Model and Solution Strategy for Placement of Rectangular Blocks in the Euclidian Plane

Amir Alon and Uri Ascher
Technical Report 86-11
May 1986

Abstract

This paper describes a nonlinear optimization model for the placement of rectangular blocks with some wire connections among them in the Euclidian plane, such that the total wire length is minimized. Such a placement algorithm is useful as a CAD tool for VLSI and PCB layout designs.

The mathematical model presented here ensures that the blocks will not overlap and minimizes the sum of the distances of the interconnections of the blocks with respect to their orientation as well as their position. We also present mechanisms for solving more restrictive placement problems, including one in which there is a set of equally spaced, discrete angles to be used in the placement. The mathematical model is based on the Lennard-Jones 6-12 potential equation, on a sine wave shaped penalty function, and on minimizing the sum of the squares of the Euclidian distances of the block interconnections. We also present some experimental results which show that good placements are achieved with our techniques.

The research of the first author was supported in part under an NSERC (Canada) postgraduate scholarship. The second author was supported in part under NSERC grant A4306.

1 Introduction

In this work we consider the placement problem for rectangular blocks of various dimensions and orientations in the plane. Each block has a set of *terminals* (or *ports*) which are fixed points on the boundary of the block. As part of the input we also get a list of pairs of terminals that are to be connected to each other. At times additional restrictions are placed on the choice of orientation or position of the blocks. The objective of our optimization is to place the blocks such that the total sum of the lengths of the interconnections is minimized without having any two blocks overlap and possibly satisfying additional orientation and location constraints.

Two common applications where some variations of this placement problem arise are Printed Circuit Boards(PCB) [35,12,6] and Very Large Scale Integration(VLSI) layouts [7,8,20,11,13]. Instances also occur in operations research location/allocation problems [12,15], as well as in office layout problems [9].

Generally, many factors are involved in producing a good placement. These include, for example, the total wire length, wire crossings, heat dissipation and total circuit area [17,27,30]. Note that while total wire length does not represent all of these design goals exactly, it is reasonable to assume that overall shorter wires lead to smaller circuit area, less resistance, and fewer wire crossings. We have adopted the total wire length measure, as most placement algorithms do [13,11,22,17], since it also gives a reasonable indication as to how expensive it is to construct the detailed routing of the

wires after the blocks have been placed.

Even after choosing the objective function to be the total wire length, placement problems are still computationally hard [5,17,12]. Mathematically, we obtain a nonlinear optimization problem with some of the variables possibly restricted to have integer values only. As a result of the difficulty of the placement problem and the heuristic meaning of "best" placement, most of the work on the subject is concentrated on getting only a "good" placement. By a "good" placement we mean that the blocks are placed in good *topological* relation to each other but are not necessarily separated exactly by the required distances between them. Thus, we accept certain suboptimal solutions, and even allow the constraint of separation of the blocks to be slightly violated. This is an approximate solution of the optimization problem, solving a reasonable relaxation of it since the exact distance separating the blocks is not known until the routing phase which follows the placement.

Most of the placement techniques reported in the literature have concentrated on placing a large (say > 50) number of blocks, ignoring their size, orientation and position of terminals [30,32,33]. Placement techniques can be roughly classified as *discrete* or *continuous*. A discrete placement technique assumes a set of discrete locations (usually on a grid lattice) on which the blocks are to be located. Continuous placement techniques assume placement in the Euclidian plane (or a bounded part of it). The choice between continuous and discrete techniques varies with the specific place-

ment model at hand. PCB placements, for example, are naturally modeled as discrete, while VLSI placements are naturally continuous or mixed. The computational difficulties associated with each technique are also an important factor. Discrete techniques offer more exact representations but are usually less tractable than continuous ones.

Discrete placement is usually performed by a two phase algorithm. An *initial constructive placement* phase is followed by an *iterative improvement* or *refinement* phase. The iterative improvement phase is a crucial one, and usually involves choosing two blocks and interchanging their position if this reduces the total wire length [17,18,19]. Alternatively a discrete model can be used to formulate an integer programming problem [25] or can be solved via branch and bound methods [29,12,2,28] or via statistical annealing methods [36].

Other placement techniques use a *continuous* model, usually drawing on physical phenomena as a model for placement. These include Hooks law force directed models [30] and Resistive network optimization [7]. Also available are various hybrid models, usually using a continuous algorithm for getting an approximate placement and a discrete algorithm for mapping the result onto grid locations.

We have chosen a continuous model for placement: that of the potential energy between particles [1,31]. We use smooth nonlinear penalty functions to represent the constraints of the model and show how to effectively solve the nonlinear unconstrained optimization problem which results from our

formulation. The problem is modeled in more detail than has been previously done by taking into account the size and orientation of the blocks, as well as the positions of the terminals on the blocks' edges. As a result, our mathematical problem is computationally more involved. This is not necessarily a severe limitation since our procedure can be used interactively to aid a human designer, or in a hierarchical design where the number of blocks in each level of the hierarchy is limited.

In the following sections we introduce our nonlinear model for the placement problem (§2). We then describe how to restate the model as an unconstrained nonlinear optimization problem by replacing the constraints with penalty functions (§3). A solution strategy for the resulting optimization problem is discussed in §4 and some experiments with it are reported in §5. Some conclusions are offered in §6.

2 The Mathematical Model

We start by formulating the mathematical entities involved. Let $B = \{1, 2, ..., n\}$ be an index set of the blocks to be placed. For each $i \in B$ we have the following variables:

$x_i =$ the X coordinate of the center of block i
$y_i =$ the Y coordinate of the center of block i
$\theta_i =$ the rotation angle of the center of block i

and the following constants:

$w_i =$ half of the width of block i
$l_i =$ half of the length of block i
$T^i = \{T_1^i, ..., T_m^i\} =$ the set of terminals for block i

For each $T_j^i \in T^i$ we have $T_j^i = (\Delta x_{ij}, \Delta y_{ij})$, where

$\Delta x_{ij} =$ the X displacement of terminal j from the center of block i
$\Delta y_{ij} =$ the Y displacement of terminal j from the center of block i.

