ZOR - Methods and Models of Operations Research (1989) 33:131–143

Optimal Module Sizing in VLSI Floorplanning by Nonlinear Programming

By E. Rosenberg[1]

Abstract: Floorplanning is the VLSI design problem of deciding the position and shapes of all modules (e.g., memory or random logic) in order to minimize the chip area and satisfy all constraints. The modules may not be fixed in shape – the ratio of height to width may be modified in order to achieve minimal chip area. In the CADRE (AT&T) and Planar Package Planner (IBM) floorplanning systems, relative positions of the modules are specified by a "relative position graph". Given the relative positions, a heuristic attempts to determine the optimal shape of each module, subject to the relative orientations dictated by the graph and also interconnection requirements, to minimize the chip area. In CADRE, the heuristics iterate between improving horizontal and vertical module dimensions; the IBM system uses a Simplex-method based heuristic. These heuristics are not guaranteed to solve the problem exactly. Consequently, it is not known how far the heuristic solution is from optimal, and it is not obvious when to terminate the heuristic. In this paper we show that, given the relative positions, we can compute the *provably optimal* shape of each module, by transforming the problem into a convex nonlinear programming problem. Each local minimum of such a problem must also be a global minimum. We illustrate the method by applying it to an example solved by IBM.

1 Introduction

VLSI design is the complex process of translating a high-level functional specification into detailed geometric masks for fabrication. This process is decomposed into two main steps. First, the functional description is mapped to a structural representation which specifies the logical units and interconnections required to perform the function. Then, the structural representation is mapped into a physical layout.

One of the most important and difficult parts of this second step (mapping the structural representation into physical layout) is the floorplanning problem. Floorplanning is the problem of deciding the shapes and locations of all modules in order to minimize the chip area and satisfy the necessary constraints. Typically, a custom VLSI

[1] E. Rosenberg, AT&T Bell Laboratories, Room LZ 3M-214, Lincroft, NJ, 07738, USA.

chip will have up to tens of modules. A module might be a section of memory, a programmable logic array, or a collection of random logic. The modules are typically not fixed in shape – their aspect ratio (ratio of height to width) may be modified in order to achieve minimal chip area. Thus a module may be squeezed into a long narrow rectangle, or kept square, as deemed appropriate by the floorplanner. (It is this feature that chiefly distinguishes the floorplanning problem from the placement problem.) The floorplanner must also determine the positions of the modules on the chip. The constraints on the floorplan include minimum area constraints for any modules and interconnection requirements. There are two main methods for treating interconnection requirements.

One way is to define routing channels between modules, and then determine loose or global routes, within the routing channels, for inter-module interconnections. Following the loose routing, a channel router is used to determine the detailed routing within each channel, and hence also the exact distances between modules.

The other approach, which is followed in this paper, is the methodology used in the AT&T Bell Laboratories CADRE experimental VLSI design system (Ackland 1985; Watanabe 1986; Yu 1987) and in the IBM Planar Package Planner (PPP) (Heller 1982; Maling 1982) design system. In this CADRE/PPP approach, all interconnections are between adjacent modules; as explained in CADRE, this can always be accomplished by the inclusion of "routing modules" (which perform no function except to hold wiring). Interconnection is accomplished by having the modules abut one another; no inter-module routing space is defined. Instead we assume that, if a connection exists between two adjacent modules, then there are adjacent pins on the two modules to achieve this interconnection. Any necessary wiring to allow such pins to be adjacent is assumed to occur within the modules, and hence is transparent to the floorplanner. However, the interconnections are constraints to the floorplan: a requirement of n interconnections between two adjacent modules is represented by requiring that the common boundary between the two modules have length at least $I(n)$ for some known function I.

In CADRE/PPP, relative positions of the modules are specified usng a "relative position graph". In CADRE, this is done, in a rule-based approach, by Flute (Watanabe 1986). Once the relative positions are defined, an algorithm is invoked to determine the optimal shape of each module, subject to the relative orientations dictated by the relative position graph and interconnection requirements, to minimize the chip area. In CADRE, the sizing algorithms of Flute or Fork (Yu 1987) are used. Finally, the detailed floorplan is evaluated. CADRE allocates the job of evaluating the floorplan generated by Flute or Fork to a "floorplan evaluator" expert agent, which, like Flute and Fork, is under the control of a "manager". If the manager is unhappy with the current floorplan, new constraints (e.g., revised relative positions, interconnection requirements, or minimum module areas) are imposed, and the floorplan is redone.

