A matrix synthesis approach to thermal placement — Source link

Chris Chu, D. F. Wong

Institutions: University of Texas at Austin

Published on: 01 Dec 1998 - IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (IEEE)

Topics: Block matrix, Approximation algorithm and Matrix (mathematics)

A Matrix Synthesis Approach to Thermal Placement

Chris C. N. Chu and D. F. Wong

***Abstract*— In this paper, we consider the thermal placement problem for gate arrays. We introduce a new combinatorial optimization problem, matrix synthesis problem (MSP), to model the thermal placement problem. Given a list of mn nonnegative real numbers and an integer t, MSP constructs a $m \times n$ matrix out of the given numbers such that the maximum sum among all $t \times t$ submatrices is minimized. We show that MSP is NP-complete and present several provably good approximation algorithms for the problem. We also demonstrate that our thermal placement strategy is flexible enough to allow simultaneous consideration of other objectives such as wiring.**

***Index Terms*—Approximation algorithm, thermal placement.**

0	3	7
8	7	0
4	8	2
3	1	3

(a)

8	1	8
3	0	3
3	0	4
7	2	7

(b)

Fig. 1. An example of MSP with $m = 4$, $n = 3$, and $t = 2$. The problem is to synthesize a 4×3 matrix out of 12 numbers (8, 8, 7, 7, 4, 3, 3, 3, 3, 2, 1, 0, 0) and to minimize the maximum sum over all 2×2 submatrices. (a) is a bad solution (maximum sum is 27), and (b) is an optimal solution (maximum sum is 13). The submatrices with maximum sum are shaded.

I. Introduction

High-performance circuits consume a considerable amount of power due to increases of frequency, bandwidth, and system integration. For examples, the two recent high-performance chips, Alpha 21 164 and PowerPC 620, consume 50 and 30 W, respectively, on 3 cm^2 dies. It can be extrapolated that a 10 cm^2 next-generation microprocessor, clocked at 500 MHz, would consume 300 W [8].

Consumed power is converted directly into dissipated heat. In the past decade, heat produced by a chip has increased from 2.2 to 10 W/cm^2 due to the continuous increase of the clock frequency and the total number of transistors [10].

Higher temperature not only affects circuit performance directly by slowing down the transistors on CMOS chips but also decreases their reliability. A circuit with considerable power consumption requires extra expensive cost to remove heat at the packaging level, and therefore the reduction of power dissipation is required at the chip design stages. (See [8] for a survey of current research efforts in power minimization in IC design.)

Even when the total power consumption of a chip is constrained, an unevenly distributed heat dissipation by the gates in the chip may produce hot spots, which can lead to reliability problems. It is also desirable to have an even temperature distribution for the temperature-sensitive circuit (whose characteristic, such as the gain factor β of a CMOS or bipolar circuit, affects its output). Therefore, during physical design of a very large scale integration (VLSI) chip, it is important to place the gates such that heat dissipation by the gates are evenly distributed.

The thermal placement problem has been studied in the past for placing chips during the packaging stage (for printed circuit boards and multichip modules) [2], [6], [7]. However, since thermal placement of gates within a single chip was not of major concern in the past, existing placement algorithms [9] only focus on minimizing area and delay but do not consider heat dissipation. One exception is [1], but it only addresses thermal issues during IC floor planning. In this paper, we consider the thermal placement problem for gate arrays. We introduce a new combinatorial optimization problem, matrix synthesis problem (MSP), to model the thermal placement problem.

Basically, MSP is to synthesize a matrix out of a given list of numbers such that no submatrix of a particular size has a large sum. In this paper, submatrix means those consisting of consecutive rows and columns. For any matrix M, let $S_t(M)$ be the set of all $t \times t$ submatrices of M. Let $\sigma(M)$ be the sum of all entries in M. Let $\mu_t(M) = \max_{S \in S_t(M)} \sigma(S)$. MSP can be defined formally as follows:

Matrix Synthesis Problem (MSP)

- INSTANCE: Integers t, m, n, and a list of mn nonnegative real numbers $x_0, x_1, \cdots, x_{mn-1}$.
- QUESTION: Synthesize a $m \times n$ matrix M out of $x_0, \cdots, x_{mn-1}$ such that $\mu_t(M)$ is minimized.

See Fig. 1 for an example.

It is not difficult to see that MSP models the thermal placement problem for gate arrays. We represent the amount of heat generated by each gate by a nonnegative real number. (If we have less gates than the number of array slots, we can add some zeros.) A submatrix in $S_t(M)$ corresponds to a region of size $t \times t$ on the chip. The submatrix with the largest sum corresponds to the hottest region on the chip. So MSP is equivalent to finding a placement of the gates such that the temperature of the hottest region is the lowest among all possible placements.

The parameter t is to model how good the heat transfer is. If the heat transfer is poor such that the effect of a gate is mostly on neighbor gates, then MSP with $t = 2$ probably is

Manuscript received April 24, 1998. This work was supported in part by the Texas Advanced Research Program under Grant 003658288. This paper was recommended by Associate Editor C. K. Cheng.

The authors are with the Department of Computer Sciences, University of Texas at Austin, Austin, TX 78712 USA.

Publisher Item Identifier S 0278-0070(98)08594-7.

a good model to use. On the other hand, if the heat transfer is good, we may want to consider larger regions and hence a larger t.

Our formulation assumes that the user will provide the data corresponding to the power dissipation of each gate. So the accuracy of our algorithms depends on the accuracy of the data provided. In general, the power dissipation of a gate is affected by several factors like circuit structure, functionality of the gate, wire loading, and input data. The power-estimation problem is a hard problem by itself. It is an active research area recently, and many different probabilistic and statistical techniques have been proposed. See [4] and [5] for survey articles.

A summary of the remainder of this paper is given below. In Section II, we show that MSP with any fixed $t \geq 2$ is NP-complete. (MSP with $t = 1$ is trivially in P.) In Section III, we give a simple algorithm (called $A1$) that approximates MSP to within a factor of two for every $t \geq 2$. In Section IV, we give a modified version of $A1$ (called $A2$). For $t = 2$, $A2$ approximates MSP to within a factor of 5/3. If a simple condition on the input is satisfied, $A2$ approximates MSP to within a factor of 1.5 for every $t \geq 2$. $A1$ and $A2$ output a placement that is good for a particular t only. In Section V, we give a recursive algorithm (called $A3$) that outputs a single placement such that besides approximating MSP with parameter t, it also approximates MSP with parameter t' to within a factor of at most five for all $t' < t$.

