A UNIFIED APPROACH TO CIRCUIT PARTITIONING AND PLACEMENT

by

Ren-Song Tsay and Ernest S. Kuh

Memorandum No. UCB/ERL M86/8

27 January 1986

COVER PAGE

A UNIFIED APPROACH TO CIRCUIT PARTITIONING AND PLACEMENT

by

Ren-Song Tsay and Ernest S. Kuh

Memorandum No. UCB/ERL M86/8

27 January 1986

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A Unified Approach to Circuit Partitioning and Placement

Ren-Song Tsay and Ernest S. Kuh
Department of EECS and the Electronics Research Laboratory
University of California, Berkeley

ABSTRACT

In this paper we introduce a unified approach to the problems of circuit partitioning and placement. By using the matrix inner product formulation, circuit partitioning and placement can be transformed into the standard quadratic programming. It is thus possible to search for the optimum solution by applying the eigenvector approach.

For both partitioning and placement, the solution is a permutation of some positions, and the solution space is found located on an $(n-1)$-dimensional supersphere. A simple matching method combining with vector projection is devised to search for the solution.

The algorithm discussed here is a constructive method which can be followed by any iterative procedure such as pairwise-interchange if needed. Experimental results indicate that this constructive method yields excellent results.

1. Introduction

Partitioning and Placement are major steps in IC packaging and layout design. In this paper we give a unified formulation for partitioning and placement and establish the relationship between them.

Research sponsored by the National Science Foundation, Grant ECS-85-06901 and the State of California Microelectronics Innovation and Computer Research Opportunities Program

Both partitioning and placement problems are known to be N-P complete. Many heuristic algorithms have been created to attack these problems.

Our approach depends on the 'matrix inner product' which along with some preliminaries are discussed in Section 2. The general formulation of our approach is given in Section 3.

The most useful aspect of the 'matrix inner product' is that it reduces both the partitioning and the placement problems to the quadratic programming problem. Because the connectivity matrix is always symmetric, there exist efficient programs to find the optimum solution on an unrestricted solution space. But the main obstacle in discrete optimization problems lies in the incomplete solution space. Our approach is to use a simple algorithm to choose the closest point to the unrestricted optimum solution from the discrete solution space. In Section 4, more details are presented. Some experimental results are shown in Section 5.

2 Matrix Inner Product

Consider two $n \times n$ matrices A and B. Let $E_{ij} := e_i e_j^T$ be the basis of $R^{n \times n}$, where $e_1 , e_2 , \cdots , e_n$ is an orthonormal basis of R^n. Then

$$A = \sum_{i,j=1}^{n} \alpha_{ij} E_{ij} \quad , \quad B = \sum_{i,j=1}^{n} \beta_{ij} E_{ij} \tag{2.1}$$

We define the matrix inner product as

$$< A , B >_M := \sum_{i,j=1}^{n} \alpha_{ij} \beta_{ij} \tag{2.2}$$

The subindex, M, is used to denote specifically for matrix inner product throughout this paper.

Now consider the case where B is a summation of a series of outer products.

$$B = \sum_{k=1}^{m} u_k \; v_k^T = UV^T \tag{2.3}$$

Then

$$< A , B >_M = < A , UV^T >_M = \text{trace}(U^T A \; V) \tag{2.4}$$

where 'trace' is defined as the summation of all diagonal elements.

For the special case, $U=V=X$, we then have

$$< A \ , X X^T >_M = \text{trace} \ (X^T A X) \tag{2.5}$$

If we let $B = X X^T$, we conclude that to optimize $< A \ , B >_M$ is the same as to optimize trace ($X^T A X$).

3. A General Formulation

3.1 Permutation

In a discrete optimization problem, we can always obtain the solution. by exhaustive permutation. We define the permutation matrix as

$$E = [\ e_{\pi(1)} \ , \ e_{\pi(2)} \ , \ \cdots \ , \ e_{\pi(n)} \ ]. \tag{3.1}$$

thus $E \ C \ E^T$ is a permutation of the matrix C. More explicitly, the element c_{ij} of matrix C is now changed to $c_{\pi(i)\pi(j)}$.

