An Engineering Application of the Simulated Annealing Algorithm

Edgar N. Reyes, Carl W. Steidley
Southeastern Louisiana University / Texas A&M University - Corpus Christi

1 Introduction

Global wiring of integrated circuits is an engineering application using both combinatorial optimization and statistical physics. In this paper, we interpret the problem of how to best wire an integrated circuit as a combinatorial optimization problem. We employ a random search technique, namely, the simulated annealing algorithm, to find near-optimal solutions of the combinatorial optimization problem. As it turns out it is well-known that the objective function of this optimization problem has the form of a Hamiltonian of an Ising model which has connections to the study of phase transitions. This paper is a continuation of [1].

To describe the wiring problem, we briefly review some of the discussions in [1]. Let us suppose the wires will be laid out in a square grid, say, $G = \{(x, y) \in \mathbb{Z}^2 : 1 \leq x \leq n, 1 \leq y \leq n\}$ for some positive integer n. Let the points (or nodes) in G represent the sources and sinks of the wires. We say that $(x_1, y_1), (x_2, y_2) \in G$ are adjacent points or adjacent nodes if either $(x_1 + 1, y_1) = (x_2, y_2)$, $(x_1 - 1, y_1) = (x_2, y_2)$, $(x_1, y_1 + 1) = (x_2, y_2)$, or $(x_1, y_1 - 1) = (x_2, y_2)$. The line segment joining two adjacent nodes is called a link. Note, the number L of links in G is given by $L = 2n(n - 1)$.

We suppose that it has been determined in advance that certain pairs of points or nodes have to be connected. A wire which starts at one node and ends at another node in the same vertical or horizontal line as the first node will be called a straight wire. We suppose that all wires must be placed along the links. A non-straight wire joining two nodes can be placed (also along the links) in at least two different paths. A wire which follows a path with only one bend is said to have an L-shaped path. In this paper, the authors restrict themselves only to straight wires and L-shaped wires.

We assign 1 as the orientation for the path connecting A to B and for the path connecting C to D shown in the two figures on the left. The two figures on the right exhibit an orientation to which we assign -1.

If a total number, M, of L-shaped wires must be used (this excludes straight wires), then a wiring of the square grid is an M-tuple $w = (w_1, ..., w_M)$ where $w_i = \pm 1$. Given a wiring configuration w, let $m_v(w)$ be the number of wires passing through the v^{th} link. An objective function which gives the most 'balanced' arrangement is

$$C(w) = \sum_{v=1}^{L} m_v(w)^2. \tag{1.1}$$

This objective function is the one that is also used in [3].

To describe $m_v(w)$ more explicity, for each $i \in \{1, ..., M\}$ and each link v, let

$$e_{i,v} = \begin{cases} 1 & \text{if } v \text{ is a link in the } i^{\text{th}} \text{ wire when the orientation of the wire is } 1 \\ -1 & \text{if } v \text{ is a link in the } i^{\text{th}} \text{ wire when the orientation of the wire is } -1 \\ 0 & \text{otherwise} \end{cases}$$

and let

$$a_{i,v} = \begin{cases} 1 & \text{if } v \text{ is a link in the } i^{\text{th}} \text{ wire in one of its orientations} \\ 0 & \text{otherwise} \end{cases}.$$

Thus, the number of wires (these include straight and L-shaped) passing through link v is given by

$$m_v(w) = m_v(0) + \sum_{i=1}^{M} \left(a_{i,v} \frac{e_{i,v} w_i + 1}{2} \right)$$

where $m_v(0)$ is the number of wires on link v coming from straight wires. Then it can be shown that

$$C(w) = \sum_{i=1}^{M} h_i w_i + \sum_{i,j=1}^{M} J_{i,j} w_i w_j + K. \tag{1.2}$$

where

$$K = \sum_{v=1}^{L} m_v^2(0) + \sum_{i,v} a_{i,v} m_v(0) + \sum_{i,j,v} \frac{a_{i,v} a_{j,v}}{4}, \tag{1.3}$$

$$h_i = \sum_{v=1}^{L} e_{i,v} m_v(0) + \frac{1}{2} \sum_{j,v} a_{j,v} e_{i,v},$$

and

$$J_{i,j} \quad = \quad \sum_{v=1}^{L} \frac{e_{i,v} e_{j,v}}{4}.$$

We point out that (1.2) takes the form of a Hamiltonian of an Ising Model.