In Section VI, some experimental results are given. First, note that the approximation factors shown in Sections III–V are worst case bounds only, and we show that the algorithms work much better in practice. Second, we consider thermal placement and optimization of other objectives at the same time. That is because when we place gates into a chip, we may have other concerns besides heat consideration. We show that the placements by $A1$ and $A3$ are so flexible that the flexibility can be used in optimizing other objectives simultaneously. We demonstrate the idea by considering thermal distribution and wiring at the same time. In Section VII, we conclude by discussing some directions for future work.

II. NP-Completeness

MSP with $t = 1$ is very easy since every placement is optimal. However, we will show that MSP with every fixed $t \geq 2$ is NP-complete. To prove this result, we need the following definitions.

Decision Version of MSP

- INSTANCE: A positive real bound B, integers $t, m, n,$ and a list of mn nonnegative real numbers $x_0, x_1, \cdots, x_{mn-1}$.
- QUESTION: Is it possible to synthesize a $m \times n$ matrix M out of $x_0, x_1, \cdots, x_{mn-1}$ such that $\mu_t(M) \leq B$?

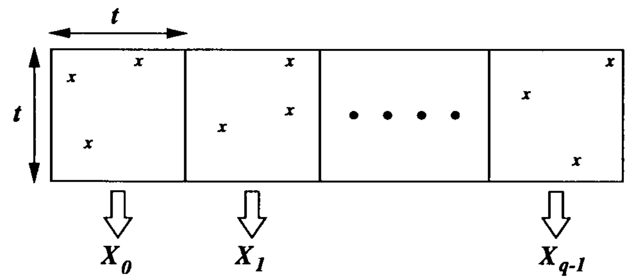


Fig. 2. Construction of a partition according to a placement.

3-Partition

- INSTANCE: A positive real bound B, and a multi-set X of $3q$ positive real numbers such that $\sum_{x \in X} x = qB$ and $\forall\, x \in X$, $B/4 < x < B/2$.
- QUESTION: Can X be partitioned into q multi-sets $X_0, \cdots, X_{q-1}$ such that for $0 \leq r \leq q-1$, $\sum_{x \in X_r} x = B$?

Note that 3-Partition is NP-complete [3].

In this paper, we assume that the indexes of matrices start at zero. Let $S_t^{ij}(M)$ be the $t \times t$ submatrix in $S_t(M)$ at the intersection of rows $i, \cdots, i+t-1$ and columns $j, \cdots, j+t-1$. Let $\hat{S}_t(M)$ be the set of all $t \times t$ submatrices $S_t^{ij}(M)$ such that $i \equiv j \equiv 0(\text{mod t})$.

Theorem 1: For every fixed $t \geq 2$, MSP is NP-complete.

Proof: Let t be any fixed integer greater than or equal to two. Given an instance of 3-Partition, we can reduce it to an instance of MSP with that particular value of t. The bound B for the MSP is the same as the B for the 3-Partition problem. We set $m = t$ and $n = tq$. The mn nonnegative real numbers are those in X together with $mn - 3q$ zeros. We have to show that the instance of 3-Partition returns "YES" if and only if the instance of MSP returns "YES."

$\Leftarrow$*:* If the instance of MSP returns "YES," then we have a $t \times tq$ matrix M such that for all $S \in S_t(M)$, $\sigma(S) \leq B$. In particular, for all $S \in \hat{S}_t(M)$, $\sigma(S) \leq B$. As the q submatrices in $\hat{S}_t(M)$ cover the whole matrix M, the sum of all the numbers in them equals qB. So $\sigma(S) = B$ for all $S \in \hat{S}_t(M)$. If we ignore all the zeros not from X, then the q submatrices in $\hat{S}_t(M)$ define a partition of X such that the sum of each partition is B (see Fig. 2). So the instance of 3-Partition will return "YES."

$\Rightarrow$*:* If the instance of 3-Partition returns "YES," let $X_0, \cdots, X_{q-1}$ be the partition. Since $B/4 < x < B/2$ for all $x \in X$, each partition should contain exactly three numbers.

Case 1) $t \geq 3$: For each r, we can put the three numbers of X_r into the first column of a distinct $S \in \hat{S}_t(M)$ and zeros into other positions (see Fig. 3) Then every submatrix in $S_t(M)$ should contain three numbers from some X_r and some zeros but nothing else. Hence every submatrix sum is B. So the instance of MSP will return "YES."

Case 2) $t = 2$: In this case, we do not have three positions in the first column of each submatrix in $\hat{S}_2(M)$. So we place two numbers in the first column and one in the second column. Let $X_r = \{a_r, b_r, c_r\}$. Without loss of generality, assume

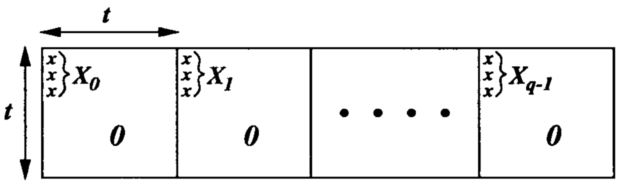


Fig. 3. Construction of a placement according to a partition when $t \geq 3$.