Now we formulate the partitioning and the placement problem as follows. At the outset we restrict ourselves to 2-terminal nets, thus multi-terminal nets must be preprocessed using any appropriate methods such as a weighted clique. Two matrices are given: a matrix C whose element c_{ij} represents the connectivity or cost between module i and module j, and another symmetric matrix D called the distance matrix. D is a partition matrix in the case of a partitioning problem. It is a Manhattan matrix or a Euclidean matrix in the case of a placement problem. These will be defined later in this section. Then the optimization in partitioning and placement amounts to minimizing the objective function:

$$L = < E \ C \ E^T \ , D >_M \tag{3.2}$$

with respect to the permutation matrix E.

By the adjoint operation, we have

$$< E\ C\ E^T\ , D >_M = < C\ , E^T\ D\ E >_M \tag{3.3}$$

Let $\tilde{E} = E^T = E^{-1}$, we have

$$< E\ C\ E^T\ , D >_M = < C\ , \tilde{E}\ D\ \tilde{E}^T >_M \tag{3.4}$$

or

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{\pi(i)\pi(j)}\ d_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}\ d_{\tilde{\pi}(i),\tilde{\pi}(j)} \tag{3.5}$$

This equation means that we can solve the problem by permuting either the vertices representing the modules or the positions of the modules to be assigned. Both will give the same result.

In the following subsections, we are going to discuss different optimization criteria for different distance matrix D. For convenience, from now on we use $\hat{C}$ and $\hat{D}$ to represent the optimum matrices after the permutation of the matrices C and D respectively. That is $\hat{C} = E\ C\ E^T, \hat{D} = \tilde{E}\ D\ \tilde{E}^T$.

3.2 The Partition Matrix

In partitioning problems, we want to separate the vertices into several groups such that the total number of interconnections among the groups is a minimum. Let us define the partition matrix P whose element p_{ij} is

$$p_{ij} = \begin{cases} 1 & \text{if vertices i \& j belong to the same group} \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

For example, for a 2-way partition matrix,

$$P = \begin{bmatrix} 1\ 1 & \cdots & 1\ 0\ 0 & \cdots & 0 \\ \cdot & \cdots & \cdot & \cdots & \cdot \\ 1\ 1 & \cdots & 1\ 0\ 0 & \cdots & 0 \\ 0\ 0 & \cdots & 0\ 1\ 1 & \cdots & 1 \\ \cdot & \cdots & \cdot & \cdots & \cdot \\ 0\ 0 & \cdots & 0\ 1\ 1 & \cdots & 1 \end{bmatrix}$$

Let us also define an all-one matrix

$$\mathbf{I} := \begin{bmatrix} 1\ 1 & \cdots & 1 \\ 1\ 1 & \cdots & 1 \\ \cdot & \cdots & \cdot \\ 1\ 1 & \cdots & 1 \end{bmatrix}$$

Then the distance matrix, D_P, for the partitioning problem is simply

$$D_P = \mathbf{I} - P \tag{3.7}$$

Since c_{ij} is the number of edges between vertices i and j, then it is esaily shown that $< \hat{C} , \mathbf{I} >_M = 2N$ [1], where N is the total number of edges. Also $< \hat{C} , P >_M = 2N_{nc}$, where N_{nc} is the total number of inner connection, $< \hat{C} , D_P > = 2N_c$, where N_c is the total number of interconnection between each group. Since

$$2N = 2N_{nc} + 2N_c \tag{3.8}$$

we have

$$< \hat{C} , \mathbf{I} >_M = < \hat{C} , P >_M + < \hat{C} , D_P >_M \tag{3.9}$$

For a given C, $< \hat{C} , \mathbf{I} >_M$ is fixed. Thus, to minimize the interconnection $< \hat{C} , D_P >_M$ is equivalent to maximize the inner cost $< \hat{C} , P >_M$. We will find later that maximizing $< \hat{C} , P >_M$ is easier and gives more insight to the problem.

