A Transistor Placement Technique Using Genetic Algorithm and Analytical Programming

Cristiano Lazzari[1,2] , Lorena Anghel[2], Ricardo A. L. Reis[1]
[1]*PGMICRO - Universidade Federal do Rio Grande do Sul*
Porto Alegre - RS, Brazil
E-mail:{clazz,reis}@inf.ufrgs.br

[2]*TIMA Laboratory - Institute National Polytechnique de Grenoble*
Grenoble - France
E-mail:lorena.anghel@imag.fr

Abstract. New technologies present a widely range of challenges in the design of standard-cell libraries, layout generation and validation of macro-blocks. Thus, the development of new tools being able to deal with these challenges is mandatory. This work presents a transistor placement technique using genetic algorithm associated to analytical programming. The genetic algorithm is used to reduce the search space of possible solutions while analytical equations are used to find out the position of each transistor in the layout.

1. Introduction

The layout automation of standard-cells and macro-blocks improves the design time due to a rapidly synthesis and this enables the designer to deal with a great range of challenges emergent in new process technologies.

New technologies challenges require additional functionalities as performance-driven placement, antenna diode placement, area-efficient placement of substrate and well ties, performance-driven detailed routing and layout compaction with preference to critical nets [1]. These new challenges increase the complexity of the existing tools and demand the development of new algorithms and methods to be used in the layout automation.

Lazzari, C., Anghel, L., Reis, R.A.L., 2007, in IFIP International Federation for Information Processing, Volume 240, VLSI-SoC: From Systems to Silicon, eds. Reis, R., Osseiran, A., Pfleiderer, H-J., (Boston: Springer), pp. 331–344.

This paper addresses the problem of transistor placement in the development of standard-cell and macro-block layouts by using a genetic algorithm integrated with a mathematical programming. The genetic algorithm provides the parameters used in transistor placement constraints and reduces the search space. These constraints are described in a mathematical language and treated by a nonlinear solver. The result is an optimal transistor placement solution given these placement parameters.

Section 2 presents a brief description of state-of-the-art and previous works. The proposed technique is described in Section 3 and 4. The parameters used in the placement constraints are presented in Section 5. Section 6 presents the mathematical language used in the transistor placement. Some preliminary results are given in Section 7 and the paper concludes with Section 8.

2. Related Work

The synthesis of standard-cell and macro-block layouts has been widely explored. In [2, 3], the placement algorithms are broadly divided into two classes: **Deterministic** and **Stochastic**.

Deterministic methods are basically divided in numerical methods and analytical methods. The forced-directed technique [4, 5] is an example of *numerical method* where elements are connected to springs. In this technique, forces are applied to springs targeting the placement of the elements. *Analytical methods* [2, 3, 6, 7] are based on mathematical programming techniques as linear programming (LP) and quadratic programming (QP). Thus, the placement problem is described in a mathematical language. Once the method is able to solve these equations, the result is the placement of each transistor in the layout.

The main examples of stochastic methods are known as *simulated annealing* and *genetic algorithms*. The simulated annealing [8] is analogous to hardware annealing process. It basically involves perturbing independent variables by random values while the temperature controls the standard deviation used by the random number generator. Genetic algorithms [9] use basic principles of biology and emulates the natural process of evolution to find solutions to a problem.

Despite the number of transistor placement algorithms proposed in the literature, many of them do not offer a good compromise between the quality of results and the model complexity. The proposed approach achieves to obtain good quality layouts by using genetic algorithms associated to the analytical programming.

3. The Transistor Placement Technique

Some design problems as transistor placement have a large range of possible solutions. These problems are computational hard or even impossible to be solved.

For some of them, methods as simulated annealing and the genetic algorithm can be used to reduce the search space.

In the genetic algorithm, each solution is represented by a *chromosome*. A chromosome is usually composed by a binary vector where the variables formed by one or more bits are described. A population of chromosomes (possible solutions) is then created and genetic operators as mutation and crossover are applied in order to evolve the solutions to better results.

The approach presented in this work is basically divided in three phases. First, a classical genetic algorithm is used to generate some parameters concerning transistor orientation and the relationship between them. These parameters are used as placement constraints described in an algebraic modeling language. The second phase consists on solving the placement constraints by a nonlinear solver in order to find the optimal solution according to the given constraints. After that, the best solutions are propagated and genetic operators are applied to the solutions.