We are also given a set of pairs

$$I = \{< T_j^i, T_l^k >, ..., < T_n^m, T_p^q >\},$$

where each pair $< T_j^i, T_l^k > \in I$ designates two interconnected terminals. For each such pair we are also given a constant C_{ijkl} representing the weight or relative importance of the connection. For example, a sixteen bit bus connection between two components may have sixteen times the weight of a single control line. Note that connections between more than two terminals, say m, can be represented by $\frac{m(m-1)}{2}$ pairs of connections or by m connections to the center of gravity of the m terminals [34].

Finally let $distance(T_j^i, T_l^k)$ be the distance between terminal T_j^i and terminal T_l^k, and $distance(Block_i, Block_j)$ be the distance between the centers of blocks i and j, where we define the function

$$distance(< x_1, y_1 >, < x_2, y_2 >) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

i.e., it is the usual Euclidian distance.

We consider three variations of the placement problem, denoted $P1, P2$, and $P3$. The model of $P1$ assumes no restriction on the orientation or location of the blocks but includes the restriction that blocks cannot overlap. The models $P2$ and $P3$ are formed by adding orientation and location constraints to $P1$. In all three problems the optimization aim is to find the

values of the variables x_i, y_i, θ_i for $i = 1, \ldots, n$ which minimize the total wire length. We write the placement problem $P1$ as follows

$$P1: \quad \underset{x_i, y_i, \theta_i}{MINIMIZE} \quad D = \sum_{i,j,k,l=1}^{n} C_{ijkl}\, distance(T_j^i, T_l^k)^2$$

such that

$$distance(Block_i, Block_j)^2 \geq \sigma_{ij}^2 \qquad i \neq j \tag{1}$$

where

$$\begin{aligned} C_{ijkl} &= \begin{cases} \text{given weight} & \text{if } < T_j^i, T_l^k > \in I \\ 0 & \text{if } < T_j^i, T_l^k > \notin I \end{cases} \\ \sigma_{ij} &= \text{minimum separation between blocks } i \text{ and } j. \end{aligned}$$

We calculate the objective function D as follows. Let

$$\begin{aligned} D &= \sum_{i=1}^{n-1} \sum_{k=i+1}^{n} D_{ik} \\ &= \sum_{i=1}^{n-1} \sum_{k=i+1}^{n} (\sum_{<T_j^i, T_l^k> \in I} C_{ijkl} distance(T_j^i, T_l^k)^2) \\ &= \sum_{i=1}^{n-1} \sum_{k=i+1}^{n} (\sum_{<T_j^i, T_l^k> \in I} C_{ijkl} ((\overline{x}_{ij} - \overline{x}_{kl})^2 + (\overline{y}_{ij} - \overline{y}_{kl})^2)) \end{aligned} \tag{2}$$

where

$$\begin{pmatrix} \overline{x}_{ij} \\ \overline{y}_{ij} \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} COS(\theta_i) & -SIN(\theta_i) \\ SIN(\theta_i) & COS(\theta_i) \end{pmatrix} \begin{pmatrix} \Delta x_{ij} \\ \Delta y_{ij} \end{pmatrix} \tag{3}$$

and $(\overline{x}_{kl}, \overline{y}_{kl})$ are defined similarly with respect to the variables of block k. Note that $(\overline{x}_{ij}, \overline{y}_{ij})$ are the coordinates of the terminal T_j^i calculated by rotating the point $(\Delta x_{ij}, \Delta y_{ij})$ (the displacement of T_j^i) around the origin $(0, 0)$ by an angle θ_i and then translating the resulting rotated point by (x_i, y_i) (the center coordinates of block i). The result of this formulation is holding the terminal in a fixed distance from the center of the block. Figure 1 illustrates this formulation graphically.

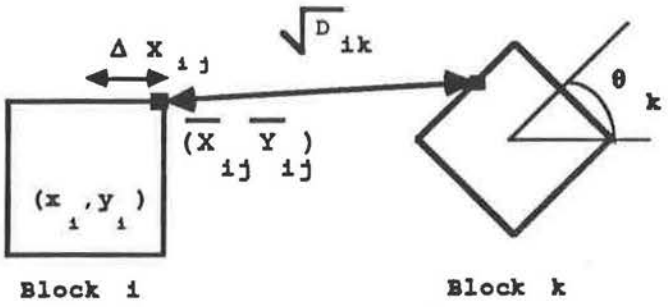

Figure 1: Terminal coordinates

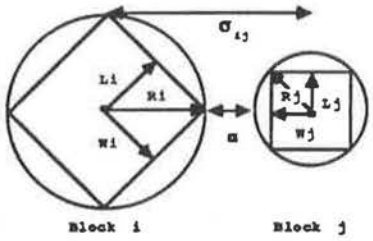

Figure 2: Calculating blocks separation

The minimum distance, σ_{ij}, separating blocks i, j is determined from the radii of the circles containing the blocks plus a minimum required separation, α, as

$$\sigma_{ij} = \sqrt{w_i^2 + l_i^2} + \sqrt{w_j^2 + l_j^2} + \alpha. \tag{4}$$

Figure 2 illustrates the calculation of σ_{ij}. However this σ_{ij} may be an overestimation of the required separation and may not represent the placement correctly if, for some blocks, $w_i << l_i$ or $l_i >> w_i$, since such narrow and long blocks would repel other blocks from their narrow side as the containing circle will be far from the block edge. A simple remedy is to subdivide such blocks into a number of blocks with aspect ratio approaching 1, and link these together via very strong connections. To avoid having these strong

connections dominate the optimization we start with an initial educated placement which ensures that the subblocks are close together and in the proper topology. Although no statistics are available, we assume that long and narrow blocks do not often arise in PCB or VLSI layouts so few new blocks will have to be generated.