2 Wiring with Long Term Connections

As discussed in the introduction, we assume that the pairs of nodes to be connected by wires have been determined in advance and that our task is to find a wiring or configuration w that minimizes (1.2). Specifically, two nodes will be connected exactly when they do not lie on a horizontal or vertical line. Since there will be no straight wires, $m_v(0) = 0$ for each link v and

$$h_i \quad = \quad \frac{1}{2} \sum_{v} e_{i,v} \sum_{j} a_{j,v}.$$

If v is the horizontal link joining (a, b) to $(a+1, b)$ where $1 \leq a \leq n-1$, then it can be shown that $\sum_{j=1}^{M} a_{j,v} = 2a(n-a)(n-1)$. Similarly, if v is the vertical link joining (a, b) to $(a, b+1)$ where $1 \leq b \leq n-1$, then $\sum_{j=1}^{M} a_{j,v} = 2b(n-b)(n-1)$. Now, given any wire say the i^{th}, if v is a horizontal link in this wire such that $e_{i,v} = 1$, then there is another horizontal link v' such that $e_{i,v'} = -1$ and $\sum_{j=1}^{M} a_{j,v} = \sum_{j=1}^{M} a_{j,v'} = 2a(n-a)(n-1)$ for some $1 \leq a < n$. Then $\sum_{\text{horiz. } v} e_{i,v} \sum_{j} a_{j,v} = 0$. Similarly, $\sum_{\text{vert. } v} e_{i,v} \sum_{j} a_{j,v} = 0$. Thus,

$$h_i = 0 \quad \text{for all } i.$$

Hence, the objective function reduces to

$$C(w) = \sum_{i,j} J_{i,j} w_i w_j + K$$

where $K = \dfrac{n^2(n-1)^2(n^4-1)}{15}$. Note, the number of wires, M, is given by $M = \dfrac{n^2(n-1)^2}{2}$.

Let $[(s,t),(x,y)]$ represent the wire that joins node (s,t) to node (x,y) where $s < x$ and $t \neq y$. Then the length of wire $[(s,t),(x,y)]$ is $\text{Length}([(s,t),(x,y)]) = x - s + |y - t|$. From this, it can be shown that the total length of all the wires is

$$\sum_{s=1}^{n-1} \sum_{t=1}^{n} \sum_{y:y \neq t} \sum_{x:s<x} x - s + |y - t| \quad = \quad \frac{n^2(n-1)^2(n+1)}{3}.$$

Rewriting the objective function, further, we find

$$\begin{aligned} C(w) &= \sum_{i=1}^{M} \frac{1}{4} \sum_{v} a_{i,v} + 2 \sum_{i<j} J_{i,j} w_i w_j + K \\ &= \sum_{i=1}^{M} \frac{1}{4} \cdot 2 \,(\text{Length of } i\text{th wire}) + 2 \sum_{i<j} J_{i,j} w_i w_j + K \\ &= \frac{1}{2} \cdot (\text{total length of all wires}) + 2 \sum_{i<j} J_{i,j} w_i w_j + K \\ &= \frac{n^2(n-1)^2(n+1)}{6} + 2 \sum_{i<j} J_{i,j} w_i w_j + \frac{n^2(n-1)^2(n^4-1)}{15} \end{aligned}$$