The pseudo code of the proposed approach is presented in Figure 1. An initial set of solutions is generated in the function `generatePopulation(N)` where each chromosome in the population *P* has a set of constraints about the transistor placement problem. The generation of this initial population is explained in Section 4.

```
P = generatePopulation( N );
do N times {
   foreach k in P {
      solveConstraints( k );
      calculateFitness( k );
   }
   P = doEvolution( P );

   }
```

Figure 1. The proposed approach

In function `solveConstraints(k)`, the parameters of the chromosome *k* are converted to an algebraic modeling language and the placement problem is solved.

The fitness of a chromosome is generated in the function `calculate-Fitness(k)`. The fitness of a chromosome is calculated based on the objective function as described in Section 6.

The function `doEvolution(P)` is basically the reproduction of the chromosomes in the population *P* to generate a new population with better results. In the generation of this new population, operations of elitism, mutation and crossover are applied to the chromosomes in order to propagate the best solutions and to evolve the other chromosomes.

4. Initial Population Generation

The range of possible solutions in the process of the layout generation is related to the number of elements in a cell or in a macro-block. Moreover, the relation between these elements makes a solution better than the others. Thus, some techniques can be used to reduce the number of elements and consequently, decreasing the complexity of the layout generation problem.

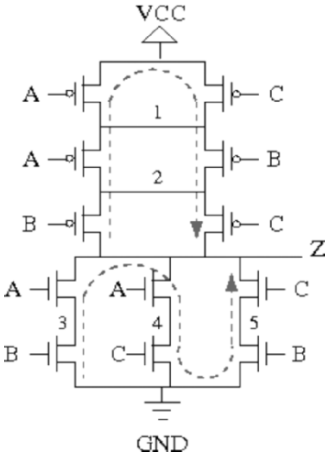


Figure 2. An Euler path example

Transistor chaining is a technique that consists of grouping transistors when their drain/source diffusions can be shared. Figure 2 illustrates the transistor chaining generation where the *Euler* path is searched to PMOS and NMOS transistors. Dashed lines show examples of *Euler* paths in which a chain of transistors is performed based on the sharing of the source/drain diffusion areas.

In this example, the two transistors chains are (*Z,B,2,A,1,A,VCC,C,1,B,2,C,Z*) to the PMOS transistors and (*GND,B,3,A,Z,A,4,C,GND,B,5,C,Z*) to the NMOS transistors.

Its is clear that many solutions can be found to these set of transistors. In the approach proposed in this work, an *Eulerian* graph is used in order to generate the *N* solutions related to the initial population. Transistor chainings are randomly chosen to be used in the genetic algorithm.

5. The Placement Parameters

Each chromosome in the genetic algorithm is a set of parameters used in the placement constraints. Parameters used in transistor placement are basically the

description of transistors orientation and the relationship between these transistors. Transistor orientation means whether a transistor must be placed horizontally or vertically and where the drain/source contacts are located, while the relationship between transistors is the relative placement of a transistor in relation to each other transistor.

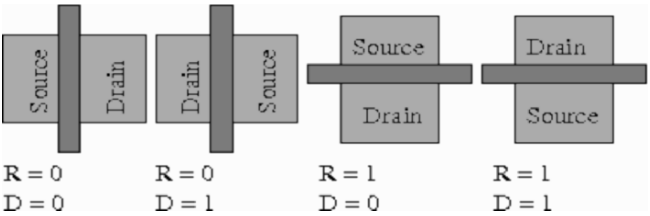


Figure 3. Transistor Orientation Constraints

Figure 3 illustrates the orientation constraints R and D. The parameter R represents the orientation of the transistors. R=0 indicates that the transistor must be placed horizontally and R=1 means that the transistor must be placed vertically.

The parameter D indicates where drain/source diffusion areas are located. D=0 means that the transistor source area is located in the left/top and D=1 means that the drain area is located in the left/top of the transistor.

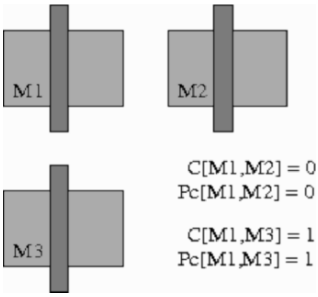


Figure 4. Transistor behavior constraints

The relationship between transistors is shown in Figure 4. The parameters C and Pc are used to describe these relationship. C indicates whether the placement constraints are related to horizontal or vertical coordinates and Pc represents the relative position of these transistors.

