Placement and channel routing by simulated annealing: some recent developments

P Chaisemartin, G Dreyfus*, M Fontet**, E Kouka, P Loubières** and P Siarry*

This paper is devoted to the applications of simulated annealing to the automatic layout of circuits in the authors' laboratories. First of all, the thermodynamical origin of the problem is presented. In the second part, the use of simulated annealing for placement of circuits initialized by exploratory data analysis is exemplified. Finally, a new algorithm for full-channel routing is given.

Keywords: electronic circuits, simulated annealing, full-channel routing

The design of electronic circuits is an increasingly difficult task because of the number and complexity of components. During the past years, a very large research effort has been devoted to the development of algorithms for optimizing the design of printed circuit boards, hybrid or integrated circuits. Recently, remarkable improvements have been obtained thanks to the emergence of a new method: the technique of simulated annealing. The method is based on the idea of using the tools and concepts of statistical mechanics to obtain approximate solutions of complex engineering problems. This idea, inspired by recent advances made in the understanding of spin glasses, has been developed simultaneously and independently in several laboratories[1-3]. The principle of the method can be easily explained in the case of the design of a circuit. The optimization problem to be solved is first the automatic placement of the components, viewed as rectangular blocks of various sizes, given the 'net-list', i.e. the list of interconnections which are to be made between the blocks. The subsequent step is the automatic routing of the interconnections. The introduction of a 'temperature' in this context is very natural, as illustrated on Figure 1. Given a random initial placement and routing, which is obviously not optimum, an ideal computer-aided design program should be able to turn it into a regular placement, with rectilinear connections, thereby performing a transition between a completely disordered state and a partially ordered state. Therefore, the techniques which have been used for many years by physicists to simulate phase transitions on computers are relevant to the design of circuits.

A large research effort has been devoted to the theoretical aspects of simulated annealing method[4]. In particular, its convergence has been proved and descriptions have been proposed for annealing schedules which lead to reasonably good solutions with an acceptable computation time. Simulated annealing has been used in a variety of fields[4-8]. Its applications to the automatic design of electronic circuits[4] have shown that it is an efficient tool, which can be easily adapted to the special constraints of various technologies.

The present paper is divided into three sections. We first develop a new presentation of the fundamental ideas underlying the technique. In particular, it is shown that the analogies with physics, which have been at the origin of the method, may be partially forgotten, but that they are still useful to understand some results.

The second part is devoted to the application of simulated annealing to the automatic layout of integrated

Laboratoire Circuits et Systèmes, UA 395 du CNRS, Institut National Polytechnique de Grenoble, 46, avenue Félix Viallet, 38000 Grenoble France
* Laboratoire d'Electronique, Ecole Supérieure de Physique et Chimie Industrielles de la Ville de Paris, 10, rue Vauquelin, 75005 Paris, France
** Laboratoire Informatique Théorique et Programmation, UA 248 du CNRS, Université Pierre et Marie Curie, 4, place Jussieu, 75252 Paris Cedex 05, France
This work has been sponsored by the GCIS (Groupement Circuits Intégrés Silicium) of CNRS