As stated above, the job of module sizing in Flute, Fork, and PPP is to determine the sizes (height and width) of each module to minimize the chip area, subject to the relative position constraints and interconnection constraints. Various heuristics are used: Flute and Fork iterate between improving horizontal (x) or vertical (y) module dimensions, while PPP modifies the Simplex method (Dantzig 1963). All of these methods are heuristics, not guaranteed to solve the problem exactly (this is explicitly stated in p. 668 of Maling 1982). There are two negative consequences of using such a heuristic: first, it is not known how far the heuristic solution is from optimal, and second, it is not obvious when to terminate the heuristic.

The contribution of this paper is to show that the floorplan can in fact be solved *exactly*. This is accomplished by transforming the problem into a special type of nonlinear programming problem (NLP). The constraints and objective function of the NLP are convex functions, which means that any local minimum is a *global* solution of the problem. We illustrate the method by applying it to the example of Maling (1982).

The paper is organized as follows. In Section 2 we define the problem, and a mathematical formulation is presented in Section 3. In Section 4 we show how the mathematical formulation can be transformed to yield a convex nonlinear programming problem. Computational results and future work are discussed in Section 5.

2 Description of the Problem

It is convenient to group the data necessary for the floorplan problem into three parts: (i) a relative position graph giving the relative positions of the modules, (ii) constraints on the shape of modules, and (iii) inter-module interconnection and inter-module one-dimensional constraints.

2.1 The Position Graph

Figure 1 is an example relative position graph. There are five modules (M1–M5) to interconnect. Four boundary circles (*North, South, East,* and *West*) representing the four sides of the chip have been added. An edge between modules p and q means that M_p and M_q overlap on one boundary; they overlap on the bottom edge of M_p and the top edge of M_q if M_p is above M_q in the graph, they overlap on the right edge of M_q and the left edge of M_p if M_p is to the right of M_q in the graph. Thus, for Fig. 1, the bottom boundary of M_1 overlaps the top boundaries of M_3 and M_4, and the right boundary of M_4 overlaps the left boundaries of M_2 and M_5. Also, since M_1 and M_3 are connected to *West,* their left boundaries form the left edge of the chip.

The relative position graph must be planar. If not, a routing module, and some new edges, can be added to the graph to achieve planarity, as described in CADRE.

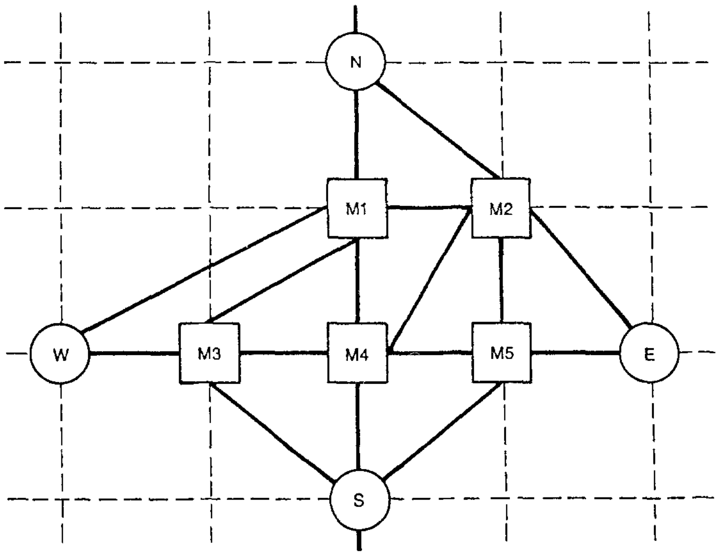

Fig. 1. A relative position graph

2.2 Module Shape Constraints

Let w_p and h_p be the width and height, respectively, of M_p. We permit two types of shape constraints.