Taking as example the transistors *M1*, *M2* and *M3* illustrated in Figure 4, C[M1,M2] = 0 means that the transistors *M1* and *M2* are placed side by side

horizontally and Pc[M1,M2] = 0 indicates that the transistor *M1* is placed in the left side of *M2*. In other words, $X_{M1} < X_{M2}$ and there is no requirements to coordinate *Y*.

The same idea is used when C[M1,M3] = 1 and Pc[M1,M3] = 1. In this case $Y_{M1} > Y_{M3}$ and any horizontal constraint is applied. Table 1 shows the possible constraints resulting of the parameters C and Pc.

Table 1. Parameters to the transistors relationship

Parameters		Constraints	
C	Pc	Horizontal	Vertical
0	0	$X_1 < X_2$	-
0	1	$X_1 > X_2$	-
1	0	-	$Y_1 < Y_2$
1	1	-	$Y_1 > Y_2$

Based on these parameters, each chromosome is a binary vector containing information about orientation and relationship between transistors. The size of a chromosome is given by the Equation 1:

$$L_{chrom} = T * 2 + \sum_{i=1}^{T-1} i * 2 \tag{1}$$

where *T* is the number of transistors. The first part of the equation 1 is related to parameters R and D, and the second part is related to parameters C and Pc.

6. The Mathematical Modeling

Once the parameters are defined in the chromosomes, they can be applied in an algebraic modeling in order to obtain the optimal placement solution with the respect to the given parameters and constraints. The main idea of this approach is to use a nonlinear solver to find the solution to the placement of transistors.

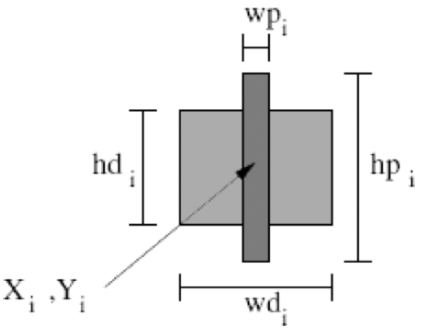


Figure 5. Width and height transistor parameters

Figure 5 shows the width and the height parameters used in the placement constraints. To each transistor $i \in T$, the parameters wd_i and hd_i are the width and the height of the diffusion area while wpi and hp_i are the parameters to the polysilicon area. Besides that, other three integer parameters $drain_i$, $source_i$ and $gate_i$ represent the connections of the transistors and the parameter $type_i$ is also used to indicate if the transistor i is PMOS or NMOS.

The variables X_i and Y_i are the central coordinates of the transistor i. Their values are given by the minimization of the objective function. The goal of the used objective function is to find the optimal X_i and Y_i by the minimization of the wire lengths. The specification of the objective function is given in more detail in Section 6.

The constraints are divided in three groups: 1) Boundary Constraints, 2) Neighborhood Constraints and 3) Connections Constraints. These constraints are presented in the following sections.

6.1 Boundary Constraints

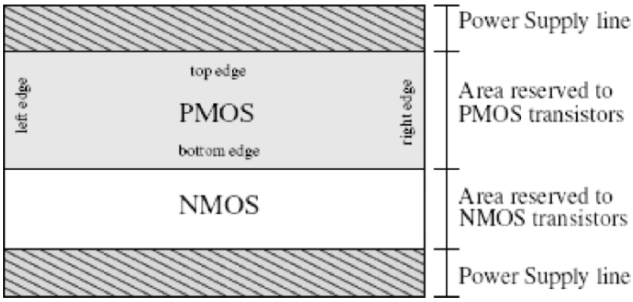


Figure 6. A Row-based boundary representation

The layout of standard-cells and macro-blocks are usually structured in rows. In these structures, layout boundaries must be regular in order to allow the connection between adjacent cells at the moment of the entire circuit generation. Figure 6 illustrates the boundaries in a row-based layout.

Regions of PMOS and NMOS transistors may be determined by the implant areas and boundary constraints may be formulated according to the edges of these areas. Thus, the boundary constraints are given by

$$B_{left} + \Delta_x + \frac{1}{2}W_i \leq X_i \leq B_{right} - \Delta_x - \frac{1}{2}W_i \tag{2}$$

$$B_{bottom} + \Delta_y + \frac{1}{2}H_i \leq Y_i \leq B_{top} - \Delta_y - \frac{1}{2}H_i \tag{3}$$

where B_{left}, B_{right}, B_{bottom} and B_{top} are the edges of the placement region, Δ_x and Δ_y are the minimal distances from the transistor i to the boundaries, W_i and H_i are the width and height of the transistor.