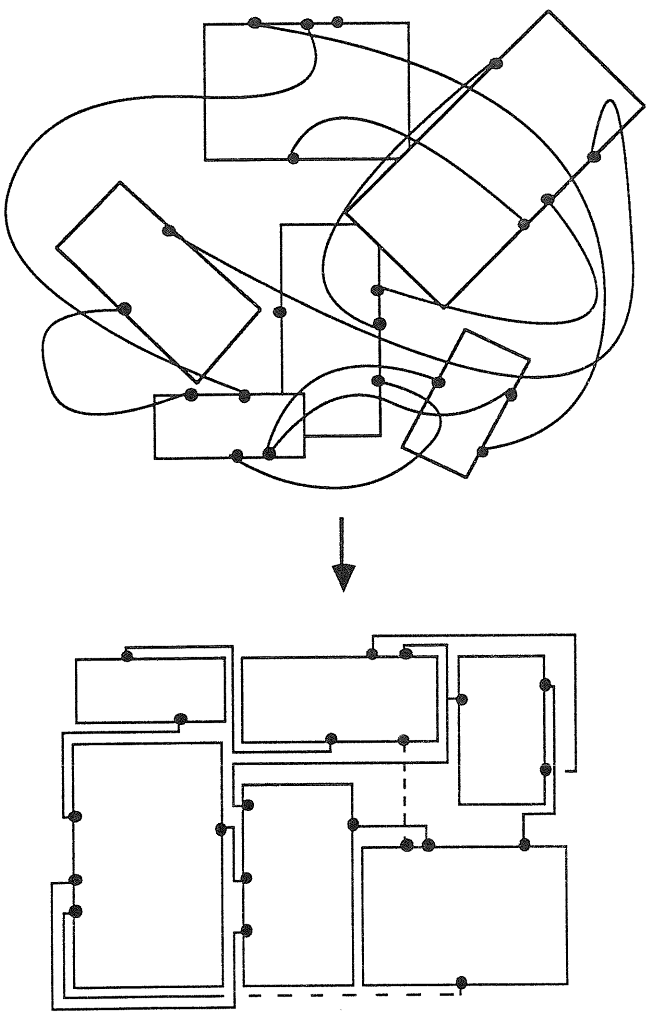


Figure 1. The design of an electronic circuit viewed as a disorder-to-order transformation

circuits. We first present the logical networks used to describe the net-list and the various ingredients of the algorithm, and we show they can be successfully adapted to the special case of placement. The originality of this section consists in the speeding up of the annealing process. This is achieved by starting the annealing algorithm with a good initial solution provided by means of exploratory data analysis techniques. Finally, we give some examples of typical results.

The third part deals with the application of simulated annealing to the routing problem. We first show how the routing problem can be expressed in terms of the graph formalism. Then, we focus on the application of simulated annealing in the specific case of full routing of channels.

THE THERMODYNAMIC OPTIMIZATION METHOD

In this section, we show how the concepts introduced by statistical physics are useful to give a microscopic explanation of classical thermodynamics. Then, in a second part, we develop the analogy with optimization problems, with emphasis on the design of electronic circuits.

The viewpoint of statistical physics

Let us consider a physical system, assumed to be perfectly insulated, so that its energy E is constant. This ensemble represents a very large system, which can be made of a large number N of subsystems in thermal contact with one another and exchanging energy. Let us consider one of these subsystems; the rest of the system is a heat bath for this subsystem.

It can easily be proved[9] that the most probable number of subsystems with energy E_i is given by:

$$n_i = N \exp(-E_i/k_B T)/\sum_i \exp(-E_i/k_B T) \qquad (1)$$

when thermal equilibrium is reached at temperature T (k_B is Boltzmann's constant)[10].

In the above distribution, the temperature T is obviously of central importance; if T is very large, all the subsystems are distributed uniformly among the possible states: developing the exponentials to first order, relation (1) becomes:

$$n_i/N \approx 1/M$$

where M is the total number of possible states, so that the total energy at infinitely large temperature, E_∞, is:

$$E_\infty = N/M \sum_{i=0}^{M} E_i = N \cdot E^0$$

where E^0 denotes the average energy of the possible states.

Conversely, if T becomes very small, all the n_i will be negligibly small, except for n_0, which is of the order of N. Therefore, all the subsystems will be in the fundamental state of energy E_0 (or one of the fundamental states), which is a state of minimal energy. Therefore, the total energy is $E = NE_0$. The parameter T thus appears to be a control parameter for obtaining the fundamental state. When we want to drive the subsystems to their fundamental state, we have to decrease the temperature T of the heat bath, but this cooling must be controlled very carefully: if the temperature is lowered too rapidly, each subsystem will be driven to a metastable state, the energy of which is higher than the fundamental state (this method is used, for instance, for obtaining the transition from a liquid state to a glassy state). To reach the fundamental state, it is necessary to perform a gradual cooling: the temperature is lowered slowly, and each step of the temperature schedule lasts long enough, so that each subsystem can reach thermodynamic equilibrium at the given temperature.

Thus, we can control the energy of each subsystem and, consequently, the total energy E of the system, by means of the temperature T.