By using calculus, one finds that a lower bound for $C(w)$ is the value of $\sum_{v} (m_v(w))^2$ when each $m_v(w)$ is replaced by $\dfrac{\text{total length of all wires}}{\text{no. of links}} = \dfrac{n(n-1)(n+1)}{6}$. That is, a bounding inequality is $C(w) \geq \dfrac{n^3(n-1)^3(n+1)^2}{18}$. As we will see, if there exists a wiring configuration w such that $C(w) = \dfrac{n^3(n-1)^3(n+1)^2}{18}$ then necessarily w is an optimal wiring configuration that minimizes (1.2) and $n = 3$. To see this, let E be the M-by-L matrix whose (i,v)-entry is $e_{i,v}$ where $1 \leq i \leq M$, $1 \leq v \leq L$. Let J be the M-by-M matrix whose (i,j)-entry is $J_{i,j}$. Then $J = \frac{1}{4} E E^t$ and

$$\begin{aligned} C(w) &= \sum_{i,j} J_{i,j} w_i w_j + K \\ &= \frac{1}{4} \langle E^t w, E^t w \rangle + \frac{n^2(n-1)^2(n^4-1)}{15} \end{aligned}$$

where $\langle \cdots, \cdots \rangle$ is the usual dot product. If there exists a configuration w such that $C(w) = \dfrac{n^3(n-1)^3(n+1)^2}{18}$, then $\langle E^t w, E^t w \rangle = -\dfrac{2n^2(n-3)(n-2)(n-1)^3(n+1)}{45}$. Thus, $n = 3$ since the left-hand side is non-negative.

For this case $n = 3$, an optimal configuration (among 196 different optimal configurations) is $w_{opt} = (1,1,-1,-1,1,-1,1,1,-1,1,-1,1,1,1,-1,-1,1,-1)$. We obtained w_{opt} by using the simulated annealing algorithm and a software package for computational discrete algebra called GAP. In the last step in section 3, the value we used for $guess$ is -12 since $\sum_{i<j} J_{i,j} w_i w_j \geq -12$. At the publication time, the authors have not found an optimal configuration even for the case when $n = 4$. One difficulty encountered here is that a feasible solution is a 72-tuple of ± 1's and there are 2^{72} different feasible solutions.

3 Implimenting Simulated Annealing with GAP

In this section, we show the program we used when implementing GAP.

```
gap > coord := [ ]; ; wires := [ ]; ; links := [ ]
gap > setup := function(n)
gap > local i, j, a, b; coord := [ ]; wires := [ ]; links := [ ];
gap > for i in [1..n] do for j in [1..n] do Append(coord, [[i, j]]); od; od;
gap > for a in coord do for b in coord do
gap > if a[1] < b[1] and a[2] <> b[2] then  Append(wires, [[a, b]]); fi; od; od;
gap > for i in [1..n - 1] do for j in [1..n] do
gap > Append(links, [[[i, j], [i + 1, j]]]); od; od;
gap > for i in [1..n] do for j in [1..n - 1] do
gap > Append(links, [[[i, j], [i, j + 1]]]); od; od; end;
gap >
gap > e := function(i, v)
gap > local m, M;
gap > m := Minimum(wires[i][1][2], wires[i][2][2]);
gap > M := Maximum(wires[i][1][2], wires[i][2][2]);
gap > if links[v][1][2] = links[v][2][2] and
gap > wires[i][1][1] <= links[v][1][1] and links[v][1][1] < wires[i][2][1] then
gap > if links[v][1][2] = wires[i][1][2] then return
gap > SignInt(wires[i][2][2] - wires[i][1][2]);
gap > elif links[v][1][2] = wires[i][2][2] then return
gap > -SignInt(wires[i][2][2] - wires[i][1][2]); else return 0; fi;
gap > elif links[v][1][1] = links[v][2][1] and
gap > m <= links[v][1][2] and links[v][1][2] < M then
gap > if links[v][1][1] = wires[i][2][1] then return
gap > SignInt(wires[i][2][2] - wires[i][1][2]);
gap > elif links[v][1][1] = wires[i][1][1] then return
gap > -SignInt(wires[i][2][2] - wires[i][1][2]); else return 0; fi;
gap > else return 0; fi; end;
```

```
gap >
gap >     eps := [ ];;
gap >     epssetup := function(w, L)
gap >     local i, v; eps := [ ];
gap >     for i in [1..w] do eps[i] := [ ]; od;
gap >     for i in [1..w] do for v in [1..L] do eps[i][v] := e(i, v) od; od; end;
gap >
gap >     jfunc := function(i, j, L)
gap >     local v, sum; sum := 0;
gap >     for v in [1..L] do sum := sum + eps[i][v] * eps[j][v]; od;
gap >     return sum/4; end;
gap >
gap >     J := [ ];;
gap >     jsetup := function(w, L)
gap >     local i, j; J := [ ];
gap >     for i in [1..w] do J[i] := [ ]; od;
gap >     for i in [1..w] do for j in [1..w] do J[i][j] := jfunc(i, j, L); od; od; end;
gap >
gap >     eval := function(w, x)
gap >     local i, j, p; p := 0;
gap >     for i in [1..w] do for j in [i + 1..w] do
gap >     p := p + J[i][j] * x[i] * x[j]; od; od;
gap >     return p; end;
gap >
gap >     ko := [ ];; prodo := 0;;
gap >     initsetup := function(w)
gap >     local i, j;
gap >     for i in [1..w] do ko[i] := 1; od;
gap >     prodo := eval(w, ko); end;
gap >
gap >     algo := function(w, max)
```

```
gap > local a, p, k, prod;
gap > for p in [1..max] do
gap > k := ko;
gap > a := Random([1..w]); k[a] := -k[a]; prod := eval(w, k)
gap > if prod < prodo then ko := k; prodo := prod; fi;
gap > Print(p, "\n"); Print(prodo, "\n"); od; return ko; end;
gap >
gap > setup(3); epssetup(18, 12); jsetup(18, 12); initsetup(18); algo(18, 2000);
gap > ko; eval(ko);
```

