus Components

Alignment Enforcement

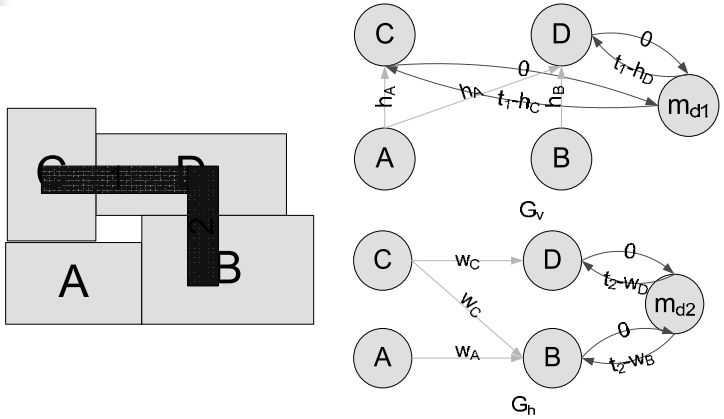
- New edges are added to the original TCG (G_h and G_v) to align modules to allow the buses to pass through.
- Now, we only need to consider each 0-bend bus component one by one.

Alignment Enforcement

- Suppose we now want to align i modules horizontally to allow a bus of width t to pass through.
- We only need to add to G_v a dummy node m_d and add edges between m_d and the modules involved.



Alignment Enforcement



Detect cycles in the TCG after adding edges for a bus.

Floorplan Realization

- Prevent bus overlapping:
  - Natural order from TCG.
  - Otherwise, impose an order arbitrarily between two bus components if they:
    - Share at least one module, and
    - Interleave in the x- or y-direction,

    by adding an edge between the two corresponding dummy vertices.

Cost Function in S.A.

$$Cost = \begin{cases} \alpha A + \beta B + \gamma I & if\ B > \delta \\ \alpha A + \gamma I & if\ B \le \delta \end{cases}$$

where
A is the chip area,
B is the total bus area,
I is the number of infeasible buses,
α, β, and γ are parameters specified by users
δ is the threshold for bus cost

Speedup of Annealing Process

- Bus assignment is the most time-consuming step.
- Compute chip area A before invoking the bus assignment step.
- Compare A with the previous cost C:
  - If $A < C \Rightarrow$ perform bus assignment.
  - Otherwise $\Rightarrow$ perform bus assignment with a probability $e^{-((C-A)/T)}$.
- Reduce over 70% run time.

Handling of Soft Modules

- The soft modules are handled by another S.A. process on the final floorplan by repeatedly:
  - Finding out a critical path p from the TCG.
  - Choosing a module m on p randomly.
  - Reducing the width or height of m.
  - Computing the new cost.

Experimental Results - 1

- Data sets from Xiang *et al.* at ICCAD 03.

Test Case	No. of Blocks	No. of Buses	Average/Max Bus Net Size
apte	9	5	2.6/3
Xerox	10	6	2.5/3
hp	11	14	2.29/4
ami33-1	33	8	4.17/6
ami33-2	33	18	2.39/4
ami49-1	49	9	4.00/6
ami49-2	49	12	3.58/6
ami49-3	49	15	3.53/6

Experimental Results - 1

	[3] Hua Xiang			Our Work			Comparison	
	Time / s	Dead Space / %		Time / s	Dead Space / %		%	
		Without Soft Adjust	With Soft Adjust		Without Soft Adjust	With Soft Adjust	Without Soft Adjust	With Soft Adjust
apte	12 (+1)	4.11	0.72	1 (+0)	2.51	0.7	-38.93	-48.61
Xerox	13 (+1)	3.88	0.95	1 (+0)	2.83	0.40	-27.06	-57.89
Hp	28 (+0)	5.02	0.62	2 (+0)	3.73	0.56	-25.70	-9.68
ami33-1	62 (+1)	6.02	0.94	22 (+1)	5.72	0.77	-4.98	-18.09
ami33-1	86 (+5)	6.10	1.27	31 (+1)	5.46	1.03	-10.49	-18.90
ami49-1	101 (+3)	5.42	0.85	70 (+3)	5.78	0.96	+6.64	+12.94
ami49-2	281 (+3)	6.09	0.84	92 (+3)	5.32	0.97	-12.64	+15.48
ami49-3	268 (+3)	7.40	1.09	121 (+2)	7.08	1.26	-4.32	+15.60
						Average:	-14.69	-13.64

Both algorithms are run on the same machine.

Experimental Results - 2

- Data sets from Law *et al*. at ISPD 05.