6.2 Neighborhood Constraints

Neighborhood constraints are related to the possibility to connect transistors together. These constraints are separated into categories and they are responsible to give the correct distance between two adjacent transistors.

In order to verify the possibility of connection between transistors, the variables *left*, *right*, *top* and *bottom* are used. They are given by the following equations:

$$\begin{aligned} left_i = {} & (D_i + R_i * D_i) * drain_i + \\ & (1 - D_i + R_i * D_i) * source_i + R_i * gate_i \end{aligned} \tag{4}$$

$$\begin{aligned} right_i = {} & (1 - D_i + R_i * D_i) * drain_i + \\ & (D_i + R_i * D_i) * source_i + R_i * gate_i \end{aligned} \tag{5}$$

$$\begin{aligned} top_i = {} & (R_i - R_i * D_i) * source_i + \\ & (R_i * (1 - D_i)) * drain_i + (1 + R_i) * gate_i \end{aligned} \tag{6}$$

$$\begin{aligned} bottom_i = {} & (R_i - R_i * D_i) * drain_i + \\ & (R_i * (1 - D_i)) * source_i + (1 + R_i) * gate_i \end{aligned} \tag{7}$$

where $i \in T$, R_i and D_i are the parameters given by the current chromosome. $drain_i$, $source_i$ and $gate_i$ are integer parameters related to the list of connections C.

Considering K_c the number of points of the connection c and assuming that $c \in C$, it is possible to know when two transistors are connected in serial or parallel. Thus,

two transistors are in serial always that $K_c=2$. In all other cases the transistor are in parallel or they are not connected.

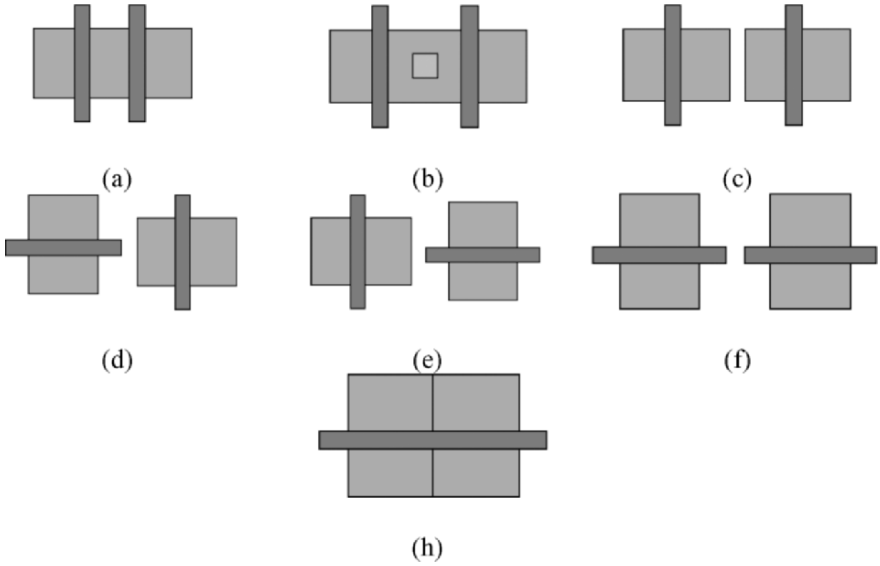


Figure 7. Graphical Repesentation of the neighborhood constraints

Table 2. Horizontal neighborhood constraints

#	*Orientation*		*Kc*	*Situation*	*Constraint*
	R_i	R_j			
1	0	0	=2	$right_i$ = $left_i$	$X_j - X_i \geq \frac{1}{2}wp_i + sp + \frac{1}{2}wp_j$
2	0	0	≠2		$X_j - X_i \geq \frac{1}{2}wp_i + 2 \times spc + wc + \frac{1}{2}wp_j$
3	0	0		$right_i$ ≠ $left_i$	$X_j - X_i \geq \frac{1}{2}wp_i + sd + \frac{1}{2}wp_j$
4	1	0			$X_j - X_i \geq \frac{1}{2}hp_i + sdp + \frac{1}{2}wd_j$
5	0	1			$X_j - X_i \geq \frac{1}{2}wd_i + sdp + \frac{1}{2}hp_j$
6	1	1		Different top, bottom or left/right	$X_j - X_i \geq \frac{1}{2}hd_i + \frac{1}{2}hd_j$
7	1	1		Top, bottom and left/right equal	$X_j - X_i \geq \frac{1}{2}hp_i + sp + \frac{1}{2}hp_j$

From the definition of these variables, it is possible to understand how the neighborhood constraints are formulated. Table 1 presents the neighborhood constraints where *sp* is the spacing between polysilicon lines, *sdc* is the distance between a polysilicon line and a contact, *wc* is the width of a contact, *sd* is the spacing of two diffusion areas and *sdp* is the distance between a polysilicon line and a diffusion area.