2.2.1 Minimum Area Constraints.

A minimum area constraint is of the form $w_p \cdot h_p \geqslant a_p$ for some constant a_p.

2.2.2 Minimum Height or Width Constraints

A minimum height constraint is of the form $h_p \geqslant \bar{h}_p$ for some constant $\bar{h}_p$. A minimum width constraint is of the form $w_p \geqslant \bar{w}_p$ for some constant $\bar{w}_p$.

2.3 Inter-Module Interconnection and One-Dimensional Constraints

Let w_p, h_p, w_q, h_q be the width and height, respectively, of modules M_p and M_q.

2.3.1 *Inter-Module Interconnection Constraints*

A horizontal inter-module interconnection constraint is of the form "the top boundary of M_p must overlap the bottom boundary of M_q by at least c_{pq} units, for some constant c_{pq}." If M_p is to the right of M_q, a vertical inter-module interconnection constraint is of the form "the left boundary of M_p must overlap the right boundary of M_q by at least d_{pq} units, for some constant d_{pq}." We show in Section 3 how these constraints can be expressed mathematically with a change of variables.

2.3.2 *Inter-Module One-Dimensional Constraints*

These constraints are of the form $w_p = w_q$ ($h_p = h_q$); i.e., M_p and M_q must be the same width (height).

2.4 The Objective Function

The problem is to minimize the chip area subject to the above constraints. Given a relative position graph G, the chip area is given by $D_w \cdot D_h$, where $D_w \equiv$ *distance* (*West–East*) and $D_h \equiv$ *distance* (*North–South*). To compute D_w, we must find the longest path from *East* to *West* on the graph (and similarly for D_h). Thus, for Fig. 1, $D_w \equiv \max \{w_1 + w_2, w_3 + w_4 + w_5\}$. Note that we need not also consider the path $w_3 + w_4 + w_2$: since M_2 and M_5 share a common left boundary with M_4 and a common right boundary with *East*, then we must have $w_2 = w_5$.

3 Mathematical Formulation of the Problem

We imagine the chip to sit in the positive orthant of R^2 so that the origin coincides with the south-western corner of the chip. We measure the module heights and widths, and hence the chip height and width, relative to the origin. Let x_i, $i = 1, 2, \ldots, I$ (y_j, $j = 1, 2, \ldots, J$) be the partially ordered sequence of x-coordinates (y-coordinates) corresponding to the position of the right hand (top) boundaries of the modules. By partially ordered we mean that some inequalities of the form $x_i < x_j$ may be specified. These inequalities can be determined immediately by inspection of the relative position graph. For example, the requirement that one module is to the right of another yields an inequality in the corresponding x_i variables. Similarly, the y_j variables may be partially ordered. Here x_I (y_J) is the distance to *East* (*North*), i.e. x_I (y_J) is the width (height) of the chip. This representation in terms of x_i and y_j is not new – see Ackland (1985).

At present, the determination of the x_i and y_j variables is not automated, but is determined manually from the relative position graph and the inter-module interconnection and one-dimensional constraints. (A scheme for automatically generating the variables is given by Yu 1987.) In this process, note that if two modules M_1 and M_2 both share a common right boundary with another module M_3 (or with *East*), then only one x_i variable is needed to represent the right boundaries of M_1 and M_2 (and similarly for top boundaries). Figure 2, which represents the example of Maling (1982), illustrates both the w_i and h_i variables for M_i and the corresponding x_i and y_j variables. The relationships between the variables are: $h_1 = y_3 - y_1$, $h_2 = y_3 - y_1$, $h_3 = y_3 - y_2$, $h_4 = y_1$, $h_5 = y_1$, $h_6 = y_2$, $w_1 = x_4 - x_2$, $w_2 = x_2 - x_1$, $w_3 = x_1$, $w_4 = x_4 - x_3$, $w_5 = x_3 - x_1$, and $w_6 = x_1$.