Conversely, we can control the temperature T of the heat bath by acting on its energy E. Let us imagine that we make some change to the distribution of the subsystems among states with energies E_i; assume that the new value of the total energy E is smaller than the previous one. Since the average energy of the subsystems is:

$$\langle E\rangle = E/N$$

a decrease in E will obviously lead to a decrease of $\langle E\rangle$, thereby driving the subsystems to states of lower energies; obviously, when the total energy becomes equal to NE_0, all subsystems are in the fundamental state.

Thus, decreasing the total energy of the system or decreasing the temperature of the heat bath have the same effect, namely to drive the subsystems to the fundamental state.

Analogy with electronic systems

Assume that we want to simulate the behaviour of the previous physical system with a hypothetical computer. It is important to notice that the structure of the system is parellel, and hierarchical: each subsystem is governed by its own physical laws; the N subsystems operate in parallel and they must obey relation (2), which expresses the constraint of the total energy E imposed by the heat bath:

$$\sum_{i=0}^{M} N_i E_i = E \tag{2}$$

This structure can be duplicated for a hypothetical electronic system which simulates the physical process: we consider N parallel processors, working under the supervision of a master processor. Each processor has the ability of choosing a state for a subsystem, and the resulting total energy of the heat bath is controlled by the master processor. We have seen that, by decreasing gradually the energy of the heat bath, we can force the subsystems to approach their fundamental state of energy. To simulate this process with our hypothetical electronic system, we can imagine the following procedure. In an initial step, each processor chooses a state of energy randomly and produces a token. We assume that we have a collection of M boxes, labelled in the following way: box number zero is intended to receive the tokens from the subsystems having the smallest possible energy E_0, box number one is intended to receive the tokens from the subsystems having the second best energy E_1, etc ... Once a processor has chosen a state, it dispatches its token to the appropriate box (Figure 2). The minimization proceeds as follows: the total energy of a given token distribution is given by:

$$E = \sum_{i=0}^{M} N_i \cdot E_i$$

where N_i is the number of tokens present in box number i. If the same process is iterated several times, the resulting distribution in the boxes is approximately uniform, and the total energy at each iteration will fluctuate slightly around a value:

$$E_\infty = N/M \sum_{i=0}^{M} E_i$$

We now proceed to the next step, in which the master processor allows each processor to make some change to its previous choice, subject to the condition that the new total energy E^1 be — allowing for some fluctuations — smaller than E_∞. The process may be iterated with a still smaller energy E^2, and so on until, hopefully, the minimum possible energy NE_0 is reached. We know that this procedure will generate successive Boltzmann distributions (with statistical fluctuations) corresponding to smaller and smaller values of the parameter T, until the optimal configurations are reached. For a given value of T, the average total energy will be:

$$E = N \sum_{i=0}^{M} E_i \exp(-E_i/k_B T) / \sum_{i=0}^{M} \exp(-E_i/k_B T)$$

and, when the optimum solution is reached:

$$E_{\text{opt}} = NE_0$$

In order to simulate the cooling process on a sequential computer, we have to generate a set of states satisfying a Boltzmann distribution. The Metropolis algorithm[11] is a computationally efficient way of doing it.

The Metropolis algorithm is a probabilistic algorithm, in which one generates a sequence of states which is a Markov chain, each state depending on the previous one by the following rule: suppose that we have a state k, with energy w_k; to obtain the configuration $k+1$, we use