The partition matrix P can also be decomposed into vector outer product. For a k-way partition

$$P = \sum_{i=1}^{k} q_i\, q_i^T = Q\, Q^T \tag{3.10}$$

where

$$Q = [\ q_1\ ,\ q_2\ ,\ \cdots\ ,\ q_k\ ] \tag{3.11}$$

$$q_i = \begin{bmatrix} 0 \\ \cdot \\ 0 \\ 1 \\ \cdot \\ 1 \\ 0 \\ \cdot \\ 0 \end{bmatrix}$$

Let $\tilde{q}_i = \tilde{E}\ q_i$ and $\tilde{Q} = \tilde{E}\ Q$, we have

$$\hat{P} = \tilde{E}\ P\ \tilde{E}^T = \sum_{i=1}^{k} \tilde{q}_i\ \tilde{q}_i\ ^T = \tilde{Q}\ \tilde{Q}^T \tag{3.12}$$

Note that for the matrix $\tilde{Q}$ there is only a single "1" on each row. According to (2.5)

$$< \hat{C}\ ,P >_M = < C\ ,\hat{P} >_M = \text{trace}\ (\ \tilde{Q}^T\ C\ \tilde{Q}\ ) \tag{3.13}$$

i.e. we have converted the problem to an integer valued quadratic programming problem [2].

3.3 The Manhattan Matrix

In some placement problem, the objective is to minimize the Manhattan length of total connections. Then the distance matrix is $D_M = [d_{ij}]$ where $d_{ij} := |\,i-j\,|$. We call D_M the 'Manhattan matrix'. Now the objective function is

$$L_M = < \hat{C}\ , D_M >_M = < C\ , \hat{D}_M >_M = \sum_{i,j=1}^{n} c_{ij}\ d_{\pi(i)\pi(j)} \tag{3.14}$$

It is not easy to solve this problem. However, we will see later that there are similarities among these three distance matrices , and the solutions are expected be near each other.

3.4 The Euclidean Matrix

For this case, the objective is to minimize the total Euclidean length for all interconnections. We define the 'Euclidean matrix' as

$$D_E = [\ d_{ij}^2\ ] \tag{3.15}$$

where $d_{ij}^2 := (i-j)^2$. Then the objective function is

$$L_E = < C\ , \hat{D}_E >_M = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}\ d_{\tilde{\pi}(i)\tilde{\pi}(j)}^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}\ (\ \tilde{\pi}(i) - \tilde{\pi}(j)\ )^2 \qquad (3.16)$$

We relax the integer requirement for $\tilde{\pi}(i)$. Let x_i be an unrestricted real number which takes place of the integer $\tilde{\pi}(i)$. The relaxed objective function is as follows:

$$L_E = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}\ (\ x_i - x_j\ )^2 \qquad (3.17)$$

Let B be the modified connectivity matrix [3], and the element of B is defined as

$$b_{ij} = \begin{cases} -c_{ij} & i \neq j \\ c_{i\bullet} & i = j \end{cases} \qquad (3.18)$$

where $c_{i\bullet} := \sum_{j=1, j\neq i}^{n} c_{ij}$. Then

$$L_E = x^T\ B\ x \qquad (3.19)$$

which is a quadratic form.

3.5 Similarity Among The Problems

Comparing the quadratic objective functions that come with the partition matrix and Euclidean matrix, we find that the only difference lie in the diagonal elements of the connectivity matrix C and the modified one B. Using Eq. (3.18), we have

$$< C\ , \hat{P} >_M + < B\ , \hat{P} >_M = \sum_{i=1}^{n} (\ c_{ii} + b_{ii}\ ) = \text{constant} \qquad (3.20)$$

This tells us that maximizing $< C\ , \hat{P} >_M$ is equivalent to minimizing $< B\ , \hat{P} >_M$ or alternatively using Eq. (3.9), minimizing $< C\ , \hat{D}_P >_M$.

To reenforce the similarity among the partition matrix, Manhattan matrix, and Euclidean matrix, we give an explicit form of each matrix in case of a 6-dimension, 2-way partition shown in Fig. (3.1).

$$D_P = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$D_M = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 4 & 3 & 2 & 1 & 0 & 1 \\ 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix}$$

$$D_E = \begin{bmatrix} 0 & 1 & 4 & 9 & 16 & 25 \\ 1 & 0 & 1 & 4 & 9 & 16 \\ 4 & 1 & 0 & 1 & 4 & 9 \\ 9 & 4 & 1 & 0 & 1 & 4 \\ 16 & 9 & 4 & 1 & 0 & 1 \\ 25 & 16 & 9 & 4 & 1 & 0 \end{bmatrix}$$

Fig. (3.1)

To extend the concept we discussed here, we propose a more general distance matrix, the l-matrix. For l-matrix

$$D_l = [\ d_{ij}^l\ ] \tag{3.21}$$

where $d_{ij}^l := |\,i-j\,|^l$. To our best knowledge, no one has tried this l-matrix as the distance matrix. It seems interesting to derive the effects of different choices of l.