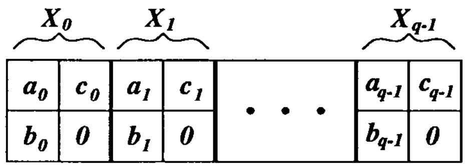


Fig. 4. Construction of a placement according to a partition when $t = 2$.

that $c_0 \leq c_1 \leq \cdots \leq c_{q-1}$. Then the placement is as shown in Fig. 4. It is obvious that for all $S \in \hat{S}_2(M)$, $\sigma(S) = B$. Consider $S_2^{0,\,2r+1}(M)$ for some $0 \leq r \leq q-1$, which is not in $\hat{S}_2(M)$. $\sigma(S_2^{0,\,2r+1}(M)) = c_r + a_{r+1} + b_{r+1} \leq c_{r+1} + a_{r+1} + b_{r+1} = \sigma(S_2^{0,\,2r+2}(M)) = B$ as $S_2^{0,\,2r+2}(M) \in \hat{S}_2(M)$. So the instance of MSP will return "YES." ■

III. A Simple Approximation Algorithm

From now on, we assume for simplicity that $m = n = tq$ for some integer q. In other words, we are placing t^2q^2 numbers into a $tq \times tq$ matrix. Note that in this case, $\hat{S}_t(M)$ is a set of q^2 nonoverlapping submatrices that covers the whole matrix M. We can obtain similar results if $m \neq n$, or m or n is not a multiple of t. Without loss of generality, we also assume that $x_0 \geq x_1 \geq \cdots \geq x_{n^2-1}$.

The algorithm $A1$ below approximates MSP to within a factor of two. The basic idea of the algorithm is to distribute the numbers evenly in the matrix. We divide the numbers into $t \times t$ groups according to their magnitudes. We observe that it is possible to have a placement with the property that every $t \times t$ submatrix contains exactly one number from each group.

ALGORITHM $A1$

1. For $0 \leq k \leq t^2 - 1$, let group G_k contain the numbers $x_{kq^2}, \cdots, x_{kq^2+q^2-1}$.
2. For $0 \leq k \leq t^2 - 1$, for all $i \equiv \lfloor k/t \rfloor \bmod t$ and for all $j \equiv (k \bmod t) \bmod t$, label $m_{i,\,j}$ (entry $(i,\,j)$ of matrix M) as L_k.
3. For $0 \leq k \leq t^2 - 1$, place each number in group G_k arbitrarily into a distinct position of M labeled with L_k.

For example, let $t = 2$, $m = n = 6$, $x_i = 35 - i$ for $0 \leq i \leq 35$. In other words, we are placing the numbers 35, 34, $\cdots$, 0 into a 6 × 6 matrix. Then G_0 contains 35, $\cdots$, 27, G_1 contains 26, $\cdots$, 18, G_2 contains 17, $\cdots$, 9, and G_3 contains 8, $\cdots$, 0. The labeling is as shown in Fig. 5. A possible placement is in Fig. 6. Note that those numbers from group G_0 are evenly distributed in the matrix. This is also true for all other groups.

L_0	L_1	L_0	L_1	L_0	L_1
L_2	L_3	L_2	L_3	L_2	L_3
L_0	L_1	L_0	L_1	L_0	L_1
L_2	L_3	L_2	L_3	L_2	L_3
L_0	L_1	L_0	L_1	L_0	L_1
L_2	L_3	L_2	L_3	L_2	L_3

Fig. 5. Labeling of algorithm $A1$ with $t = 2$ and $n = 6$. Note that there is exactly one of each of L_0, L_1, L_2, and L_3 inside every 2 × 2 submatrix.

30	19	33	25	27	24
17	1	16	5	15	4
35	21	29	18	32	26
14	0	13	2	12	8
31	20	28	22	34	23
11	7	10	3	9	6

Fig. 6. A possible placement by algorithm $A1$ for the numbers $35, 34, \cdots, 0$. The entries with label L_0 (i.e., numbers from group G_0) are shaded.

Let OPT_t be the optimal placement for MSP with parameter t. Before proving the approximation factor for $A1$, we first give two lower bounds on $\mu_t(OPT_t)$.

Lemma 1: For every $t \geq 2$, $0 \leq k \leq t^2 - 1$, $\mu_t(OPT_t) \geq (k+1)x_{kq^2}$.

Proof: $x_0, \cdots, x_{kq^2}$ are $kq^2 + 1$ numbers at least as large as x_{kq^2}. Consider the q^2 submatrices in $\hat{S}_t(OPT_t)$. By pigeonhole principle, there must be a submatrix containing at least $k+1$ numbers larger than or equal to x_{kq^2}. So $\mu_t(OPT_t) \geq (k+1)x_{kq^2}$. ■

Lemma 2: For every $t \geq 2$, $\mu_t(OPT_t) \geq 1/q^2 \sum_{i=0}^{n^2-1} x_i$.

Proof:

$$\mu_t(OPT_t) \geq \frac{1}{q^2} \sum_{S \in \hat{S}_t(OPT_t)} \sigma(S) \tag{1}$$

$$= \frac{1}{q^2} \sum_{i=0}^{n^2-1} x_i. \tag{2}$$

Equation (1) follows from the fact that $\mu_t(OPT_t) \geq \sigma(S)$ for any $S \in \hat{S}_t(OPT_t)$ and $|\hat{S}_t(OPT_t)| = q^2$. Equation (2) follows from the fact that $\hat{S}_t(M)$ is a set of nonoverlapping submatrices that covers the whole matrix M. ■

Theorem 2: For every $t \geq 2$, $\mu_t(A1) \leq 2 \cdot \mu_t(OPT_t)$.

Proof:

$$\mu_t(A1) \le x_0 + x_{q^2} + \cdots + x_{(t^2-1)q^2} \tag{3}$$

$$\le x_0 + \frac{1}{q^2}\sum_{i=0}^{q^2-1} x_i + \frac{1}{q^2}\sum_{i=q^2}^{2q^2-1} x_i$$

$$+ \cdots + \frac{1}{q^2}\sum_{i=(t^2-2)q^2}^{(t^2-1)q^2-1} x_i \tag{4}$$

$$\le x_0 + \frac{1}{q^2}\sum_{i=0}^{n^2-1} x_i$$

$$\le 2 \cdot \mu_t(OPT_t). \tag{5}$$

By the way we place the numbers, each $t \times t$ submatrix contains exactly one number from each group G_k. Note that $x \le x_{kq^2}$ for every number x in G_k. So for any $S \in S_t(A1)$, $\sigma(S) \le x_0 + x_{q^2} + \cdots + x_{(t^2-1)q^2}$. Equation (3) immediately follows. Equation (4) follows from the fact that $x_{kq^2} \le x_{kq^2-r}$ for $1 \le r \le q^2$ as the numbers are sorted in decreasing order. Equation (5) follows from Lemma 1 with $k = 0$ and Lemma 2. ■

IV. A Better Approximation Algorithm

In Step 3 of algorithm $A1$, the placement of numbers from group G_k into entries marked with label L_k is done arbitrarily. The algorithm $A2$ given below makes use of this flexibility on placement to improve the approximation factor.