In many applications the blocks are not free to assume any orientation. For these applications we assume that the allowed angles are equally spaced. More formally, we allow placed blocks to be oriented in angles $\frac{2k\pi}{m}$ for $k = 0, \ldots, m-1$. If $m = 4$, for example, only horizontally or vertically placed blocks are allowed, i.e., $0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}$ are the only allowed angles. For these cases we add the n constraints to $P1$, obtaining a more restricted optimization problem $P2$

$$\theta_i \in \{\phi_0, \ldots, \phi_{m-1}\} \quad 1 \leq i \leq n \tag{5}$$

for a given set of angles (e.g. $\phi_k = \frac{2k\pi}{m}$).

In applications such as Printed Circuit Boards Placement (PCB's) one may want to consider a set of *discrete* slots into which the blocks are to be placed. Here we are given a set $S = \{< x_1^s, y_1^s >, \cdots, < x_m^s, y_m^s >\}$ of locations such that $m \geq n$ and $< x_i^s, y_i^s >$ are the coordinates of the i^{th} slot. Such a restriction gives rise to the following constraints, which we add to $P2$ to get our third model $P3$

$$(x_i, y_i) \in \{< x_1^s, y_1^s >, \cdots, < x_m^s, y_m^s >\} \quad 1 \leq i \leq n, \tag{6}$$

for a given set of m grid locations ($m \geq n$). The constraint in (1) ensures that two blocks will not be assigned to the same grid location.

Occasionally it is useful to fix some of the block positions and/or orientations before applying the placement algorithms. This may be done both in order to speed up the convergence of the placement algorithm and in order to incorporate additional constraints. An example of such an application is a case where the placement has to fit into a given bounded area. By fixing small blocks around the perimeter of the designated area we can ensure that other blocks will not "jump outside" the boundary. This is easily handled in this model by changing the indexing of the variables. In the next section we show how to replace the constraints by penalty functions.

3 Our Penalty Function Approach

To handle the constraints (1) we use a penalty function approach and reformulate the placement problem as follows: find x_i^*, y_i^*, θ_i^* for $i = 1, \ldots, n$ such that

$$Z = D + \lambda_1 V \tag{7}$$

is minimized. Here D is the sum of the squares of the interconnection lengths as before; V is a penalty function term designed to ensure that blocks do not overlap each other; and λ_1 is a parameter, or weight used to define the relative importance of the penalty term at different stages in the solution process.

Sometimes the penalty function

$$V = \lambda \sum_{\substack{i,j \in B \\ i \neq j}} V_{ij} \tag{8}$$

where

$$V_{ij} = [max((\sigma_{ij} - d_{ij}), 0)]^2 \tag{9}$$

and

$$d_{ij} = distance(Block_i, Block_j) \tag{10}$$

is used in the literature [34]. Thus $V_{ij} > 0$ and V_{ij} is sensitive to the amount of violation of the $ij - th$ constraint, as long as that constraint is violated. Moreover, $V_{ij} = 0$ as soon as the constraint is satisfied, no matter how much larger d_{ij} is than σ_{ij}. An advantage of this is that when constraint set (1) is satisfied,

$$D = D + \lambda_1 V \tag{11}$$

which means that the original problem $P1$ is indeed solved. A disadvantage is that a sequence of problems has to be solved in a continuation chain and a certain ill-conditioning is introduced because it is necessary to successively increase λ such that $\lambda \rightarrow \infty$ in order to ensure feasiblity.

Instead we use a variation of the barrier potential function

$$V_{ij} = 4\epsilon \left[\left(\frac{\sigma_{ij}}{d_{ij}} \right)^4 - \left(\frac{\sigma_{ij}}{d_{ij}} \right)^2 \right]. \tag{12}$$

Observing Figure 3 we note that equation (12) has the properties of both smoothness and rapid increase of the penalty when $d_{ij} < \sigma_{ij}$. Also note that this penalty function automatically eliminates far-away inactive constraints, that it is bounded from below, and that the optimum of (7) is slightly shifted from the boundary and into the interior of the feasibility region of

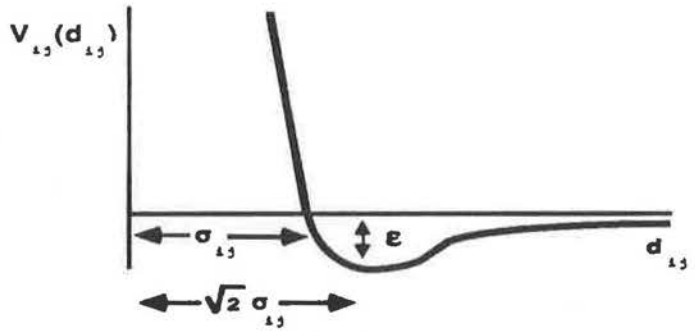

Figure 3: Graph of V_{ij} of (12)

D. Furthermore, as we will see in the next section, it usually suffices to use only a few continuation steps.

A price paid for using (12) instead of (9) is that the objective function is nonconvex away from the optimum. The optimization problem is also relaxed, i.e., at optimum equation (11) does not hold, and we are not guaranteed that an optimum of $P1$ is achieved, unless $\lambda_1 = 0$. Still, the solution is good because the minimum of (12) is at $d_{ij} = \sqrt{2}\sigma_{ij}$ which is not far from the feasibility edge $d_{ij} = \sigma_{ij}$. Note that since our block separation α is defined heuristically (since we do not know how much space is needed for routing), there is no need to insist on the exact solution to $P1$. Note also that one can relax the problem using (9) as well by bounding λ.