3.6 The Geometric Model

A general placement problem is to assign n objects on n positions which is not necessarily uniformly spaced. For a general linear placement, the n positions $m_1 \leq m_2 \leq \cdots \leq m_n$ are ordered along the real line as shown in Fig.(3.2).

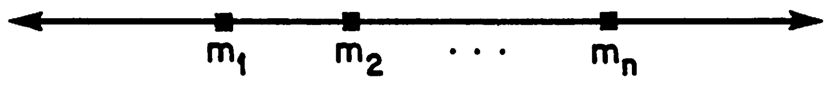

Fig.(3.2)

Now consider a 2-way partitioning problem. Suppose that we want to put k elements in one group and the remaining $(n-k)$ elements in the other group. Then the geometric representation is as shown in Fig.(3.3).

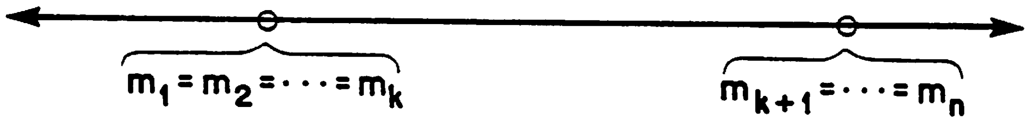

Fig.(3.3)

It is similar for a general 2-dimensional placement model. There are n positions on a 2-dimensional space as shown in Fig.(3.4).

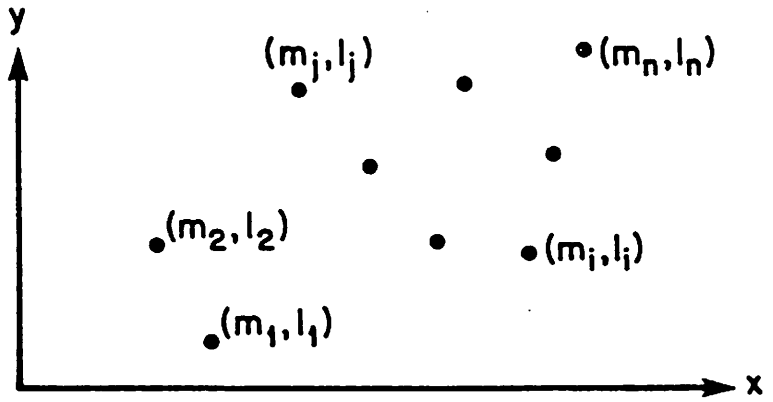

Fig.(3.4)

Then the 3-way partition problem becomes a special case of the 2-dimensional placement problem. The three groups are represented as the three vertices of a equilateral triangle as shown in Fig.(3.5).

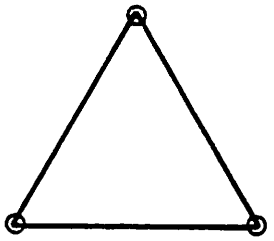

Fig.(3.5)

As in the last subsection, this general placement problem can be transformed into a quadratic programming problem.

$$\min\ L = x^T Bx + y^T By \tag{3.22}$$

where

$$x = (\ m_{\pi(1)}\ ,\ m_{\pi(2)}\ ,\ \cdots\ ,\ m_{\pi(n)}\ )$$

$$y = (\ l_{\pi(1)}\ ,\ l_{\pi(2)}\ ,\ \cdots\ ,\ l_{\pi(n)}\ ) \tag{3.23}$$

We list the constraints for discrete positions as follows[4]. In the x direction,

$$\sum_i x_i = \sum_i m_i = c_1$$

$$\sum_i x_i^2 = \sum_i m_i^2 = c_2$$

$$\cdots$$

$$\sum_i x_i^n = \sum_i m_i^n = c_n \tag{3.24}$$

In the y direction,

$$\sum_i y_i = \sum_i l_i = d_1$$

$$\sum_i y_i^2 = \sum_i l_i^2 = d_2$$

$$\cdots$$

$$\sum_i y_i^n = \sum_i l_i^n = d_n \tag{3.25}$$

Also we have to consider the correlation between x and y.