ALGORITHM $A2$

1. For $0 \le k \le t^2 - 1$, let group G_k contains the numbers $x_{kq^2}, \cdots, x_{kq^2+q^2-1}$.
2. For $0 \le k \le t^2 - 1$, for all $i \equiv \lfloor k/t \rfloor \bmod t$ and for all $j \equiv (k \bmod t) \bmod t$, label m_{ij} (entry (i, j) of matrix) as L_k.
3. Place each number of group G_0 into a distinct position of M labeled with L_0 (i.e. into m_{ij} s.t. and j are multiple of t) such that $m_{ut,\,vt} \ge m_{ut+t,\,vt}$ and $m_{ut,\,vt} \ge m_{ut,\,vt+t}$ for all u, v.
4. For $0 \le r \le q^2 - 1$, let $S_r \in \hat{S}_t(M)$ be the submatrix where x_r is placed at step 3. For $1 \le k \le t^2 - 1$, place $x_{kq^2+q^2-1-r} \in G_k$ into the entry with label L_k in S_r.

One way to do Step 3 is to place x_r into $m_{ut,\,vt}$, where $u = \lfloor r/q \rfloor$, $v = (r \bmod q)$. Fig. 7 illustrates this step.

The algorithm matches larger numbers from group G_0 with smaller numbers from other groups. So it prevents all the largest numbers of the groups from being placed into the same $t \times t$ submatrix. Intuitively, one might think that it would be better to match larger numbers from half of the groups with

x_0	x_{17}	x_1	x_{16}	x_2	x_{15}
x_{26}	x_{35}	x_{25}	x_{34}	x_{24}	x_{33}
x_3	x_{14}	x_4	x_{13}	x_5	x_{12}
x_{23}	x_{32}	x_{22}	x_{31}	x_{21}	x_{30}
x_6	x_{11}	x_7	x_{10}	x_8	x_9
x_{20}	x_{29}	x_{19}	x_{28}	x_{18}	x_{27}

Fig. 7. A possible implementation for Step 3 of algorithm $A2$ with $n = 6$ and $t = 2$. The entries with label L_0 are shaded.

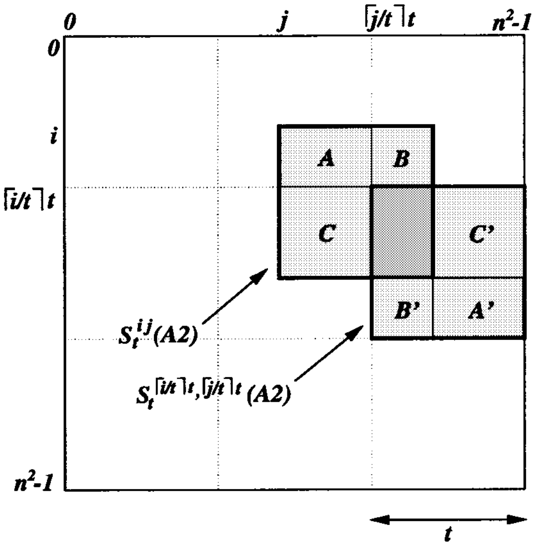


Fig. 8. Illustration to show that $\sigma(S_t^{ij}(A2)) \le \sigma(S_t^{\lceil i/t\rceil t,\, \lceil j/t\rceil t}(A2))$ for all i, j.

smaller numbers from the other half of the groups. However, the worst case bound is better for our algorithm.

Theorem 3: For every $t \ge 2$, if $x_{q^2-1} = \alpha x_0$, then $\mu_t(A2) \le \max(1.5, 2-\alpha) \cdot \mu_t(OPT_t)$.

Proof: As in algorithm $A1$, every $t \times t$ submatrix contains exactly one number from each group. However, because of the way we do the placement in Step 3, $m_{\lceil i/t\rceil t,\, \lceil j/t\rceil t} \le m_{\lceil i/t\rceil t-t,\, \lceil j/t\rceil t}$ for any i, j. Hence by Step 4, this implies $\sigma(B) \le \sigma(B')$, where B and B' are submatrices as shown in Fig. 8. Similarly, since $m_{\lceil i/t\rceil t,\, \lceil j/t\rceil t} \le m_{\lceil i/t\rceil t,\, \lceil j/t\rceil t-t}$ and $m_{\lceil i/t\rceil t,\, \lceil j/t\rceil t} \le m_{\lceil i/t\rceil t-t,\, \lceil j/t\rceil t-t}$, we can prove that $\sigma(C) \le \sigma(C')$ and $\sigma(A) \le \sigma(A')$. So $\sigma(S_t^{ij}(A2)) \le \sigma(S_t^{\lceil i/t\rceil t,\, \lceil j/t\rceil t}(A2))$ (see Fig. 8). Therefore, the sum of every submatrix is dominated by the sum of some submatrix in $\hat{S}_t(A2)$. Hence we can focus on those submatrices in $\hat{S}_t(A2)$.