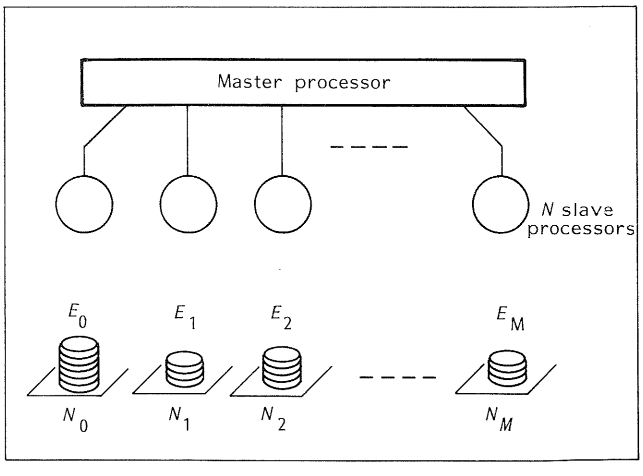


Figure 2. Hypothetical machine

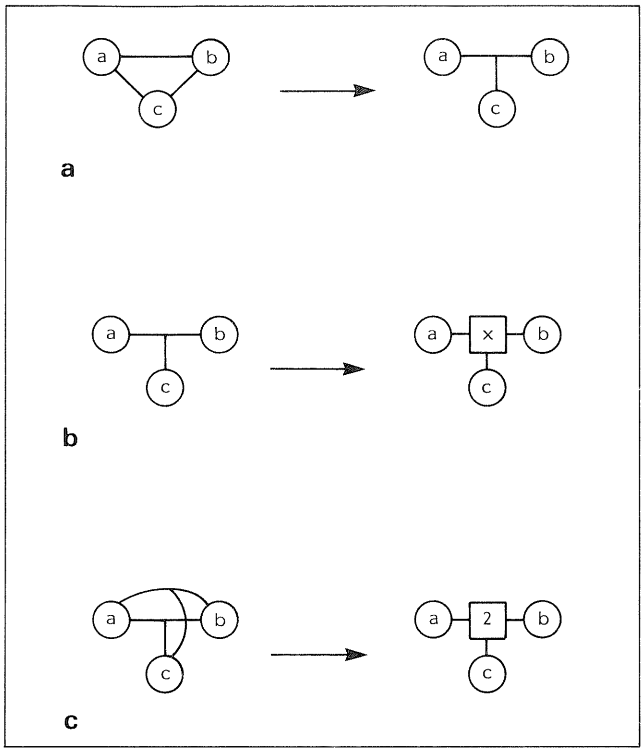


Figure 3. Logical net representation

the following procedure:

- make some change to state k; compute the new energy w_{k+1};
- if $w_{k+1} \leqslant w_k$, accept this new configuration: change k to $k+1$ and iterate to the first step;
- if $w_{k+1} > w_k$, compute: $A = \exp - [(w_{k+1} - w_k)/T]$, and generate a random number R between 0 and 1; if $R \leqslant A$, accept this new state: change k to $k+1$ and iterate to the first step; if $R > A$, reject this state: iterate to the first step.

For a given value of T, the average of the energy over the states generated by the above Markov chain is asymptotically equal to the average energy over a Boltzmann ensemble.

THE AUTOMATIC PLACEMENT

The problem which is presented here mainly concerns the automatic layout of integrated circuits, which is, in essence, identical to the placement of components on a printed board. We shall not present this second aspect in detail because it can be considered as a particular application of the methods which have been developed for integrated circuits.

Circuit design always requires a placement step. This step is performed in order to place all the components of the integrated circuit on the chip area. Actually, one of the most important problems in VLSI computer aided design, is to find optimal placement strategies, with fast and efficient algorithms.

Before presenting such a method, we must define the starting point of the layout process.

The logical network

Generally, the logical description of a circuit is given by a net-list in which each component is assigned to a set of signals. These signals define the connections between the components.

Two different types of logical networks can be used to describe the net-list. The first and easiest way consists in defining each component as a vertex of a graph and each signal as an edge. Such a representation is often used but cannot give all the required information. For example, in case of a multiterminal signal (i.e. a connection between three or more components) it is impossible to define a fully equivalent network. This type of network (Figure 3a) allows the description of two-terminal wires only.

In order to represent any type of components and of connections, a second model of network must be defined. Two node types are used. One is associated to components, and one is associated to signals. The previous example is then represented by Figure 3b.

Some information can be added to such a representation. A weight can be assigned to each vertex. This weight is used in order to define a priority to different signals. This priority is defined by the user, according to electrical or propagation time constraints.

A weight can also be added to the various nodes representing the connections. This is performed to reduce the network complexity. In this case, two distinct signals connecting the same components will be represented by only one node which is assigned a weight of two (Figure 3c).