$$\sum_i x_i y_i = \sum_i m_i l_i = e_1$$

$$\sum_i x_i^2 y_i^2 = \sum_i m_i^2 l_i^2 = e_2$$

$$\cdots$$

$$\sum_i x_i^n y_i^n = \sum_i m_i^n l_i^n = e_n \tag{3.26}$$

The first order constraint is a restriction on the d.c level (center of gravity or average). The second order constraint corresponds to an a.c level (variance). These two are the properties of the position vectors. The interesting thing is that the modified connectivity matrix B happens to have the d.c removing property, i.e.

$$B\,\mathbf{1} = 0 \cdot \mathbf{1} \tag{3.27}$$

where $\mathbf{1} = [\ 1\ ,\ 1\ ,\ \cdots\ ,\ 1\ ]^T$.

Now, let $\bar{p}$ and $\tilde{p}$ be the d.c. and a.c components of the position vector p, respectively. Then

$$\bar{p} = < p\ ,\ \frac{\mathbf{1}}{|\mathbf{1}|} > \cdot \frac{\mathbf{1}}{|\mathbf{1}|} = \frac{c_1}{n} \cdot \mathbf{1} \tag{3.28}$$

$$|\,\tilde{p}\,|^2 = |\,p\,|^2 - |\,\bar{p}\,|^2 = c_2 - \frac{c_1^2}{n} = \text{constant} \tag{3.29}$$

This is illustrated in Fig.(3.6).

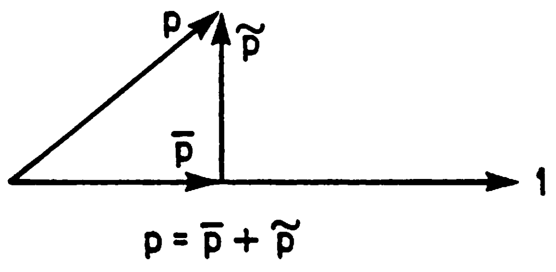

Fig.(3.6)

Armed with these facts, we can rewrite the optimization equation as follows.

$$\min\ L = x^T B x + y^T B y = \tilde{x}^T B \tilde{x} + \tilde{y}^T B \tilde{y}$$

subject to

$$\sum_i \tilde{x}_i = 0$$

$$\sum_i \tilde{x}^2_i = c_2 - \frac{c_1^2}{n}$$

$$\sum_i \tilde{y}_i = 0$$

$$\sum_i \tilde{y}_i^2 = d_2 - \frac{d_1^2}{n}$$

$$\sum_i \tilde{x}_i \cdot \tilde{y}_i = e_1 - \frac{c_1 d_1}{n} \qquad (3.30)$$

where only the first and second order constraints are considered.

In the case of linear placement with $m_i = i$, we have

$$\min\ L = \tilde{x}^T B \tilde{x}$$

subject to

$$\sum_i \tilde{x}_i = 0$$

$$\sum_i \tilde{x}_i^2 = \frac{n(n^2-1)}{12} \tag{3.31}$$

For a 2-way partitioning with k and $(n-k)$ elements in two separate groups, then we have

$$\min\ L = \tilde{x}^T B \tilde{x}$$

subject to

$$\sum_i \tilde{x}_i = 0$$

$$\sum_i \tilde{x}_i^2 = \frac{k(n-k)}{n} \tag{3.32}$$

Now we want to show mathematically that a 2-way partitioning is a special case of the linear placement problems. In 2-way partition, we have

$$q_1 = [\, 1 \,, \cdots , 1 \,, 0 \,, \cdots , 0 \,]$$

$$q_2 = [\, 0 \,, \cdots , 0 \,, 1 \,, \cdots , 1 \,] \tag{3.33}$$

Then

$$\tilde{q}_1 = -\tilde{q}_2 = [\, \frac{n-k}{n} \,, \cdots , \frac{n-k}{n} \,, \frac{-k}{n} \,, \cdots , \frac{-k}{n} \,] \tag{3.34}$$

Hence

$$L = q_1^T B\, q_1 + q_2^T B\, q_2 = 2 \cdot (\, \tilde{q}_1^T B\, \tilde{q}_1 \,) \tag{3.35}$$

That is exactly what we have in the general linear placement formulation.