For any $S_r \in \hat{S}_t(A2)$ as defined in Step 4 of the algorithm

$$\sigma(S_r) = x_r + x_{2q^2-1-r} + x_{3q^2-1-r} + \cdots + x_{t^2q^2-1-r}$$

$$= \left(\frac{r}{q^2}x_r + \frac{q^2-r}{q^2}x_{q^2-1} + \frac{q^2-r}{q^2}x_{2q^2-1-r}\right.$$

$$
\begin{aligned}
&+\frac{q^2}{q^2}\,x_{3q^2-1-r}+\cdots+\frac{q^2}{q^2}\,x_{t^2q^2-1-r}\Bigg)\\
&+\left(\frac{q^2-r}{q^2}\,x_r-\frac{q^2-r}{q^2}\,x_{q^2-1}+\frac{r}{q^2}\,x_{2q^2-1-r}\right)\\
&\le\frac{1}{q^2}\left(\sum_{i=0}^{r-1}x_i+\sum_{i=r}^{q^2-1}x_i+\sum_{i=q^2}^{2q^2-1-r}x_i\right.\\
&\qquad\left.+\sum_{i=2q^2-r}^{3q^2-1-r}x_i+\cdots+\sum_{i=(t^2-1)q^2-r}^{t^2q^2-1-r}x_i\right)\\
&+\left(\frac{q^2-r}{q^2}\,x_0-\frac{q^2-r}{q^2}\,\alpha x_0+\frac{r}{q^2}\,x_{q^2}\right) \qquad (6)\\
&\le\frac{1}{q^2}\left(\sum_{i=0}^{n^2-1}x_i\right)\\
&+\left(\left(1-\frac{r}{q^2}\right)x_0-\left(1-\frac{r}{q^2}\right)\alpha x_0+\frac{r}{q^2}\,x_{q^2}\right)\\
&\le\left(1+\left(1-\frac{r}{q^2}\right)(1-\alpha)+\frac{r}{q^2}\,\frac{1}{2}\right)\cdot\mu_t(OPT_t). \qquad (7)
\end{aligned}
$$

Equation (6) follows from the fact that the x's are sorted in decreasing order and that $x_{q^2-1}=\alpha x_0$. Equation (7) follows from Lemma 1 with $k=0$ and $k=1$ and Lemma 2.

If $\alpha<1/2$, then

$$
\begin{aligned}
\sigma(S_r)&\le\left(2-\alpha-\frac{r}{q^2}\left(1-\alpha-\frac{1}{2}\right)\right)\cdot\mu_t(OPT_t)\\
&\le(2-\alpha)\cdot\mu_t(OPT_t).
\end{aligned}
$$

If $\alpha\ge 1/2$, then

$$
\begin{aligned}
\sigma(S_r)&\le\left(1+\frac{1}{2}\left(1-\frac{r}{q^2}\right)+\frac{1}{2}\,\frac{r}{q^2}\right)\cdot\mu_t(OPT_t)\\
&=1.5\cdot\mu_t(OPT_t).
\end{aligned}
$$

So $\mu_t(A2)\le\max(1.5,\,2-\alpha)\cdot\mu_t(OPT_t)$. ■

Note that Theorem 3 gives a bound worse than 1.5 only when α is small (less than 0.5). In this case, the input should contain a few large numbers and many small numbers.

For the case $t=2$, we can prove a bound that holds for any input. But we need to use another lower bound of $\mu_t(OPT_t)$.

Lemma 3: For all $t\ge 2$ and for all r such that $0\le r\le n^2-1$, $\mu_t(OPT_t)\ge x_r+x_{n^2-1-r}$.

Proof: $x_0,\cdots,x_r$ are $r+1$ numbers larger than or equal to x_r. Consider the q^2 submatrices in $\hat{S}_t(OPT_t)$. If any two of these numbers are in the same submatrix, then the lemma is obviously true. Consider the case when they are in $r+1$ different submatrices in $\hat{S}_t(OPT_t)$. Since there are at most r numbers less than x_{n^2-1-r}, at least one of these $r+1$ submatrices must contain some number larger than or equal to x_{n^2-1-r}. Hence the result follows. ■

Theorem 4: For $t=2$, $\mu_2(A2)\le 5/3\cdot\mu_2(OPT_2)$.

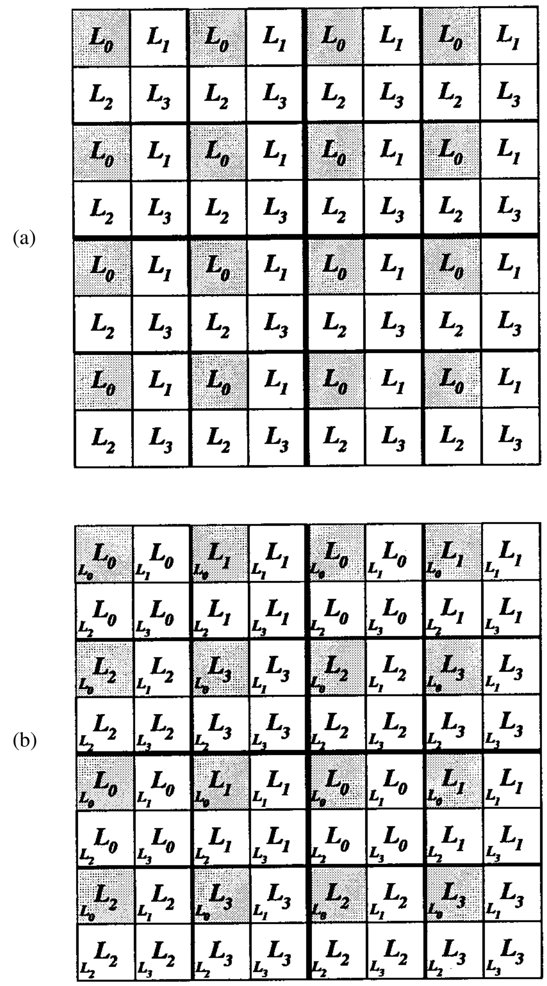


Fig. 9. The labeling of $A3$ with $t=4$ and $n=8$. (a) is the labels for the first level of recursion. Those entries labeled with L_0 at this step are shaded. (b) is the labels for the second level of recursion. The labels for the first level are written at lower left corners.