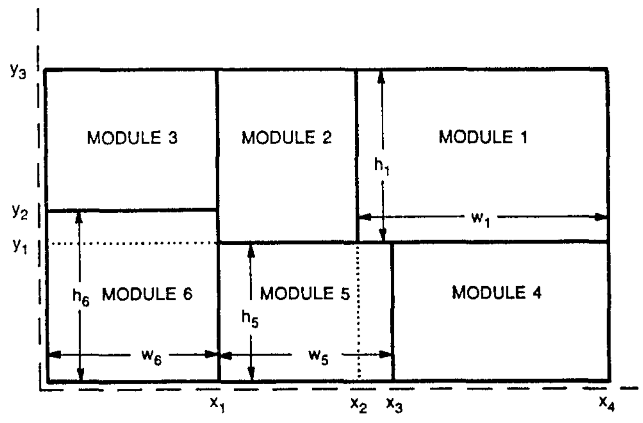

Fig. 2. Relationship of x, y, w, h variables

Following Watanabe (1986), we may now state the floorplanning problem as the nonlinear programming problem (NLP). Let *VER*1 (*HOR*1) be the set of all module height (width) constraints, so that $p \in VER1$ means that module p has a height constraint. Let *VER*2 (*HOR*2) be the set of all inter-module vertical (horizontal) interconnection constraints. That is, $VER2 = \{(p, q) | p$ and q have a vertical interconnection constraint$\}$. Let A be the set of all module minimum area constraints. Note that the inter-module one-dimensional constraints need not be explicitly modeled; they were used, together with the relative position graph, to generate the partially ordered sets of positive variables $\{x_i\}$ and $\{y_j\}$. Let $y_{t(p)}$ be the y variable corresponding to the top edge of module p. Let $y_{b(p)}$ (bottom edge), $x_{r(p)}$ (right edge), and $x_{l(p)}$ (left edge) be similarly defined. Note that these variables are not necessarily distinct (i.e., $x_{r(p)} = x_{r(q)}$ if modules p and q are both connected to *East* in the graph). The floorplanning problem *NLP* may now be written as

$$\textit{minimize } x_I \cdot y_J$$

$$\textit{subject to: } x_{r(p)} - x_{l(p)} \geqslant \bar{w}_p, \quad p \in HOR1,$$

$$y_{t(p)} - y_{b(p)} \geqslant \bar{h}_p, \quad p \in VER1,$$

$$x_{r(p)} - x_{l(q)} \geqslant c_{pq}, \quad (p, q) \in HOR2,$$

$$x_{r(q)} - x_{l(p)} \geqslant c_{pq}, \quad (p, q) \in HOR2,$$

$$y_{t(p)} - y_{b(q)} \geqslant d_{pq}, \quad (p, q) \in VER2,$$

$$y_{t(q)} - y_{b(p)} \geqslant d_{pq}, \quad (p, q) \in VER2,$$

$$(y_{t(p)} - y_{b(p)}) \cdot (x_{r(p)} - x_{l(p)}) \geqslant a_p, \quad p \in A,$$

$$x_i > 0, \quad i = 1, 2, \ldots, I,$$

$$y_j > 0, \quad j = 1, 2, \ldots, J.$$

Note that the inter-module interconnection constraints appear as two inequalities Watanabe (1986). This is illustrated in Fig. 3.