Similarly, for 3-way partitioning,

$$q_1 = [\, 1 \,, \cdots , 1 \,, 0 \,, \cdots , 0 \,, 0 \,, \cdots , 0 \,]$$

$$q_2 = [\, 0 \,, \cdots , 0 \,, 1 \,, \cdots , 1 \,, 0 \,, \cdots , 0 \,]$$

$$q_3 = [\, 0 \,, \cdots , 0 \,, 0 \,, \cdots , 0 \,, 1 \,, \cdots , 1 \,] \tag{3.36}$$

We have

$$\tilde{q}_1 + \tilde{q}_2 + \tilde{q}_3 = 0 \tag{3.37}$$

or

$$\tilde{q}_3 = -(\tilde{q}_1 + \tilde{q}_2) \tag{3.38}$$

and then

$$L = 2(\tilde{q}_1^T B \tilde{q}_1 + \tilde{q}_2^T B \tilde{q}_2) \tag{3.39}$$

Again, this is precisely the same objective function as that of the general 2-dimensional placement.

4. The Searching Algorithm

4.1 Eigenvector Matching [3]

Recall from the last section, the objective function for linear placement based on minimal Euclidean length is

$$L = x^T B x \tag{4.1}$$

where B is the modified connectivity matrix . The rank of this modified connectivity matrix B is $n-1$, and the corresponding eigenvalues are $0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{n-1}$. Now the goal is to minimize $x^T B x$, subject to B being positive semidefinite. Since B is symmetric, it can be expanded into the outer product of the eigenvectors. More explicitly,

$$B = \sum_{i=0}^{n-1} \lambda_i v_i v_i^T = \sum_{i=1}^{n-1} \lambda_i v_i v_i^T \tag{4.2}$$

where v_i is the i-th eigenvector corresponding to λ_i. Then

$$x^T B x = \sum_{i=1}^{n-1} \lambda_i (x^T v_i)(v_i{}^T x) = \sum_{i=1}^{n-1} \lambda_i \mid x^T v_i \mid^2 \tag{4.3}$$

Since $x = v_0 = [1, 1, \cdots, 1]^T$ is a meaningless solution, we try to align x with v_1, the eigenvector corresponding to the smallest nontrivial eigenvalue.

In other words, we are trying to maximize $x^T v_1$. We show in the following theorem that the optimal linear placement is to assign the positions sequentially corresponding to the order of the element of the searching vector.

THEOREM (4.1): Let $v = (\nu_1 , \nu_2 , \cdots , \nu_n)^T$ where $\nu_1 \leq \nu_2 \leq \cdots \leq \nu_n$, and $p = (m_1 , m_2 , \cdots , m_n)^T$ where $m_1 \leq m_2 \leq \cdots \leq m_n$, and let the position vector be $x = (m_{\pi(1)} , m_{\pi(2)} , \cdots , m_{\pi(n)})$, which is a permutation of the ordered position, then

$$p^T v \geq x^T v \quad , \text{ for any } x \quad .$$

Proof:

$$\sum_{i=1}^{n} m_i = \sum_{i=1}^{n} m_{\pi(i)} \tag{4.4}$$

Then

$$T = p^T v - x^T v = \sum_{i=1}^{n} (m_i - m_{\pi(i)})\nu_i \tag{4.5}$$

From equation (4.4), we have

$$m_1 - m_{\pi(1)} = \sum_{i=2}^{n} (m_{\pi(i)} - m_i) \tag{4.6}$$

substituting this into equation (4.5),

$$T = \sum_{i=2}^{n} (m_{\pi(i)} - m_i) (\nu_1 - \nu_i) \tag{4.7}$$

Since m_1 is the smallest, it is obvious that $m_1 \leq m_{\pi(1)}$. Thus from equation (4.6) we have