Proof: As in Theorem 3, we will focus on those submatrix in $\hat{S}_2(A2)$. For any $0\le r\le q^2-1$

$$
\begin{aligned}
\sigma(S_r)&=x_r+x_{2q^2-1-r}+x_{3q^2-1-r}+x_{4q^2-1-r}\\
&=\left(\frac{r}{2q^2}\,x_r+\left(1-\frac{r}{q^2}\right)x_{2q^2-1-r}\right.\\
&\qquad\left.+\tfrac{1}{2}\,x_{3q^2-1-r}+\tfrac{1}{2}\,x_{4q^2-1-r}\right)\\
&\quad+\left(\left(1-\frac{r}{2q^2}\right)x_r+\frac{r}{q^2}\,x_{2q^2-1-r}\right.\\
&\qquad\left.+\tfrac{1}{2}\,x_{3q^2-1-r}+\tfrac{1}{2}\,x_{4q^2-1-r}\right)\\
&\le\frac{1}{2}\,\frac{1}{q^2}\left(\sum_{i=0}^{r-1}x_i+\sum_{i=r}^{2q^2-1-r}x_i\right.\\
&\qquad\left.+\sum_{i=2q^2-r}^{3q^2-1-r}x_i+\sum_{i=3q^2-r}^{4q^2-1-r}x_i\right)\\
&\quad+\left(1-\frac{r}{2q^2}\right)(x_r+x_{4q^2-1-r})+\frac{r}{q^2}\,x_{q^2}+\frac{1}{2}\,x_{2q^2} \qquad (8)
\end{aligned}
$$

TABLE I
AVERAGE APPROXIMATION FACTORS FOR $A1$ AND $A2$

t	avg. approx. factor		
	$A1$	$A2$	Random
2	1.218	1.125	1.899
3	1.079	1.085	1.714
4	1.033	1.054	1.646
5	1.018	1.038	1.480
average	1.087	1.076	1.685

TABLE II
THE WORST CASE BOUNDS $(1-(2^p/n)^2+(2^p/t')^2$, WHERE $p=\lceil \log_2 t' \rceil)$ AND THE AVERAGE VALUES OF THE APPROXIMATION FACTORS OF ALGORITHM $A3$ WITH $t=8$ FOR DIFFERENT t'

t'	worst-case bound for $A3$	avg. approx. factor	
		$A3$ with $t=8$	Random
2	2.000	1.247	1.899
3	2.777	1.370	1.714
4	1.999	1.053	1.646
5	3.556	1.222	1.480
6	2.773	1.084	1.388
7	2.302	1.263	1.454
8	1.996	1.006	1.287
average	2.486	1.178	1.553

$$\leq \frac{1}{2}\frac{1}{q^2}\left(\sum_{i=0}^{n^2-1} x_i\right) + \left(1-\frac{r}{2q^2}\right)(x_r + x_{4q^2-1-r}) + \frac{r}{q^2}x_{q^2} + \frac{1}{2}x_{2q^2}$$

$$\leq \left(\frac{1}{2} + \left(1-\frac{r}{2q^2}\right) + \frac{r}{q^2}\frac{1}{2} + \frac{1}{2}\frac{1}{3}\right)\cdot \mu_2(OPT_2) \tag{9}$$

$$= \tfrac{5}{3}\cdot \mu_2(OPT_2).$$

Equation (8) follows from the fact the x's are sorted in decreasing order and that $1 - r/2q^2 \geq 1/2$. Equation (9) follows from Lemma 1 with $k=1$ and $k=2$, Lemmas 2 and 3.

So for $t=2$, $\mu_2(A2) \leq 5/3 \cdot \mu_2(OPT_2)$. ■

V. A RECURSIVE APPROXIMATION ALGORITHM

For the thermal placement problem, if the heat transfer is good, it is reasonable to consider larger regions and hence to use a larger t. Smaller regions will become less important as heat generated will be dissipated to other parts of the chip easily. Even if a lot of heat is generated in a small region, if its surrounding region does not generate much heat, the heat will spread out quickly to a larger region. However, it does not mean that the heat consideration of smaller regions is totally unimportant. One may still want to have some bounds on the amount of heat generated by smaller regions.

In the previous two sections, we present two algorithms $A1$ and $A2$ that give placements that are good for a particular t. If we consider a parameter $t' < t$, those placements generated with parameter t do not give much guarantee on the approximation factor. For example, if we run $A1$ with $t=4$, the numbers from G_0, G_1, G_4, and G_5 will be placed next to each other. As the numbers from these four groups are relatively large, if we run $A1$ with $t=4$, $\mu_2(A1)$ may be large.

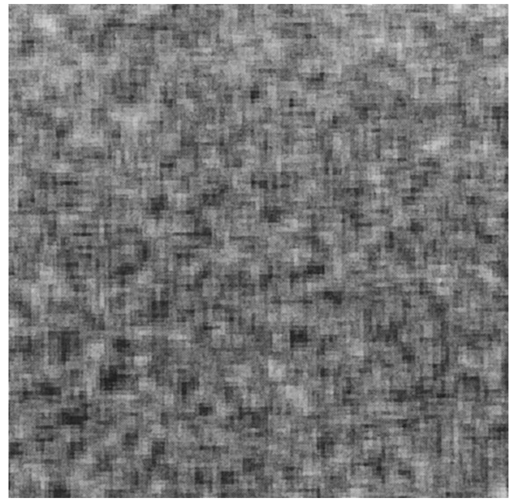


Fig. 10. Heat distribution of a random placement. There are many hot spots (white spots) in this placement.

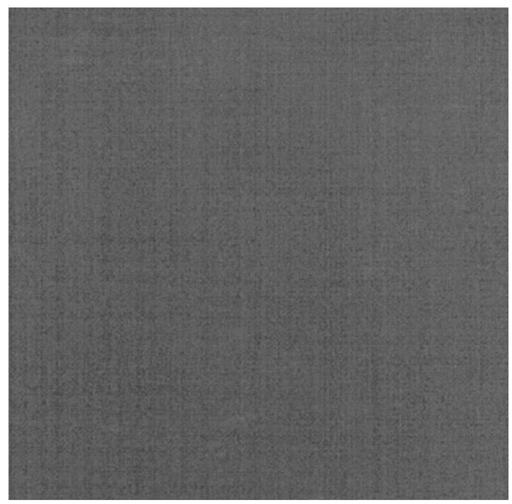


Fig. 11. Heat distribution of a placement by $A1$. There is no hot spot (white spot) in this placement. The heat is evenly distributed.

It can be easily seen that the problem with the previous two algorithms is that there is no intention to distribute the numbers from different groups evenly inside a $t \times t$ submatrix. If we do the labeling carefully, we should be able to obtain better bounds for smaller submatrices. In this section, we give an algorithm $A3$ that outputs a single placement such that besides approximating MSP with parameter t to within a factor two, it also approximates MSP with parameter t' to within a factor of at most five for all $t' < t$, when t is a power of two.