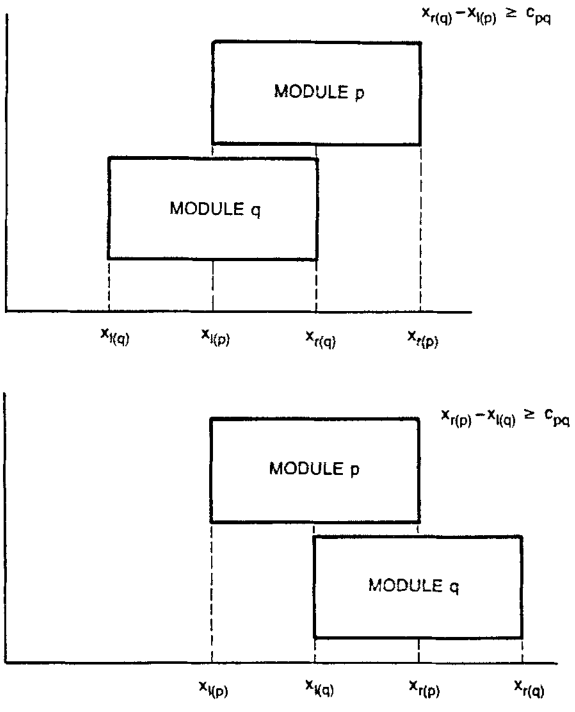

Fig. 3. Horizontal inter-module interconnection constraints

4 Optimal Solution Method

In this section we describe our new solution method, which is guaranteed to find the globally optimal solution (as opposed to a locally optimal solution) of the floorplanning problem. The key to the method is to make a transformation of variables and rewrite the constraints as convex functions.

Program NLP is, at first glance, very nasty. It has a nonlinear objective function, and quadratic constriants. Moreover, neither the objective function nor the constraints are convex. However, it turns out that by a simple (and rather esoteric) change of variables, and rewriting the constraints, we can reformulate NLP as a *convex* program, that is, a program whose objective function and constraints are convex. Convex programs have the special property that each locally optimum solution is a global optimum. Moreover, nonlinear programming algorithms behave especially well when applied to convex programs (Gill 1981, p. 257).

To explain the reformulation, consider first a module width constraint: $x_{r(p)} - x_{l(p)} \geqslant \bar{w}_p$. (Although this constraint is linear, and hence convex, it serves as our starting point.) For ease of notation, let $r = r(p)$ and $l = l(p)$. Then the constraint may be written as $x_r \geqslant x_l + \bar{w}$, or $\bar{w}x_r^{-1} + x_l x_r^{-1} \leqslant 1$, where $x_i^{-1} = (x_i)^{-1}$. Note that, since each x_i is positive, the division is allowed. Now for the change of variables: for $i = 1, \ldots, I$ let $x_i = e^{S_i}$ and for $j = 1, \ldots, J$ let $y_j = e^{T_j}$. Then the width constraint may be rewritten as

$$\bar{w}e^{-S_r} + e^{S_l - S_r} \leqslant 1, \tag{5}$$

which is *convex* in the variables S_r and S_l. (It is easily shown, using the convexity of the exponential function, that if f_l, $l = 1, \ldots, L$ are constants then $e^{\sum_{l=1}^{L} f_l z_l}$ is convex in the variables z_l, $l = 1, \ldots, L$.) Since the module height constraints and the inter-module interconnection constraints have a similar form, they can be transformed in the same manner. So far, nothing has been gained, since these constraints were convex to begin with. The value is in transforming the objective function and area constraints.

The objective function may be written as

$$e^{S_I + T_J}, \tag{6}$$

which is now convex in S_I and T_J. Consider next the area constraint $(x_{r(p)} - x_{l(p)}) \cdot (y_{t(p)} - y_{b(p)}) \geqslant a_p$. Since $x_{r(p)} - x_{l(p)} > 0$ and $y_{t(p)} - y_{b(p)} > 0$, we may define two new positive variables, α_p and β_p and rewrite the area constraints as the set of three constraints

$$\alpha_p \cdot \beta_p \geqslant a_p, \tag{7.1}$$

$$x_{r(p)} - x_{l(p)} \geqslant \alpha_p, \quad \text{and} \tag{7.2}$$

$$y_{t(p)} - y_{b(p)} \geqslant \beta_p. \tag{7.3}$$

Note that inequalities, rather than strict equalities, may be used in the final solution – it can be shown, using the Kuhn-Tucker optimality conditions of nonlinear programming, that the inequalities will be satisfied as strict equalities for an optimal solution of the problem. We next introduce two new variables U_p and V_p defined by $\alpha_p = e^{U_p}$ and $\beta_p = e^{V_p}$. Constraints (7.1)–(7.3) may be written as

$$e^{U_p} \cdot e^{V_p} \geqslant a_p, \tag{8.1}$$

$$e^{S_{r(p)}} - e^{S_{l(p)}} \geqslant e^{U_p}, \quad \text{and} \tag{8.2}$$

$$e^{T_{t(p)}} - e^{T_{b(p)}} \geqslant e^{V_p}, \tag{8.3}$$

which in turn can be written as the *convex* set of constraints

$$a_p \cdot e^{-U_p - V_p} \leqslant 1, \tag{9.1}$$

$$e^{U_p - S_{r(p)}} + e^{S_{l(p)} - S_{r(p)}} \leqslant 1, \quad \text{and} \tag{9.2}$$