Test Case	No. of Blocks	No. of Buses	Average/Max Bus Net Size
ami33-3	33	1	10/10
ami33-4	33	3	10/10
ami33-5	33	5	10/10
ami49-4	49	1	15/15
ami49-5	49	3	11.67/15
ami49-6	49	4	11.25/15

Experimental Results - 2

	[1]		Our Work		Comparison
	Time / s	Dead Space / %	Time / s	Dead Space / %	%
ami33-3	32	1.01	10	0.84	-16.83
ami33-4	92	1.90	31	1.28	-66.32
ami33-5	95	3.80	31	1.28	-66.32
ami49-4	88	0.63	32	0.87	+38.10
ami49-5	261	1.17	44	1.13	-3.42
ami49-6	140	2.19	104	1.64	-25.11
				Average:	-22.62

Both algorithms are run on the same machine.

Experimental Results - 3

- Consider the cases when the bus nets are large.
- Data sets are derived from ami33

Test Case	No. of Blocks	No. of Buses	Average/Max Bus Net Size
ami33-a	33	5	3/4
ami33-b	33	5	4/5
ami33-c	33	5	5/6
ami33-d	33	5	6/7
ami33-e	33	5	7/8
ami33-f	33	5	8/9

Experimental Results - 3

	[1] Jill Law			Our Work			Comparison / %		
	Time / s	Dead Space / %		Time / s	Dead Space / %		Time	Dead Space	
		Without Soft Adjust	With Soft Adjust		Without Soft Adjust	With Soft Adjust		Without Soft Adjust	With Soft Adjust
ami33-a	31.3 (+0)	6.40	1.80	9.7 (+0)	6.30	1.42	-69.01	-1.56	-21.08
ami33-b	32.8 (+1)	8.23	3.16	12.3 (+0)	6.59	1.84	-62.50	-19.92	-41.73
ami33-c	34.0 (+1)	8.51	2.58	16.4 (+0)	7.82	2.34	-51.76	-8.10	-9.46
ami33-d	31.6 (+1)	10.35	2.57	22.0 (+1)	8.36	2.35	-30.38	-19.24	-8.64
ami33-e	20.4 (+0)	*10.35	*3.13	24.5 (+2)	8.99	1.97	+20.10	-17.58	-36.83
ami33-f	26.5 (+1)	*9.52	*2.21	26.8 (+1)	9.46	1.98	+1.13	-0.63	-10.29
						Average:	-32.07	-11.17	-21.34

* Some buses are in-feasible

Experimental Results - 3

- More detailed study of the S.A. process:

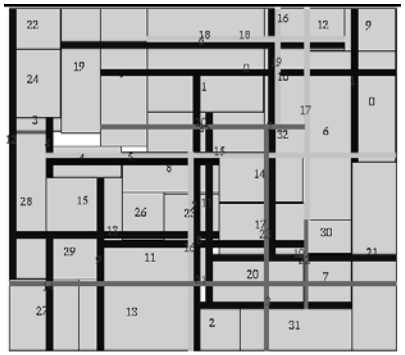
	[1] Jill Law		Our Work	
Test Case	No. of SA Iteration	Feasible Floorplan / %	No. of SA Iteration	Feasible Floorplan / %
ami33-a	1,557,336	73.8	73,971	92.1
ami33-b	1,557,336	58.7	81,388	84.6
ami33-c	1,557,336	50.2	63,881	73.7
ami33-d	1,557,336	29.0	84,149	71.7
ami33-e	1,557,336	0.0	69,557	75.6
ami33-f	1,557,336	0.0	81,590	69.6

Experimental Results - 4

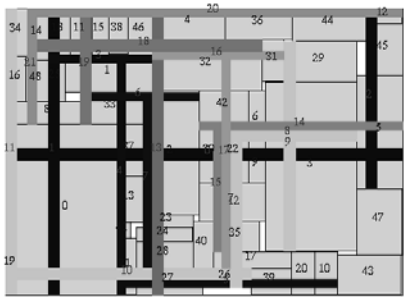
- Data sets derived from ami49 with only one bus of very large bus net:

Test Case	Net Size	Run Time / s	Dead Space / %	Dead Space with Soft Adj. /%	Feasible Floorplan / %
ami49-a	10	40 (+2)	5.32	1.21	91.78
ami49-b	20	40 (+2)	5.97	1.55	89.36
ami49-c	30	40 (+2)	6.47	1.50	84.86
ami49-d	40	49 (+2)	7.86	1.74	81.02
ami49-e	49	51 (+2)	9.35	2.16	77.82

Some Resultant Floorplans



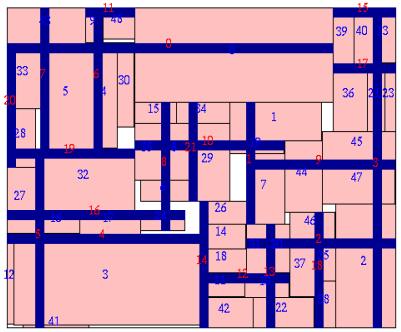
- ami33-5



- ami49-1

Some Resultant Floorplans

- ami49-3



- ami49-e