The idea is to do the labeling by $A1$ with $t=2$ recursively. For a $2q \times 2q$ matrix labeled by $A1$ with $t=2$, if we consider the $q \times q$ matrix formed by removing all the entries other than those marked with L_0, and apply $A1$ with $t=2$ again to place the q^2 numbers of G_0 into it, then we know that the largest

TABLE III
COMPARISON OF TRADITIONAL PLACEMENT BASED ON THE WIRING OBJECTIVE ONLY AND OUR APPROACH OF PLACEMENT, WHICH CONSIDERS BOTH HEAT DISTRIBUTION AND WIRING

Circuit			Wiring			Heat		
name	size	n	Traditional	Our alg.	inc%	Traditional	Our alg.	dec%
s5378	2978	55	23912	23912	0.0	1.878	1.224	34.8
s9234	5844	77	58209	58546	0.6	1.882	1.226	34.9
s13207	8727	94	94698	95547	0.9	1.934	1.224	36.7
s15850	10397	102	128369	130003	1.3	1.889	1.216	35.6
s38584	20871	145	375121	375577	0.1	1.949	1.222	37.3
s38417	24061	156	444150	447792	0.8	1.893	1.244	34.3
		average			0.6			35.6

numbers of G_0 will not be placed adjacent to each other in the original matrix. We can continue the idea recursively until the groups we are considering are small enough. Then we can apply the same procedure to G_1, G_2, and G_3. The algorithm is given below.

ALGORITHM $A3$

1. Divide the input numbers into 4 groups G_0, G_1, G_2 and G_3 and label the matrix by L_0, L_1, L_2 and L_3 as in step 1 and 2 of algorithm $A1$ with $t = 2$.
2. Recursively place the numbers in G_0 into the submatrix formed by entries marked with L_0 until the size of each group is n^2/t^2. In that case, we do the placement arbitrarily instead of doing it recursively.
3. Apply the same procedure to G_1, G_2 and G_3.

Note that we assume t is a power of two in algorithm $A3$. If t is not a power of two, we can use the smallest power of two bigger than t as the parameter for $A3$ instead.

An example of the labeling is shown in Fig. 9. Basically, as in $A1$ with $t = 4$, we are dividing the input numbers into 16 groups (four groups in the first level of recursion and then 16 groups in the second level) such that there is exactly one number from each group in every 4×4 submatrix. So the sum of every 4×4 submatrix will not differ by too much. However, because of the way we do the labeling, numbers from different groups are evenly distributed inside every 4×4 submatrix. So we can obtain some bounds for 3×3 and 2×2 submatrices too.

Theorem 5: Suppose t is a power of two. For any t' such that $2 \le t' \le t$, let p be the integer such that $2^{p-1} < t' \le 2^p$. Then $\mu_{t'}(A3) \le (1 - (2^p/n)^2 + (2^p/t')^2) \cdot \mu_{t'}(OPT_{t'})$.

Proof: Let r be the integer such that $t = 2^r$. For any $2 \le t' \le t$ and $S \in S_{t'}(A3)$

$$\begin{aligned}\sigma(S) &\le x_0 + x_{2^{2r-2p}q^2} + x_{2\cdot 2^{2r-2p}q^2} \\ &\quad + \cdots + x_{(2^{2p}-1)2^{2r-2p}q^2} \\ &\le \left(1 - \frac{1}{2^{2r-2p}q^2}\right) x_0 + \frac{1}{2^{2r-2p}q^2} \sum_{i=0}^{n^2-1} x_i \qquad (10) \\ &= \left(1 - \frac{2^{2p}}{n^2}\right) x_0 + \frac{2^{2p}}{n^2} \sum_{i=0}^{n^2-1} x_i \\ &\le \left(1 - \frac{2^{2p}}{n^2}\right) \cdot \mu_{t'}(OPT_{t'}) + \frac{2^{2p}}{t'^2} \cdot \mu_{t'}(OPT_{t'}) \qquad (11) \\ &= \left(1 - \left(\frac{2^p}{n}\right)^2 + \left(\frac{2^p}{t'}\right)^2\right) \cdot \mu_{t'}(OPT_{t'}).\end{aligned}$$

Equation (10) follows from the fact that x's are sorted in decreasing order. Equation (11) follows from Lemma 1 with $k = 0$ and Lemma 2.

Hence the theorem follows. ■

Note that if t' is a power of two, the approximation factor is at most two. Otherwise, a rough upper bound on the approximation factor is $(1 + (2^p/2^{p-1})^2) = 5$.

VI. EXPERIMENTAL RESULTS

The approximation factor bounds for the algorithms shown in the previous three sections are all worst case bounds only. We show here that these algorithms perform much better in practice.

As we do not have any actual thermal information for circuits, we generate thermal information uniformly at random. Ten sets of data of size 120×120 are generated. In Table I, the average approximation factors over the ten data are shown when algorithms $A1$ and $A2$ with various values of t are used to place them into a 120×120 matrix. For algorithm $A1$, the placement of numbers inside a group is done randomly. We also include the results of random placements for comparison. If the placement of gates is independent of the amount of heat generated by the gates, then the resulting placement should be similar to a random placement in terms of heat distribution.

As shown in Table I, the approximation factors of our algorithms are very close to optimal in practice. They also perform much better than random placements. Note that as we do not know the optimal value $\mu_t(OPT_t)$, we only use the maximum of the lower bounds in Lemmas 1–3 as an approximation of it. The approximation factors should be even better if optimal values are used.

In Table II, the average approximation factors over the same sets of data for algorithm $A3$ are shown. We use $t = 8$ here, and the approximation factors for $t' < 8$ are also shown. The worst case bounds proved in Theorem 5 and the results of random placements are included for comparison.

As shown in Table II, the algorithm gives pretty good approximation factors simultaneously for all t'. It performs much better in practice than the upper bounds suggest. It also performs much better than random placements. Again, we can only use the lower bounds in Lemmas 1–3 to approximate the optimal values.