The physical analogy to potential energy is made when we note that equation (12) is a variation on the Lennard-Jones 6-12 potential equation [31]. If we think of the blocks i and j as "charged particles" then V_{ij} is a measure of the potential energy between the particles, which is very high when the particles are very close together and almost zero when they are far apart. Observing the graph of V_{ij} in Figure 3, we note that the (negative)

minimum is achieved at $d_{ij} = \sqrt{2}\sigma_{ij}$ and that V_{ij} grows rapidly when d_{ij} is below σ_{ij} and $V_{ij} \to +\infty$ as d_{ij} approaches 0. If we now take σ_{ij} to be as in equation (4) we practically ensure that those blocks do not overlap. This type of penalty function is sometimes called a *barrier* function [24]. Of course a very large number of interconnections may result with overlaps, but this is easily taken care of by changing equation (4) to read

$$\sigma_{ij} = C_{ij}(\sqrt{w_i^2 + l_i^2} + \sqrt{w_j^2 + l_j^2} + \alpha) \tag{13}$$

for some $C_{ij} > 1$, say $C_{ij} \approx \sum_{k,l=1}^{n} C_{ijkl}$ proportional to the total weight of interconnections between block i and block j.

Note that the 4ϵ in equation (12) can be grouped with λ_1. Alternatively we can also make $\epsilon = \epsilon_{ij}$ a parameter such that at its minimum point $V_{ij} = \epsilon_{ij}$. We can, therefore, adjust ϵ_{ij} to reflect the importance of having any two blocks being near each other.

To formulate an equality constraint like (5) there are two approaches, *discrete* and *continuous*. The discrete approach would yield a mixed (non-linear) integer programming problem and a branch and bound algorithm could be used for its solution, where on each leaf of the search tree a problem like the original $P1$ (but smaller) is solved. This is not very efficient and we again resort to a continuous formulation. To obtain an unconstrained problem we again use a penalty function, R, and minimize

$$Z = D + \lambda_1 V + \lambda_2 R \tag{14}$$

where

$$\begin{aligned} R &= \sum_{i \in B} R_i \\ &= \sum_{i \in B} -COS(m\theta_i). \end{aligned}$$

We note that the function $R_i = -COS(m\theta_i)$ achieves minimum value at each angle $\theta_i = \frac{2k\pi}{m}$ for $k = 0, \ldots, m-1$. With a large enough weight λ_2, we can make θ_i for $i = 1, \ldots, n$ arbitrarily close to the allowed angles. The choice of a smooth penalty function with moderate slopes is commensurate with the use of the algorithm to find approximate solutions within an interactive design process. For $m \rightarrow \infty$ the problem becomes "closer" to continuous. Note that in contrast to the constraints (1), the constraints (5) are not in direct competition with the objective of minimizing the wire length. This suggests that different penalty functions (loss functions) should be used for the different constraints. We may have to fix the angles θ_i so obtained exactly afterward, but this should not offer any practical difficulty.

Unfortunately we cannot use the same methodology of continuous formulation and penalty functions for the constraints (6). While it is clear that in equation (14) we could enforce feasibility by making λ_1 and λ_2 large enough, this is not the case for a penalty function representing (6). The problem here is that the constraint sets (1) and (6) may be in direct competition. If, for example, a grid point falls near the center of gravity of a subset of the blocks, all these blocks will be "attracted" to this grid point while they still repel one another. We have chosen to use a discrete approximation which will map the placement resulting from minimizing equation (14) onto grid locations, as described in [11,30].

If we use the model given by problem $P2$ then by this mapping we may lose the good orientations which were achieved in the continuous optimization. We can now fix all the variables $\{x_i, y_i,\ \ 1 \leq i \leq n\}$ to their assigned grid points and optimize $P2$ again only over the variables $\{\theta_i,\ \ 1 \leq i \leq n\}$.

To get a rough measure of the computational complexity of the method we may consider the cost of function evaluations. Note that for n blocks with a total of m interconnections the complexity of evaluating the objective function is $O(n^2 + m)$. Evaluating the first partial derivatives in each iteration can also be seen to cost $O(n^2 + m)$ operations, which results in a complexity of $O(n^2+m)$ for each iteration. Since the number of iterations is independent of n (the quasi-Newton method converges sufficiently fast)and $m = O(n^2)$, we get an overall complexity of $O(n^2)$ for our placement algorithm. From the construction of D, V, R note the possibility of applying $\frac{n(n-1)}{2}$ parallel processors for calculating the D_{ij} and V_{ij} terms in constant time and applying n parallel processors for calculating the R_i terms.

4 The Solution Strategy

The unconstrained minimization problem with which we have ended up is still nontrivial. Particularly difficult is finding a *global* minimum for $P1, P2$ or $P3$. For the solution process we use continuation. The question we are faced with now is how to choose two sequences of values $\lambda_1^1, \lambda_1^2, ..., \lambda_1^m$ and

$\lambda_2^1, \lambda_2^2, ..., \lambda_2^m$ such that minimizing the sequence of problems

$$\begin{aligned} Z_1 = &\quad D + \lambda_1^1 V + \lambda_2^1 R \\ Z_2 = &\quad D + \lambda_1^2 V + \lambda_2^2 R \\ &\quad ... \\ Z_m = &\quad D + \lambda_1^m V + \lambda_2^m R \end{aligned} \tag{15}$$

where the placement resulting from minimizing Z_i is the starting point for minimizing Z_{i+1}, will result in identifying a local minimum which is as close as possible to the global minimum.

We are looking for effective values for λ_1 and λ_2. We note that for any $\lambda_1 > 0$ the term $\lambda_1 V_{ij}$ will act as a penalty function allowing the blocks i and j to come arbitrarily close to each other as $\lambda_1 \rightarrow 0$ but not to slide over each other since $V_{ij} \rightarrow \infty$ as $d_{ij} \rightarrow 0$. Setting $\lambda_1 > 0$, therefore, will not result in a significant change in the *topology* of the placement. We control the scaling of the parameters by insisting that the placement be done in a unit square. This means that wire length, for example, cannot exceed $\sqrt{2}$ and $D \leq \sqrt{2}m$ where m is the total number of interconnecting nets. We then try $\lambda_1^i \in \{0, 1\}$ only.