Neighborhood constraints are separated in categories with the effort to deal with every possible relationship between two transistors. Only horizontal constraints are discussed here but similar equations are used vertically.

Seven different constraints are shown in Figure 7 and Table 2. In the case of $R_i=0$ and $R_j=0$, equation 1 treats situations where transistors are in serie, equation 2 deals with parallel transistors and equation 3 takes situations where transistors are not connected.

The equation 3 and 4 treat situations where there are different transistor orientation parameters ($R_i \neq R_j$). In these cases, the sharing of diffusion areas is impossible.

When transistors are placed vertically ($R_i=1$ and $R_j=1$), the connection between two transistors is possible only whether $top_i=top_j$, $right_i=left_i$ and $bottom_i=bottom_j$, and (Equation 6). Equation 7 takes all other cases to $R_i=1$ and $R_j=1$, in which the connection between transistors cannot be done.

6.3 Connection Constraints

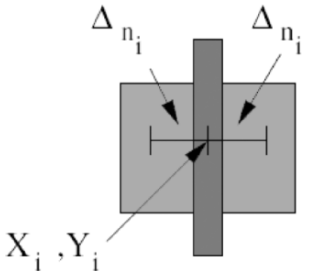


Figure 8. The Δni Representation

Let *n* be the number of connections and *m* the number of transistors, the position to gate, drain and source can be inserted in matrix notation to the horizontal and

vertical coordinates, Q_x and Q_y. Thus, Q_x and Q_y are $n \times m$ matrices where the coordinates X and Y of the nets are given by

$$Q_x(drain_i, i) = (1 - R_i) * D_i * (X_i - \Delta_{n_i}) + (1 - R_i) * (1 - D_i) * (X_i + \Delta_{n_i}) + R_i * X_i \quad (8)$$

$$Q_x(source_i, i) = (1 - R_i) * (1 - D_i) * (X_i - \Delta_{n_i}) + (1 - R_i) * D_i * (X_i + \Delta_{n_i}) + R_i * X_i \quad (9)$$

$$Q_x(gate_i, i) = X_i \quad (10)$$

where $i \in T$ and Δ_{ni} is the distance from the center of the transistor to the point where the connection is located as shown in Figure 8. The matrix to vertical coordinates Q_y is composed based on the same idea.

6.4 The Objective Function

The goal of the proposed technique is to reduce the wire length connecting the transistors. Thus, the objective function is based on the connection constraints and it is obtained by

$$OBJ : min\left\{ \sum_{c \in C} W_c * S(c) \right\} \quad (11)$$

where $S(c)$ is the half perimeter wire length and W_c is the weight of the connection c. The wire length of a connection c is calculated by the coordinates of the points of a net in the matrices Q_x and Q_y. Then, $S(c)$ is given by

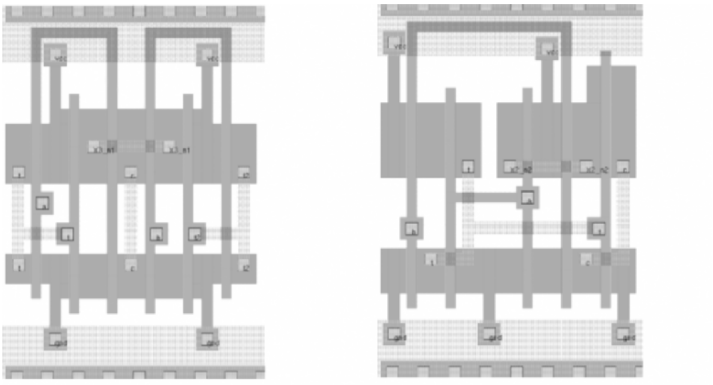
$$S(c) = \sum_{i \in T, j \in T, i \neq j} HP(c, i, j) * I(c, i) * I(c, j) \quad (12)$$

and

$$HP(c,i,j) = |Q_x(c,i) - Q_x(c,j)| + |Q_y(c,i) - Q_y(c,j)| \quad (13)$$

where $I(c,i)$ are binary values indicating whether the wire c is connected to the transistor i. The same principle is used to $I(c,j)$ with the connection c and the transistor j.