Figs. 10 and 11 show the heat distribution of a random placement and a placement by $A1$ with $t = 4$, respectively. The brightness at each point is proportional to the total amount of heat generated by a surrounding region of size 4×4. As we can see, there are many hot spots in the random placement. On the contrary, the heat is very evenly distributed in the placement by $A1$.

When we place gates into a chip, we usually have to optimize other objectives at the same time. For algorithms $A1$ and $A3$, there is large flexibility to do the placement because the algorithms only require a number to be placed in any of those entries with a particular label. Moreover, the entries with that particular label are plenty and are evenly distributed on the matrix.

We observe that such flexibility can be used to simultaneously optimize other objectives. We demonstrate the idea by considering heat distribution and wiring at the same time. A set of MCNC benchmark circuits was used. Since thermal data of these circuits were not available, we generated a number uniformly at random for each gate representing the amount of heat dissipated by the gate. We first obtain a thermally good placement by our thermal placement algorithm $A1$ with $t = 2$. Then we try to improve the total wiring length by simulated annealing. However, we only allow the exchange of two entries such that the differences in row indexes and in column indexes are both multiples of t. So as far as heat is concerned, the placement after the simulated annealing is as good as the one before. As for comparison, we also consider traditional placement based on the wiring objective only. That is, in our experiment, we apply simulated annealing to a random initial placement, using total wire length as the objective, and without imposing any restrictions on the gate locations as was done in the other case. It corresponds to the case when heat is not taken into consideration. Table III gives the results of the experiment.

As expected, our algorithm is not as good as usual simulated annealing in terms of total wire length. However, the increase is very insignificant. On the other hand, our algorithm performs much better in distributing the heat.

VII. Concluding Remarks

We have introduced a new combinatorial problem, MSP, to model the thermal placement problem. We show that MSP is NP-complete and we give three provably good approximation algorithms for it. All three algorithms run in just $O(mn \log mn)$ time for a problem of mn data. Besides, the algorithms are flexible and are good both theoretically and practically in providing an approximate solution.

A direction of future work is to design algorithms with provably better approximation factors for MSP. As we pointed out in Section V, one may want to have bounds on several values of t simultaneously. The worst case bounds given by $A3$ sometimes can be as large as five. It is good to have algorithms with better worst case bounds. We can also generalize MSP by considering a weighted average of the approximation factors for different values of t. This model gives more guarantee than MSP, and it may be easier to work with than the model of providing several bounds simultaneously. However, we have no idea how the weights should look. It is worthwhile to investigate what the weights should be and to design approximation algorithms according to the weight distribution. Another direction is to obtain a simple model that gives the temperature for each point on the chip. In fact, the temperature distribution for a given placement can be found by numerically solving differential equations, but such calculations are too expensive to be used by a placement algorithm.

Acknowledgment

The authors thank Dr. K. Y. Chao of Intel Corp. for his helpful comments.

References

[1] K. Y. Chao and D. F. Wong, "Low power considerations in floorplan design," in *Proc. 1994 Int. Workshop Low Power Design,* 1994, pp. 45–50.

[2] ——, "Thermal placement for high performance multi-chip modules," in *Proc. IEEE Int. Conf. Computer Design (ICCD),* Oct. 1995.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* New York: Freeman, 1979.

[4] E. Macii, M. Pedram, and F. Somenzi, "High level power modeling, estimation and optimization," in *Proc. 34th Design Automation Conf.,* 1997, pp. 504–510.

[5] F. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Trans. VLSI Syst.,* vol. 2, no. 4, pp. 446–455, 1994.

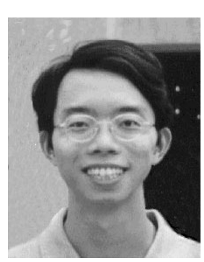
[6] M. D. Osterman and M. Pecht, "Component placement for reliability on conductively cooled printed wiring boards," *ASME J. Packaging,* vol. 111, no. 3, pp. 149–156, 1989.

[7] ——, "Placement for reliability and routability of convectively cooled PWB's," *IEEE Trans. Computer-Aided Design,* vol. 9, no. 7, pp. 734–744, 1990.

[8] M. Pedram, "Power minimization in IC design: Principles and applications," *ACM Trans. Design Automat. Electron. Syst.,* vol. 1, no. 1, pp. 3–56, 1996.

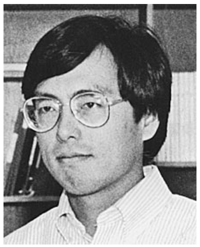
[9] B. T. Preas and M. J. Lorenzetti, *Physical Design Automation of VLSI Systems.* Menlo Park, CA: Benjamin Cummings, 1988.

[10] R. E. Simons, "Microelectronics cooling and semitherm: A look back," in *Proc. 10th Semiconductor Thermal and Temperature Measurement Symp.,* 1994, pp. 1–16.



Chris C. N. Chu received the B.S. degree in computer science from the University of Hong Kong, Hong Kong, in 1993 and the M.S. degree in computer science from the University of Texas at Austin in 1994. He is now pursuing the Ph.D. degree in the Computer Sciences Department of the University of Texas at Austin.

His research interests include design and analysis of algorithms, CAD of VLSI physical design, and performance-driven interconnect optimization.



D. F. Wong received the B.Sc. degree in mathematics from the University of Toronto, Canada, and the M.S. degree in mathematics from the University of Illinois at Urbana-Champaign. He received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign in 1987.

He is currently an Associate Professor of computer sciences at the University of Texas at Austin. His main research interest is CAD of VLSI. He has published more than 160 technical papers and has graduated 15 Ph.D. students in this area. He is a coauthor of *Simulated Annealing for VLSI Design* (Norwell, MA: Kluwer Academic, 1988).

Dr. Wong received Best Paper Awards at DAC-86 and ICCD-95 for his work on floorplan design and FPGA routing, respectively. He is the General Chair of the 1999 ACM International Symposium on Physical Design (ISPD-99) and was the Technical Program Chair of the same conference in 1998 (ISPD-98). He has also served on the technical program committees of many other VLSI CAD conferences (e.g., ICCAD, ED&TC, DATE, ISCAS, and FPGA). He is an Editor of IEEE TRANSACTIONS ON COMPUTERS.