Looking for possible discrete values for the λ_2^i parameters, we first note that the R functions are bounded, namely $-n \leq R \leq n$, which means that the penalty term is scaled as $-\lambda_2^i n \leq \lambda_2^i R \leq \lambda_2^i n$. If λ_2^i is large enough such that

$$\lambda_2^i R >> D + \lambda_1^i V \tag{16}$$

holds, then the blocks would tend to get fixed in the nearest allowed angle, since the $\lambda_2^i R$ term assumes more importance in the minimization (which does not necessarily coincide with the best allowed orientation). The value

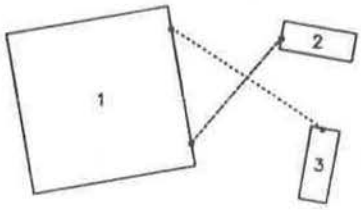

Figure 4a: Initial

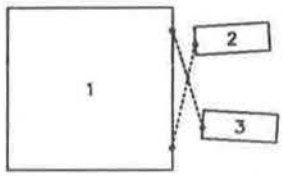

Figure 4b: Final

Figure 4: 3 block placement using $\lambda_1^m = \lambda_2^m = 1$

for which this happens depends on the length of the interconnections, and was, for the problems we have tried, roughly $\lambda_2^i \approx 1.0$. We therefore used λ_2^i values in the interval $[0, 1]$.

One simple strategy in minimizing Z is to let $m = 1$ and $\lambda_1^m = \lambda_2^m = 1$. Then we minimize

$$Z_m = Z = D + V + R. \tag{17}$$

As the penalty function V is turned on, blocks are not allowed to slide over one another. As the penalty function R is also turned on, blocks tend to get fixed in the nearest allowed orientation. The improvement over the initial placement, therefore, may not be significant and depends strongly on the initial placement of the blocks. Figure 4 presents a simple 3 block placement problem and the result of optimizing with $\lambda_1^m = \lambda_2^m = 1$ and multiples of $\frac{\pi}{2}$ as the allowed angles. This is clearly far from the global minimum.

A far better strategy is to use the following minimization sequence:

$$\begin{aligned} Z_1 &= D \\ Z_2 &= D + V + \epsilon_1 R \\ Z_3 &= D + V + \epsilon_2 R \\ &\cdots \\ Z_m &= D + V + R \end{aligned} \tag{18}$$

with $0 < \epsilon_1 < \epsilon_2 < ... < \epsilon_{m-2} < 1$. Note that the elimination of the penalty terms in Z_1 of (18) allows the blocks to slide over one another, therefore resulting in overlaps among the blocks, but also in a much better topological placement. This is really solving the "relaxed" problem which occurs when we solve for the original objective of $P1$ ignoring all the constraints. Note also that in contrast to other placement methods [18,30], since blocks occupy *area* and are not reduced to points (D is measured from the terminals and not from the centers) the event of collapsing all blocks into a single point is very unlikely. Finally note that since Z_1 of (18) is convex, its global minimum can be solved for.

In minimizing $Z_1, ..., Z_n$ in sequence we increment λ_2 from 0 to 1, first giving the blocks full freedom of rotation and then restricting them to the allowed angles only. Figure 5 presents the sequence of solution steps for the simple 3 block placement problem of Figure 4 starting from the initial placement of Figure 4a, and the result of optimizing using this strategey with multiples of $\frac{\pi}{2}$ as the allowed angles. This is clearly a global minimum for this problem. Note that block 3 has been rotated into its optimal orientation, and that the two blocks 2 and 3 have slid over each other into their optimal positions.

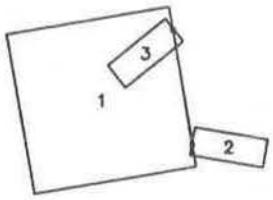

Figure 5a: $\lambda_1 = \lambda_2 = 0.0$

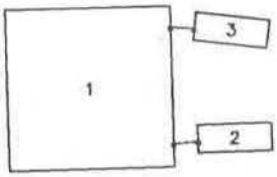

Figure 5b: $\lambda_1 = 1, \lambda_2 = 0.001$

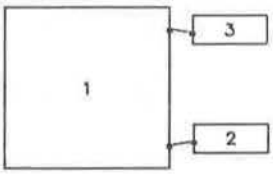

Figure 5c: $\lambda_1 = 1, \lambda_2 = 1$

Figure 5: 3 block placement using improved strategy

We note that a more gradual continuation sequence in λ_2^i than in λ_1^i is sometimes needed. An example is provided in Table 4.1 and Figure 4.3 of [1]. Unfortunately, we do not know of any automated way to produce the optimal sequence of values for λ_2^i. Various heuristics can be employed here, such as applying binary search to λ_2, or incrementing it with a fixed small value. However we have found experimentally that only few iterations with $\lambda_2^i = 10\lambda_2^{i-1}$ are usually sufficient for good placements.

As the experimental results of the next section indicate, our strategy proves useful, and in particular minimizing $Z = D$ gives an excellent first step. This is reasonable since the problem with $Z = D$ is convex [1].

PROBLEM	*FIGURE*	*#FUNCTIONS*
$Z_1 = D$	6*b*	114
$Z_2 = D + V + 0.001R$	6*c*	79
$Z_3 = D + V + 0.01R$	6*d*	33
$Z_4 = D + V + 1.0R$	6*e*	37

Table 1: Continuation sequence for Figure 6

5 Experimental Results

We have implemented an experimental program on a SUN workstation running UNIX. In our program we calculate D, V and R and the gradient vector for the variables. We use a canned routine, FNMIN, which implements the Variable Metric Method based on the quasi-Newton approach for the unconstrained minimization of nonlinear functions [10].