$$\sum_{i=2}^{n} m_{\pi(i)} \leq \sum_{i=2}^{n} m_i \tag{4.8}$$

so

$$m_{\pi(2)} - m_2 \leq \sum_{i=3}^{n} (m_i - m_{\pi(i)}) \tag{4.9}$$

Hence

$$T \geq \sum_{i=3}^{n} (m_{\pi(i)} - m_i) (\nu_2 - \nu_i) \tag{4.10}$$

Similarly,

$$\sum_{i=3}^{n} m_{\pi(i)} \leq \sum_{i=3}^{n} m_i \tag{4.11}$$

and

$$m_{\pi(3)} - m_3 \leq \sum_{i=4}^{n} (m_i - m_{\pi(i)}) \tag{4.12}$$

also

$$T \geq \sum_{i=4}^{n} (m_{\pi(i)} - m_i) (\nu_3 - v_i) \tag{4.13}$$

Proceeding in this manner, we obtain

$$T \geq (m_{\pi(n)} - m_n) (\nu_{n-1} - \nu_n) \tag{4.14}$$

Since $m_{\pi(n)} \leq m_n$, $\nu_{n-1} \leq \nu_n$, we conclude that $T \geq 0$ or $p^T\ v \geq x^T\ v$. #

4.2 Searching Through The Feasible Region

Let us represent the position vector x in terms of the eigenvectors of B. Then

$$x = \sum_{i=0}^{n-1} \alpha_i v_i \tag{4.15}$$

Thus the objective function can be written as

$$L = x^T\ B\ x = \sum_{i=1}^{n-1} \lambda_i \alpha_i^2 \tag{4.16}$$

If we draw the equipotential contour for different values of L, we will have

$$\sum_{i=1}^{n-1} \frac{\alpha_i^2}{(L/\lambda_i)} = 1 \tag{4.17}$$

This is the equation of an ellipsoid [5].

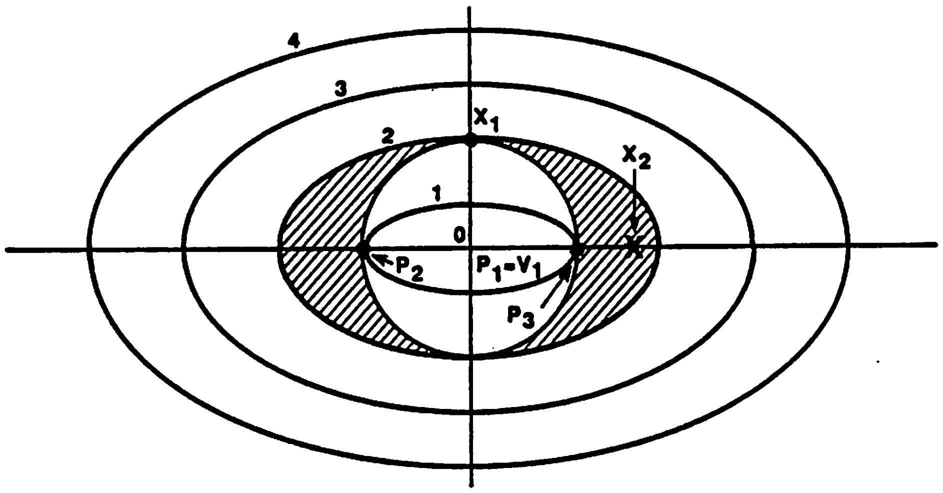

Fig.(4.1)

One very interesting point is that the closest point we found by applying the theorem (4.1) is not necessarily the optimum solution. This is obvious from the example shown in Fig. (4.1). The position vector x_1 is closer to v_1 but is not better than x_2.

However, there is some information that we can collect from the initial solution x_1. We know that there is no other feasible points closer to v_1 than x_1. Also, if there are better solutions they must lie inside the equipotential contour where x_1 belongs. Hence the feasible region is shown as the shaded area, which is much smaller than the whole space.

Then we can easily find the new unrestricted optimum points p_2 and p_3. Here the word "unrestricted" means that the objects are not required to be legally on slot. Using the same matching method to p_2 and p_3 , we pick two new closest points. For the example in Fig. (4.1), the better solution x_2 is chosen to replace x_1.

At each step, we confine the feasible region in a smaller area and update the newest set of unrestricted optimum points for use in the next step. In practice, only a few steps are needed to obtain fairly good results.