$$e^{V_p - T_{t(p)}} + e^{T_{b(p)} - T_{t(p)}} \leqslant 1. \tag{9.3}$$

Thus we have shown that the floorplanning problem can be written as

$$\textit{minimize } e^{S_I + T_J} \tag{10.1}$$

$$\textit{subject to: } \bar{w}e^{-S_{r(p)}} + e^{S_{l(p)} - S_{r(p)}} \leqslant 1, \quad p \in HOR1, \tag{10.2}$$

$$\bar{h}e^{-T_{t(p)}} + e^{T_{b(p)} - T_{t(p)}} \leqslant 1, \quad p \in VER1, \tag{10.3}$$

$$c_{pq}e^{-S_{r(p)}} + e^{S_{l(q)} - S_{r(p)}} \leqslant 1, \quad (p, q) \in HOR2, \tag{10.4}$$

$$c_{pq}e^{-S_{r(q)}} + e^{S_{l(p)} - S_{r(q)}} \leqslant 1, \quad (p, q) \in HOR2, \tag{10.5}$$

$$d_{pq}e^{-T_{t(p)}} + e^{T_{b(q)} - T_{t(p)}} \leqslant 1, \quad (p, q) \in VER2, \tag{10.6}$$

$$d_{pq}e^{-T_{t(q)}} + e^{T_{b(p)} - T_{t(q)}} \leqslant 1, \quad (p, q) \in VER2, \tag{10.7}$$

$$a_p \cdot e^{-U_p - V_p} \leqslant 1, \quad p \in A, \tag{10.8}$$

$$e^{U_p - S_{r(p)}} + e^{S_{l(p)} - S_{r(p)}} \leqslant 1, \quad p \in A, \tag{10.9}$$

$$e^{V_p - T_{t(p)}} + e^{T_{b(p)} - T_{t(p)}} \leqslant 1, \quad p \in A. \tag{10.10}$$

Program (10.1)–(10.10), which we call *CP* (for Convex Program) is in fact a convex program equivalent to *NLP*: given a solution $\{S^*, T^*, U^*, V^*\}$, the solution to *NLP* is given by $x_i^* = e^{S_i^*}$, $i = 1, 2, \ldots, I$, and $y_j^* = e^{T_j^*}$, $j = 1, 2, \ldots, J$.

Notice that *CP* has a very special structure: the objective function, and each constraint, is of a very special form, namely, a sum of functions of the form $c \cdot e^{\sum_{l=1}^{L} f_l z_l}$, where $c > 0$, and each f_l is a real number. Such nonlinear programs are called *geometric* programs, and their study forms a sub-speciality within the field of nonlinear programming. Geometric programming was "discovered" by Clarence Zener (1971). The elegant mathematical theory of geometric programs is largely due to Duffin, Peterson, and Zener (Duffin 1967) and still remains an area for current research, e.g., (Rosenberg 1981; Rosenberg 1982). Geometric programming, from the start, was found to be especially suitable for modelling and optimization of a wide range of engineering problems (including chemical equilibria and ship design (Wilde 1978)) so it is not surprising that it can be applied to "truly geometric" problems such as VLSI floorplanning.

5 Computational Results and Future Directions

In order to verify the above mathematics, and to demonstrate that computer solution of nonlinear programs need not be burdensome, we solved the floorplanning problem given in Maling 1982, p. 666. (This is the only published example we are aware of.) This example has six modules, six module height constraints, six module width constraints, six minimum module area constraints, three inter-module one-dimensional constraints, and two inter-module interconnection constraints. The resulting convex program *CP* has 15 variables and 25 nonlinear constraints.