Various placement problems were tested. Figure 6 presents a 6 block placement problem (allowed angles which are multiples of $\frac{\pi}{2}$) solved in stages with the continuation sequence of Table 1. A measure of computational complexity is $\#FUNCTIONS$ which is the number of Z_i function evaluations before convergence is reached. Note that for this example $\lambda_2^i = 0.01$ was already large enough to fix the blocks, so there is no noticeable difference between Figure 6d and Figure 6e. Figure 7 presents the final result when angles which are multiples of $\frac{\pi}{4}$ are allowed. The number of function evaluations was roughly the same as in Table 1. We found that for all the block configurations tested the continuation sequence $\lambda_2^i = 0.0, 0.001, 0.01, 0.1, 1.0$ was sufficient to achieve good orientation in a placement problem with a

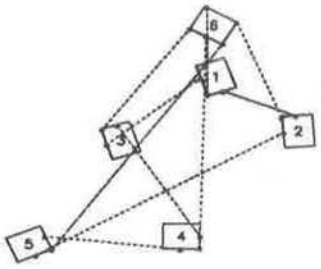

Figure 6a: Initial

Figure 6b: $\lambda_1 = 0.0, \lambda_2 = 0.0$

Figure 6c: $\lambda_1 = 1.0, \lambda_2 = 0.001$

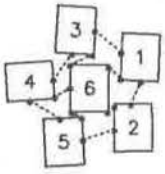

Figure 6d: $\lambda_1 = 1.0, \lambda_2 = 0.01$

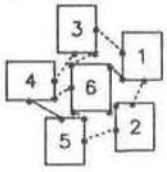

Figure 6e: $\lambda_1 = 1.0, \lambda_2 = 1.0$

Figure 6: 6 block placement

fixed set of allowed angles.

Figure 8 presents a 10 block placement problem (allowed angles which are multiples of $\frac{\pi}{2}$) solved in stages with the same strategy as above, resulting in the optimization sequence of Table 2. Note that although Figures 8d and 8e are almost identical there is a computational effort involved in solving Z_4. This is the result of additional degrees of freedom which cause the final convergence to be at a slower rate. One way to overcome this problem is to fix the placement of some blocks before the optimization is done. Of course such an arbitrary fixing of some blocks may result in a less favourable placement than is otherwise possible.

To further support the claim for good placement achievable by this strategy, we have applied it to the above 6 and 10 blocks placement problems

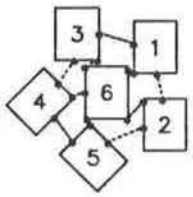

Figure 7: 6 block placement with 45^0 angles

PROBLEM	*FIGURE*	*#FUNCTIONS*
$Z_1 = D$	8*b*	73
$Z_2 = D + V + 0.001R$	8*c*	129
$Z_3 = D + V + 0.01R$	8*d*	53
$Z_4 = D + V + 1.0R$	8*e*	61

Table 2: Continuation sequence for figure 8

with 10 random starts generated for each problem. The random starts were generated by randomly setting x_i, y_i, θ_i for $i = 1, \ldots, n$ at the initial placement before optimizing Z_1. In both problems we have found that all 10 trial starts have converged to the same minimum. We have also noticed that the convergence to the same placement occurred very early in the sequence, after minimizing $Z_1 = D$. Unfortunately, this "globally convergent" quality of the algorithm does not always work. Figure 9 presents a simple example of such a case. The difficulty illustrated here is that the blocks may not be able to slide over each other at the first stage when optimizing $Z_1 = D$, if they have reached the minimum in such an orientation that their *centers* are still placed on the "wrong" sides. Figures 9d-9f show how a different

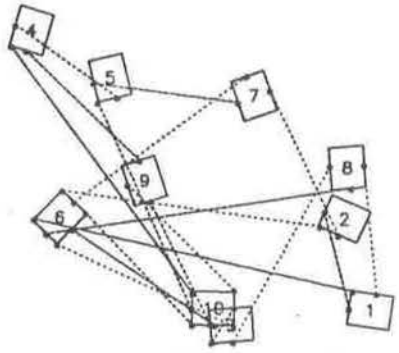

Figure 8a: Initial

Figure 8b:$\lambda_1 = \lambda_2 = 0.0$

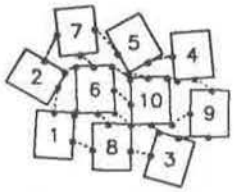

Figure 8c:$\lambda_1 = 1.0, \lambda_2 = 0.001$

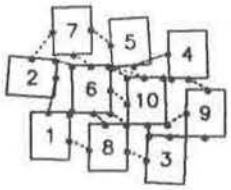

Figure 8d: $\lambda_1 = 1.0, \lambda_2 = 0.01$

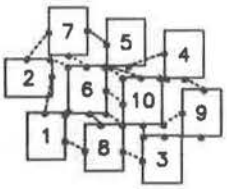

Figure 8e: $\lambda_1 = \lambda_2 = 1.0$

Figure 8: 10 block placement

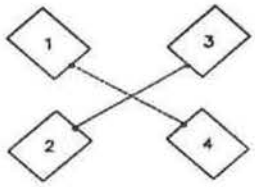

Figure 9a: Initial

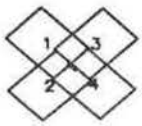

Figure 9b:
$Z_1 = D$

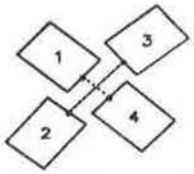

Figure 9c: Final

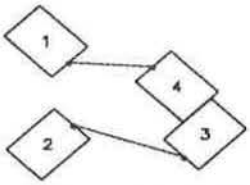

Figure 9d: Initial

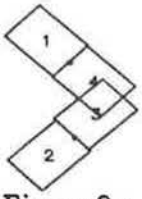

Figure 9e:
$Z_1 = D$

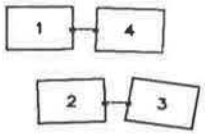

Figure 9f: Final

Figure 9: Placement problem where initial placement is important

starting position leads to an optimal placement.

We have also experimented with the fixed slots problem. Figures 10a-f present the result of fixing the final placement of Figure 8 into the given slots, using the continuation optimization sequence of Table 3. Figure 10c shows the result of letting the 10 blocks rotate freely again and then fixing their angles using the above sequence. Note that block 5 was rotated from its initial position after fixing the blocks into a better fixed orientation.

