Integer linear programming in routing problems

Kelly Kaoudis
CSCI 5654, Fall 2013

December 10, 2013

Introduction

Routing problems, which are NP-complete, can be reduced to ILPs and iteratively solved in reasonable time. Linear program standard form[7] is

$$max\ \zeta = \sum_{j=1}^{n} c_j x_j$$

$$st\ w_i = b_i - \sum_{j=1}^{n} a_{ij} x_j\ where\, i = 1, ..., m$$

$$x_j, w_i \geq 0$$

We examine methods for *reducing* to an ILP, but it is worth mentioning that with heuristics such as interior point methods or the simplex method with LP relaxation, branch-and-bound, and/or cuts including lift-and-project, optimal solutions for ILPs are found in polynomial time. In routing problems, we *minimize* to find the least interrupted (and possibly even shortest) route from each start to endpoint, simultaneously. In an LP, decision variables x_j and problem-internals ("slack" variables, w_i) can take any real values. ILPs constrain decision and slack variables $x_j, w_j \in \mathbb{Z}$. In a binary ILP, decision variables are further constrained: $0 \leq x_j \leq 1$, $\forall x_j \in \vec{x}$. Two problems which reduce to ILPs are the global routing approximation stage in laying out VLSI circuits, and reducing rail line crossings in metropolitan public transportation route-maps.

Integrated circuit layout

A minimal interconnection set with low route congestion, especially at nanometer scale, improves circuit layout[9]. Layout has four subproblems: partitioning, placement, global routing approximation, and detailed local routing. In global routing, which is NP-hard, we represent each net by a set of feasible rectilinear minimum Steiner trees (RMSTs), then choose the set of trees (one per net in the netlist) to minimize interconnection.

Given a set of pins that should connect, a spanning tree[6] built with only vertical and horizontal line segments is a feasible route for that net. For a net of r cells, there are $r^{r-2}2^{r-1}$ constructable RMSTs with minimum vias, so linear programming becomes a very attractive way to find the optimal tree set.

Our tree sets N_k also help approximate routing demand per net. Assign a $y_j \in \vec{y}$ to each tree j with $y_j = 1$ if $j \in$ final layout, 0 otherwise. $\vec{y}$ is the set of *all* trees for the netlist. The probability that a $y \in N_k$ is the best tree for net k is $p(y_j) = \frac{1}{no.\, trees \in N_k}$. $\forall$ edges $e_i \in y_j$, estimate the number of *other* trees that pass through: $r(e_i) = \sum_{j=1}^{t} a_{ij} p(y_j)$, the routing demand for tree j. t is the total trees $\in \vec{y}$. $a_{ij} = 1$ if j intersects e_i and 0 otherwise.

The sum of routing demands of other trees' edges y_j passes through, $c_j = \sum r(e_i)$ where $e_i \in y_j \in \vec{y}$, is produced once all trees are generated, and will be incorporated as a penalty function on the ILP's objective $\forall$ trees j. For nets passing through congested edges, construct additional trees with perhaps a greater wire length or number of pins, but avoiding those edges.

We also calculate $f_{congestion} = c - r(e_i)$ and $f_{utility} = \frac{c - r_l(e_i)}{r_u(e_i) - r_l(e_i)}$, the ratio between the "available" vias for an edge after routing nets with one

or two possible trees, and the number of trees that could pass through the currently available tracks. c is the max via-capacity of an edge (channel). If $f_{congestion}(e_i) < 0$ our estimation is greater than the max capacity of e_i, so routing the circuit as-is is infeasible. If $f_{congestion}(e_i) < \frac{c}{10}$, (channel is at more than 90 per cent capacity), re-route nets with vias on e_i until e_i is no longer above the threshold and don't use it as a possibility for more vias. Only consider the pre-ILP circuit feasible if edge-congestion is below 90 per cent $\forall$ edges in our selected trees.

Metro rail-map layout

Designing a readable, reasonably geographically coherent, aesthetically pleasing metro map also has an NP-hard[1] routing component. Asquith et al.[2] were the first to translate line-crossing (visual congestion) minimization to an integer linear program over an unrestricted plane graph underlying the metro map where stations are vertices and edges are parts of transit lines, which may cross at or between vertices. The ILP construction is a bit different from [3], though parallels exist. Visual clarity in a metro map improves with fewer crossings at vertices; however, transit route utility is greater when a route has more shared stops. While calculation time for the optimal number of line-crossings in a metro layout is polynomial when the map's terminal positions are predetermined[5], the general case is NP-complete.

Their algorithm requires four steps:

1. Calculate all maximal common sub-paths for each pair of individual routes (metro lines) (g, h) traversed in the network $\delta_1(g, h)...\delta_m(g, h)$
2. Convert each of these into a crossing rule c dependent upon position of the terminals that tells whether g and h cross
3. Formulate an ILP with the problem information to determine how many crossings exist in a minimal representation of the graph for the metro map
4. Decide the ordering of lines at each vertex for aesthetics

A crossing rule tells in what ordering a crossing occurs. The authors prove that in an optimal ILP solution, two paths g and h cross exactly 1 or 0 times. Either

- path g is above path h at v_0 AND path h is above path g at v_m or
- path h is above path g at v_0 AND path g is above path h at v_m

where v_0 and v_m are the end vertices of the maximal common subpath for (g, h). Each clause in the previous crossing rule is a Boolean variable during construction of the ILP.