7. Cell Placement Results



(a) A 12-transistor XOR (B) A 10-transistor XOR

Figure 9. Preliminary placement examples

Figure 9 shows the placement of two cells using the proposed algorithm. The transistor placement of a 12-transistor XOR gate is shown in (a) and the placement of a 10-transistor XOR is shown in (b). These cells were routed with a simple routing algorithm. The compaction step is under development and will be applied in the layout as a last step.

Table 3. Placement results

Cell Name	Area (μm) [10]	Proposed	Gain (%)	Execution Time
NOR2	7.9	8.1	-3	2s
OR2	10.8	10.9	-1	1m 15s
AOI22	13.6	13.0	4	4m 10s
AOI222	19.4	17.9	8	18m 30s
Full Adder	52.3	45.1	13	3h 15m

Table 3 shows some results of the comparison between the proposed technique and a pure Eulerian placement algorithm used in [10]. Results show that the proposed technique deals with the transistor placement problem. The area gain is around 4.5 % .

The drawback of this technique is the execution time. While a pure Eulerian algorithm execute the placement task very quickly, the proposed technique take hours in some cases to solve the placement problem. As the genetic algorithm works with random information, the execution time presented in Table 3 is the average time of at least 5 executions of each cell.

The used mathematical language is the AMPL [11] associated to a linear/non linear problem solver called MINOS [12]. Academic versions of these softwares were used in this work.

8. Conclusion

This work presents an approach where a genetic algorithm is used in association with mathematical programming to address the transistor placement problem. The following points summarize the proposed approach when working with the transistor placement problem:

1. The search for the *Euler* path is used to create transistor chains. A set of these chains is chosen randomly and used as initial set of possible solutions;
2. A genetic algorithm is used to reduce the search space of solutions;
3. An analytical programming is solved by a non-linear solver and the solution is the optimal position to each transistor.

References

1. M. Guruswamy, R. L. Maziasz, D. Dulitz, S. Raman, V. Chiluvuri, A. Fernandez, and L. G. Jones. CELLERITY: A fully automatic layout synthesis system for standard cell libraries. In *Proceedings of the 34th Design Automation Conference*, pages 327–332, 1997.
2. S. Askar and M. Ciesielski. Analytical approach to custom datapath design. In *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers*, pages 98–101, 1999.

3. M. Ciesielski, S. Askar, and S. Levitin. Analytical approach to layout generation of datapath cells. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, number 12, pages 1480–1488, Dec 2002.
4. F. Mo, A. Tabbara, and R. Brayton. A force-directed macro-cell placer. In *IEEE/ACM International Conference on Computer Aided Design, ICCAD-2000*, pages 177–180, 2000.
5. S.-W. Hur, T. Cao, K. Rajagopal, Y. Parasuram, A. Chowdhary, V. Tiourin, and B. Halpin. Force directed mongrel with physical net constraints. In *Proceedings on the Design Automation Conference*, pages 214–219, 2003.
6. J. Kleinhans, G. Sigl, F. Johannes, and K. Antreich. Gordian: Vlsi placement by quadratic programming and slicing optimization. In *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, volume 10, pages 356–365, March 1991.
7. N. Viswanathan and C. C.-N. Chu. Fastplace: efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model. In *Proceedings of the 2004 international symposium on Physical design*, pages 26–33, 2004.
8. C. Sechen. Chip-planning, placement, and global routing of macro/custom cell integrated circuits using simulated annealing. In *25th ACM/IEEE Proceedings of the Design Automaion Conferenc*, pages 73–80, 1988.
9. A. Bahuman, B. Bishop, and K. Rasheed. Automated synthesis of standard cells using genetic algorithms. In *Proceedings on the IEEE Computer Society Annual Symposium on VLSI*, pages 126–133, 2002.
10. C. Lazzari, C. V. Domingues, J. L. Güntzel, and R. A. L. Reis. A new macro-cell generation strategy for three metal layer cmos technologies. In *Proceedings of the VLSI-Soc*, pages 143–147, Darmstadt, Germany, 2003.
11. R. Fourer, D. M. Gay, and B. W. Kernighan. *AMPL - A Modeling Language For Mathematical Programming*. Duxbury Press / Brooks/Cole Publishing Company, 2002.
12. *MINOS - Linear and Non Linear Problems Solver*. Stanford Business Software, Inc., 2005.