4 Summary

In this paper, which is a continuation of our earlier paper [1], we have examined in some detail the problem of wiring an integrated circuit. Following and expanding on the discussion given in [3], this wiring problem can be seen as and is equivalent to a combinatorial minimization problem. We took a square grid $G = \{(x, y) \in \mathbb{Z}^2 : 1 \leq x, y \leq n\}$ and assumed that any pair of points that do not lie in the same horizontal or vertical line had to be connected by an L-shaped wire. The number of such pairs, which is $M = \frac{n^2(n-1)^2}{2}$, is the number of wires needed to wire the square grid G. An L-shape wire has two orientations, that is, ± 1. The wiring problem reduces to deciding, for each wire, whether to assign 1 or -1. In deciding this, our criterion for a best wiring configuration is a configuration that minimizes the sum of the squares of the number of wires passing through each link - this is the objective function described in (1.1).

When $n = 3$, we have exhibited an optimal configuration. This configuration shows nice uniformity and symmetry. That is, in each link, there are exactly four wires passing through each link. The simulated annealing algorithm was used to generate an optimal configuration. The algorithm was implemented with the use of the software GAP - a system used in computational discrete algebra (which can be downloaded free of charge) [2]. For the case $n = 4$, we encountered computational difficulty since the set of feasible solutions becomes large and we were not able to identify an optimal configuration. In section 3, we included the GAP program we used. Finally, we remark that the simulated annealing algorithm with the code described in section 3, one can generate *near-optimal* solutions for any case $n \geq 4$.

References

[1] Reyes, E.N. and Steidley, C., *"Optimization Using Simulated Annealing"*, Conference Proceedings of Northcon '98, pp. 120-126, Seattle, WA, October, 1998.

[2] The GAP Group, Lehrstuhl D für Mathematik, RWTH Aachen, Germany and School of Mathematical and Computational Sciences, U. St. Andrews, Scotland. *GAP – Groups, Algorithms, and Programming, Version 4*, 1997.

[3] Veechi, M.P. and Kirkpatrick, S., *Global Wiring by Simulated Annealing*, IEEE Transactions on Computer-Aided Design, Vol. Cad-2, No. 4, pp. 215-222, October 1983.

EDGAR N. REYES

Edgar N. Reyes is an Associate Professor in the Department of Mathematics at Southeastern Louisiana University in Hammond, Louisiana. His interests include optimization and group representation theory. Dr. Reyes received a Ph.D. from the Department of Mathematics at Louisiana State University in 1988.

CARL W. STEIDLEY

Carl W. Steidley is Professor and Chair of Computing and Mathematical Sciences at Texas A&M University - Corpus Christi. His research interests include: computational mathematics and applied artificial intelligence.