We summarize the algorithm briefly as the following:

step 1: Perform eigenvector decomposition of the matrix B.

step 2: Let the constraint set S include the first order and second order constraints. Initialize the minimum $MIN = \infty$.

step 3: Based on the constraints in S, find the unrestricted optimum point p_k.

step 4: Find the closest point x_k to p_k.

step 5: If $MIN > cost(\ x_k\ )$ then $MIN = cost(\ x_k\ )$. Stop when the user defined condition is satisfied.

step 6: Add the constraint $\{ < x\ , p_k >_V \leq < x_k\ , p_k >_V$, for any $x\}$ to S.

step 7: $k = k + 1$; Go to step 3.

During the first few iterations, the unrestricted optimum points can be derived explicitly. For instance, the unstricted optimum point, p_1 , at the first iteration is v_1. Suppose the normalized projection of the closest point to p_1 is equal to a_1, then the unrestricted optimum points for the next two iterations are $p_2 = a_1 v_1 + (\sqrt{1 - a_1^2}\,)v_2$ and $p_3 = a_1 v_1 - (\sqrt{1 - a_1^2}\,)v_2$. In our implementation , we continue the procedure until the eigenvector v_4 is involved. The reason we stop at this step is due to the large size of the constraint set S making it difficult to derive an explicit formula. The user may wish to try different conditions for terminating.

The complexity of a standard eigenvector decomposition for symmetric real matrix is $O(n^3)$, which is the critical part of this algorithm. Sparse matrix eigenvector decomposition technique may be used to speed up the procedure.

5. Experimental Results

A general routine was implemented. We tried some bench mark examples from published papers [1,4,6,7] ,and obtained promising results. We list the results on the following tables. Note that for multiple-pin nets, we use clique model and set the edge weight as $2/n$ for minimum squared length and weight $2^3/n^3$ for minimum Manhattan length. The first three examples deal with linear placement and the last two deal with partitioning.

Example 1: 9 vertices, 16 nets and 43 pins [4,6]

	Tsay-Kuh	*Cheng*	*Network Optimization*	*Kang*
Manhattan length	50	50	50	50
Squared length	151	152	152	152

Example 2: 16 vertices, 17 nets and 42 pins [4,7]

	Tsay-Kuh	*Cheng*	*Network Optimization*	*Wing*
Manhattan length	65	73	79	78
Squared length	375	509	451	628

Example 3: 31 vertices, 33 nets and 81 pins [4,6]

	Tsay-Kuh	*Cheng*	*Network Optimization*	*Kang*
Manhattan length	89	88*	102	95
Squared length	500	556*	464	887

* The results are incomplete because the "IO pads", P1 and OP1, were not taken into account. The actual values are 94 and 578 for Manhattan and squared length, respectively.

Example 4: 20 vertices, 55 nets and 110 pins [1]

	Tsay-Kuh	*Barnes*
cut	13	13

Example 5: 20 vertices, 51 nets and 102 pins [1]

	Tsay-Kuh	*Barnes*
cut	13	13

References

[1] E. R. Barnes, "An Algorithm for Partitioning the Nodes of a Graph," SIAM J. Alg. Disc. Meth., Vol. 3, No. 4, December, 1982, pp. 541-550.

[2] C. C. Chen, "Placement and Partititoning Methods for Integrated Circuits Layout," Ph.D. dissertation, UC Berkeley, EECS Dept., 1985.

[3] K. M. Hall, "An r-Dimensional Quadratic Placement Algorithm," Management Science, Vol. 17, No. 3, November, 1970, pp. 219-229.

[4] C. K. Cheng, "Placement Algorithms and Applications to VLSI Design", Ph.D. dissertation, UC Berkeley, EECS Dept., 1984.

[5] J. P. Blanks, "Near-Optimal Placement Using a Quadratic Objective Function," 22nd Design Automation Conference, 1985, pp. 609-617.

[6] S. Kang, "Linear Ordering and Application to Placement," Proc. 9th Design Automation Workshop, 1972, pp. 50-56.

[7] O. Wing, "Interval-Graph-Based Circuit Layout," Proc. IEEE Int. Conf. on CAD, 1983, pp. 84-85.