Figure 11 presents our best result on the Steinberg problem. This problem consists of placing 34 blocks with a total number of 2,620 connections on a 4 by 9 grid. The original specification of the problem, including the connectivity matrix, can be found in [35]. Distances are measured as one unit between neighboring grid points. Other researchers [15,12,14,2,16,6,7] have

PROBLEM	*FIGURE*	*#FUNCTIONS*
$Z_1 = D$	10*c*	25
$Z_2 = D + 0.001R$	10*d*	22
$Z_3 = D + 0.01R$	10*e*	29
$Z_4 = D + 1.0R$	10*f*	23

Table 3: Continuation sequence for Figure 10

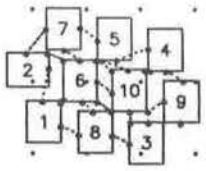

Figure 10a: Initial

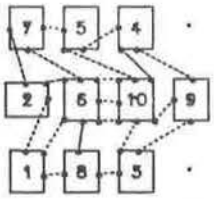

Figure 10b: In Slots

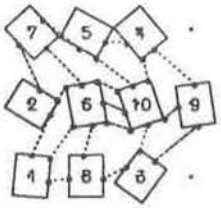

Figure 10c: $\lambda_2 = 0.0$

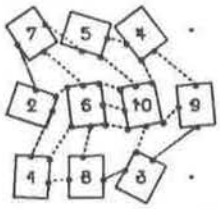

Figure 10d: $\lambda_2 = 0.001$

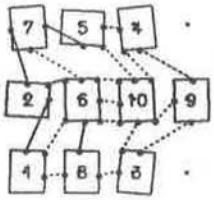

Figure 10e: $\lambda_2 = 0.01$

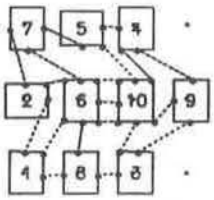

Figure 10f: Final

Figure 10: 10 Block placement into fixed slots

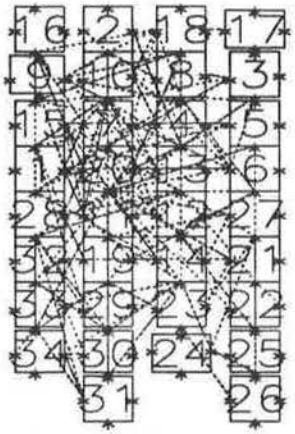

Figure 11: Steinberg example

used the Manhattan norm l_1, the Euclidian norm l_2 and the Euclidian norm squared l_2^2 to measure the wire length. Since we represent the placement problem in more detail, we had to change the problem by adding terminals to the blocks and routing the connections through the terminals. We have arbitrarily added four terminals to each block and connected the wires to one of the terminals. Steinberg's best placement results are summarized in Table 4 under the row labeled "Steinberg".

Our best results are better than any previously known in all norms. Table 4 presents two of our best placements and Figure 11 gives the resulting placement for the first of the two. We note that experimenting with various initial configurations, 4 by 9 grid spacing and fixed blocks, we were able to find many placement configurations with values near our minimum.

We have found that although fixing some of the variables may speed up the convergence of the optimization routines, it may also result in subopti-

Author	l_1	l_2	l_2^2
Carter & Breuer	*N.A.*	5575	*N.A.*
Hillier & Connors	*N.A.*	4821.7	10929.9
Grave & Whinston	*N.A.*	4490	11909
Gilmore(n^4 algorithm)	*N.A.*	4547.5	10656
Gilmore(n^5 algorithm)	*N.A.*	4680.3	11929
Steinberg	*N.A.*	4894.5	11875
Bazaraa	*N.A.*	4800	*N.A.*
Hall	5139	4419.1	9699
Cheng − Kuh	5316	4358.4	8596
Ours(1)	5016	4271.8	8552
Ours(2)	5225	4296.2	8475

Table 4: Steinberg example

mal placements. Regarding run time we have found that small placement problems ($n \leq 15$) can be solved interactively on a SUN workstation (without floating point hardware). We expect much larger problems ($n \approx 100$) to be solvable efficiently on a mainframe computer.

6 Conclusions

Our nonlinear model for the placement problem has proven successful in overcoming two of the limitations of previous force directed models. The potential function ensures that blocks do not overlap, and the calculation of distances from the terminals instead of from the centers, forces the blocks to rotate, resulting in a better placement. Our model allows for the handling of fixed and movable blocks, and for convenient control over the separation between blocks via the parameter σ_{ij} in the potential function. The

quasi-Newton method used in the penalty approach, being superlinearly convergent, ensures reasonable running times.

We have also implemented the penalty functions and transformations described, allowing restrictive forms of the placement problem to be solved in our system. Our experience with the minimization sequence has resulted in good placements. We further conclude that our optimization procedure is robust, being fairly independent of the initial placement.

We have conducted some simple experiments and a larger one and our results compare favourably with previous results.

References

[1] A. Alon, , "Model and Solution Strategy for Placement of Rectangular Blocks in the Euclidian Plane," M.Sc. Thesis, Department of Computer Science, Univ. of B.C., Vancouver, Canada 1986.

[2] M.S. Bazaraa O. Kirca, "A Branch-and-Bound-Based Heuristic for Solving the Quadratic Assignment Problem," *Naval Res. Log. Quarterly* Vol. 30, pp. 287-304, 1983.

[3] J.P. Blanks "Initial Placement of Gate Arrays Using Least-Squares Methods," Proc. 21^{th} Design Automation Conference. pp. 670-671, June 1984.

[4] J.P. Blanks "Near-Optimal Placement Using A Quadratic Objective Function," Proc. 22^{th} Design Automation Conference. pp. 609-615, June 1985.

[5] M.A. Breuer "A Class of Min-Cut Placement Algorithms," Proc. 13^{th} Design Automation Conference. pp. 284-289, June 1976.