To solve the problem, we used the nonlinear programming code *E*04*VDF* from the *NAGFLIB* library NAG (1984). (This software, designed for general (non-convex) nonlinear programs, was chosen because a geometric programming code, exploiting the special structure of geometric programs, was not available.) A small program (80 lines of FORTRAN), calling *E*04*VDF* and supplying function and gradient values for the objective function and constraints, was written. A solution was found in 1.15 seconds CPU time on a CRAY X-MP. The solution is given in Fig. 4. We note that our solution coincides exactly with (and thus verifies the correctness of) the solution found in Maling (1982). However, because no convergence proof is given in Maling (1982), this indicates only that their heuristic method is successful on this one problem. With the nonlinear programming method described here, the *provably global optimal* solution

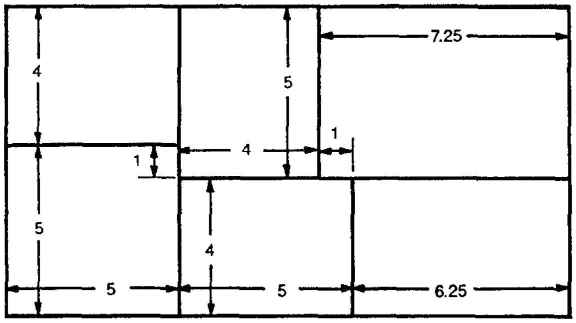

Fig. 4. Optimal floorplan

to the floorplanning problem has been found with extremely modest computational effort.

It is important to realize that there are two possible reasons for the success of the IBM heuristic on the one problem solved. First, it could be luck, and on other problems the method might not obtain the global optimum. Second, it could be that, despite the non-convexity of problem *NLP*, the fact that it is equivalent to (i.e., can be transformed into) a convex program means that any solution method (whether exact or heuristic) will do just as well as if it were applied to the transformed convex program, and thus a good heuristic will in fact obtain the global optimum. As mentioned above, the authors of PPP did not rule out the possibility of getting caught in a local minimum. We have not been able to resolve this issue theoretically.

Future plans are for further experiments with the method, using actual AT&T floorplanning problems, in order to determine if it should be incorporated into CADRE. Of particular concern is understanding how the CPU time to solve the nonlinear program increases as a function of the problem size. Incorporating the method into CADRE would require writing interface software which, given the relative position graph and constraints from CADRE, defines the x_i and y_j variables and sets up files so the function values and gradients can be computed.

Acknowledgments: The author benefited greatly from conversations with Brian Ackland, Mark Yu, and Ravi Varadarajan, who also made valuable comments on a draft of this paper.

References

Ackland B, Dickenson A, Ensor R, Gabbe J, Kollaritsch P, London T, Poirier C, Subrahmanyam P, Watanabe H (1985) CADRE – A system of cooperating VLSI design experts. Proc. 1985 International Conference on Computer Design, pp 99–104

Dantzig GB (1963) Linear programming and extensions. Princeton University Press, New Jersey

Duffin RJ, Peterson EL, Zener C (1967) Geometric programming. Wiley, New York

Gill PE, Murray W, Wright MH (1981) Practical optimization. Academic Press, New York

Heller WR, Sorkin G, Maling K (1982) The planar package plannar for system designers. Proc. 19 Design Automation Conference, pp 253–260

Maling K, Mueller SH, Heller WR (1982) On finding most optimal rectangular package plans. Proc. 19 Design Automation Conference, pp 663–670

NAG Fortran Library Manual, Mark 11, Volume 1 (1984) Numerical analysis group. Mayfield House, Oxford, UK

Rosenberg E (1981) On solving a primal geometric program by partial dual optimization. Mathematical Programming 21:319–330

Rosenberg E (1982) A globally convergent condensation method for geometric programming. Utilitas Mathematica 22:47–64

Watanabe H, Ackland B (1986) Flute – a floorplanning agent for full custom VLSI design. Proc. 23 Design Automation Conference, pp 601–607

Wilde DJ (1978) Globally optimal design. Wiley, New York

Yu M-LM (1987) A new floorplan representation for VLSI design. Proc. 1987 IEEE Custom Integrated Circuits Conference, pp 625–628

Zener C (1971) Engineering design by geometric programming. Wiley, New York

Received 1. 7. 1988
Revised version received 3. 10. 1988