Reducing rail-route crossings in [2] and reducing vias and edge-congestion in [3] appear to be very similar problems: both are NP-complete, NP-hard, and affect multiple aspects of the rest of the problem. In both IC layout and map construction, we lessen graph congestion as much as possible. We consider a via in circuit design and a rail line crossing analogous here.

Problem representations

We'll sketch the ILP formulations resulting from the two above procedures and briefly compare.

Each crossing rule c in the metro graph problem is represented by a $w_i \in \vec{w}$ and constrained such that its w_i is 1 if there is a crossing and 0 otherwise. Every terminus referenced in at least one $c_i \in \vec{C}$ is included in the ILP. The metro-line ILP ζ is $min \sum_{i=1}^{n} w_i$, subject to at least one constraint on each w_i, dependent on the specific graph instance. The circuit problem is set up a bit differently in that its objective takes into account more aspects of via reduction; one might even go so far as to say the metro problem is akin to a smaller, perhaps more constrained instance of the circuit problem, depending on the sizes and edge/route-congestion of the map and circuit in question.

In the global routing problem, Vannelli originally proposed an ILP for just minimizing total wire length [8]. Behjat et al.[3] say this older model can produce infeasible results on several of the largest MCNC[4] benchmark circuits. Un-necessary congestion may remain in the optimized result because the ILP considers incomplete information, which results in hotspots,

capacitive coupling, and other instability. If channel capacities are fixed at 90 per cent of their actual values, these issues are much less likely. Their second previous attempt at ILP formulation reduces vias on a tree j as well as in j's path but doesn't minimize channel capacities. The model they propose combines elements of the previous to minimize the channel capacity and also wire length, number of vias, and congestion simultaneously.

$$min \sum_{j=1}^{t} w_j y_j + \beta_z z, \ where\, z \in 0, ..., c$$

$$st \sum_{y_j \in N_k} y_j = 1, \ k = 1, ..., n$$

$$\sum_{j=1}^{t} a_{ij} y_j - z \leq 0, \ i = 1, ..., p$$

$$y_j \in 0, 1, \ where\, j = 1, ..., t$$

$w_j = \beta_l w_{l_j} + \beta_v w_{v_j} + \beta_c w_{c_j}$ where w_{v_j} is a weight on the number of vias, w_{c_j} is a weight for congestion in the path of tree j, and scalars $\beta_{l,v,c}$ change weighting factor emphasis. z is the channel capacity, and $\beta_z \in \mathbb{R}$ is the weight for max channel capacity.

While the metro-crossing ILP appears more constrained in channel capacity (a line is crossed at most once by every other metro line) than the circuit routing problem, if the metro map in question has enough intertwined lines, the channel capacities of rail-line segments between vertices and RMST edges could be comparable. If there are a great enough number of lines (or sub-circuits) in a routing problem and we apply both ILP formulations, it may be possible that the IC formulation, which requires all edges in the RMSTs in a feasible solution to have 90 per cent or less of their channel capacities fulfilled, could produce a more minimal result than the metro ILP.

Further work

Additional exploration into ILPs for routing with minimal congestion should examine more works and also verify results. It would be interesting to try the metro map ILP on MCNC benchmarks of varying sizes, and also to fit Behjat et al.'s circuit ILP on metro graphs for time-efficiency and optimization comparison. Due to space constraints, not everything presented in [3] and [2] could be covered this time.

References

[1] Evmorfia N. Argyriou, Michael A. Bekos, Michael Kaufmann, and Antonios Symvonis. On metro-line crossing minimization. *J. Graph Algorithms Appl.*, 14(1):75–96, 2010.

[2] Matthew Asquith, Joachim Gudmundsson, and Damian Merrick. An ilp for the metro-line crossing problem. *Proceedings of the Fourteenth Symposium on Computing: The Australasian Theory - Volume 77*, pages 49–56, 2008.

[3] Laleh Behjat, Anthony Vannelli, and William Rosehart. Integer linear programming models for global routing. *INFORMS J. on Computing*, 18(2), jan 2006.

[4] Xiaolei Chen. Mcnc golden 20 fpga benchmarks. `https://sites.google.com/site/xiaoleicustc/research/mcnc`.

[5] Martin Nollenburg. An improved algorithm for the metro-line crossing minimization problem. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing*, volume 5849 of *Lecture Notes in Computer Science*, pages 381–392. Springer Berlin Heidelberg, 2010.

[6] Naveed A. Sherwani. *Algorithms for VLSI Physical Design Automation*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[7] Robert J. Vanderbei. *Linear Programming: Foundations and Extensions*. 1996.

[8] Anthony Vannelli. An adaptation of the interior point method for solving the global routing problem. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 10:193–203, 1991.

[9] Tai-Hsuan Wu, Azadeh Davoodi, and Jeffrey T. Linderoth. Grip: Global routing via integer programming. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 30:72–84, 2011.