[6] H.W. Carter, M.A. Breuer, Z.A. Zyed, "Incremental Processing Applied to Steinberg's Placement Procedure," Proc. 16^{th} Design Automation Conference. pp. 26-31, June 1979.

[7] C. Cheng, K.S. Ernest, "Module Placement Based on Resistive Network Optimization," *IEEE Ttransaction on Computer-Aided Design,* Vol.3 #3 pp. 218-225, July 1984.

[8] D. Chyan, M.A. Breuer, "A Placement Algorithm for Array Processors," Proc. 20^{th} Design Automation Conference. pp. 182-188, June 1983.

[9] Z. Drezner, "DISCON: A New Method for the Layout Problem," *Operation Research,* Vol. 28, #6, pp. 1375-1384, 1980.

[10] R. Fletcher, "A New Approach to Variable Metric Algorithms," *Computer Journal,* Vol 13, pp. 317-322, 1970.

[11] K. Fukunaga, S. Yamada, H.S. Stone, T. Kasai, "Placement of Circuit Modules Using A Graph Space Approach," Proc. 20^{th} Design Automation Conference. pp. 465-471, June 1983.

[12] P.C. Gilmore, "Optimal and Suboptimal Algorithms for the Quadratic Assignment Problem," *J. Soc. Indust. Appl. Math.* Vol. 10, #2, pp. 305-313, 1962.

[13] S. Goto, "A Two-Dimensional Placement Algorithm for the Master Slice LSI Layout Problem," Proc. 16^{th} Design Automation Conference. pp. 11-17, June 1979.

[14] G.W. Graves, A.B. Whinston, "An Algorithm for the Quadratic Assignment Problem," *Management Science,* Vol. 17, #2, pp. 453-471, 1970.

[15] K.M. Hall, "An r-Dimensional Quadratic Placement Algorithm," *Management Science,* Vol. 17, #3, pp. 219-229, 1970.

[16] F.S. Hillier, M.M. Conners, "Quadratic Assignment Algorithms and the Locations of Indivisible Facilities," *Management Science*, Vol. 13, #1, pp. 42-57, 1966.

[17] S.C. Hoffman, *COMPUTER-AIDED DESIGN of digital electronic circuits and systems*, North-Holland, Brussels' pp. 229-236, 1979.

[18] A. Iosupovici, C. King, M.A. Breuer, "A Module Interchange Placement Machine," Proc. 20^{th} Design Automation Conference. pp. 171-174, June 1983.

[19] B.W. Kernighan, S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *The Bell System Technical Journal,* pp. 291-307, February 1970.

[20] I.H. Kirk, P.D. Crowhurst, J.A. Skingley, J.D. Bowman, G.L. Taylor, "Placement of Irregular Circuit Elements on Non-Uniform Gate Arrays," Proc. 20^{th} Design Automation Conference. pp. 637-643, June 1983.

[21] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, "Optimization by Simulated Annealing," *Science,* Vol. 220, #4598, pp. 671-680, May 1983.

[22] T. Kozawa, H. Terai, T. Ishii, M. Hayase, C. Miura, Y. Ogawa, K. Kisida, N. Yamada, Y. Ohno, "Automatic Placement Algorithms for High Packing Density VLSI," Proc. 20^{th} Design Automation Workshop. pp. 175-181, June 1983.

[23] H.W. Kuhn, "The Hungarain method for the Assignment Problem," *Naval Res. Log. Quarterly*, Vol. 2, pp. 83-97, 1955.

[24] F.A. Lootsma, "A Survey of Methods for Solving Constrained-Minimization Problems via Unconstrain Minimization," in Optimization and Design, Premtice-Hall Inc, Englewood Cliffs, N.J., pp. 88-113, 1971.

[25] R.F. Love, J.Y. Wong, "Solving Quadratic Assignment Problems with Rectangular Distances and Integer Programing," *Naval Res. Log. Quarterly*, Vol. 24, pp. 623-627, 1976.

[26] L.A. Markov, J.R. Fox, J.H. Blank, "Optimization Techniques for Two Dimensional Placement," Proc. 21^{th} Design Automation Conference. pp. 652-654, June 1984.

[27] G. Odawara, K. Iijima, K. Wakabayashi, "Knowledeg-Based Placement Technique for Printed Circuit Boards," Proc. 22^{th} Design Automation Conference. pp. 616-622, June 1985.

[28] B.T. Preas, W.M. VanCleemput, "Placement Algorithms for Arbitrarily Shaped Blocks," Proc. 16^{th} Design Automation Conference. pp. 474-480, June 1979.

[29] J.F. Pierce, W.B. Crowston, "Tree Search Algorithms for Quadratic Assignment Problems," *Naval Res. Log. Quarterly*, Vol. 18, pp. 1-36, 1971.

[30] N.R. Quinn, M.A. Breuer, "A Force Directed Component Placement Procedure for Printed Circuit Boards," *IEEE Transaction on Circuits and Systems,* Vol.26, #6, pp. 377-388, June 1979.

[31] L. E. Reichl. *A Modern Course in Statistical Physics.* p. 363, University of Texas Press, 1980.

[32] B.D. Richard, "A Standard Cell Initial Placement Strategy," Proc. 21^{th} Design Automation Conference. pp. 392-398, June 1984.

[33] H.J. Rothermel, D.A. Mlynski. "Routing Method for VLSI Design Using Irregular Cells," Proc. 20^{th} Design Automation Conference. pp. 257-262, June 1983.

[34] L. Sha, R.W. Dutton, "An Analytical Algorithm for Placement of Arbitrarily Sized Rectangular Blocks," Proc. 22^{th} Design Automation Conference. pp. 602-607, June 1985.

[35] L. Steinberg, "The Backboard Wiring Problem: A Placement Algorithm," *SIAM Review*, Vol. 3, #1, pp. 37-50, January 1961.

[36] M.P. Vecchi, S. Kirkpatrick, "Global Wiring by Simulated Annealing," *IEEE Transaction on Computer-Aided Design*, Vol.2, #4, pp. 215-222, October 1983.