The host structure

This is the internal structure of the chip on which all the components must be placed. Various kinds of structures can be defined (Figure 4):

- *Fixed structure.* In such a case, many constraints are defined on the host structure:
  - the whole chip area is defined,
  - the various locations to which the components must be assigned have fixed sizes and coordinates. In

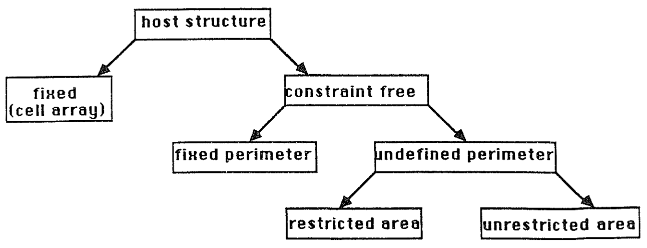


Figure 4. Hierarchical constraints

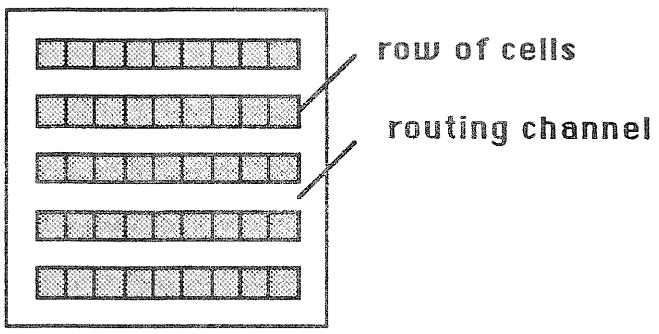


Figure 5. A gate array structure

practice, such a structure can be considered as an array in which each cell is dedicated to receive a component or a set of wires. Generally, the cells are divided into two clusters: the first one is defined to receive the components, and the second one, to receive the connections. Adjacent cells of second type define a 'routing channel'. This structure is used in gate arrays or cell based circuits (Figure 5).

- *Constraint free structure.* In this case, there are no dedicated locations on the chip area. Some restrictions can be applied to the whole chip area:
  - The chip perimeter can be defined. In such a case, all the components must be placed in a restricted area.
  - The whole area can be restricted. In this case, the chip area cannot exceed a maximum value, but the sizes (length and width) can be adjusted in order to satisfy other placement constraints. The length/width ratio (aspect ratio) is usually assigned an upper limit in order to avoid too large deformations.
  - If no constraint is defined, any placement is allowed, but the enclosing rectangle is desired to be as small as possible.

Note that, in all cases, the chip areas are restricted to rectangle-like areas, i.e. polygons with vertical and horizontal edges only. Such polygons can be described as a set of contiguous rectangles.

The components to be placed

Various types of components can be used according to the nature of the host structure (Figure 6). In the case of a fixed host structure, the components are adapted to the cell sizes. Therefore, a component is designed to be assigned to a single cell or to an integer number of contiguous cells. In the case of a constraint free area, various types of components may be found:

- *Hard components.* In this case, the whole component geometry is defined and cannot be changed.
- *Soft modules.* The component geometry is not yet defined. In most cases, the only available information is related to area, but sometimes the aspect ratio may be an additional constraint.
- *Set type.* In this particular case, a limited choice is given for the component geometry. A set of various hard modules is given for each component and the appropriate choice is performed during placement according to the relevant quality criteria.

The placement problem

The placement step consists in assigning locations and orientations to each component. To perform such an operation, various geometric transformations can be applied to the components. These transformations are the following:

- translation, which is the basic operation, and which allows to assign the coordinates to the various components;
- rotation, which is generally restricted to 90° steps;
- mirror symmetry, which is also restricted to horizontal and vertical axis;
- deformations, which can only be applied to soft components.

In fact, each component or component type can have a restricted set of allowed transformations. For example, a hard component cannot be modified by a deformation.

Finally, the placement consists in assigning a location (i.e. coordinates) to each component by applying the various legitimate geometric transformations.

The cost function

Any placement procedure aims at obtaining a 'good' result. In order to achieve this, a quality evaluation must be defined. The most common criteria used to improve the quality of a placement are the following:

- obeying all topological constraints:
  - no overlapping of the components,
  - no overflow of the components on the host structure.
- optimizing various parameters:
  - enclosing rectangle area,
  - total connection length.

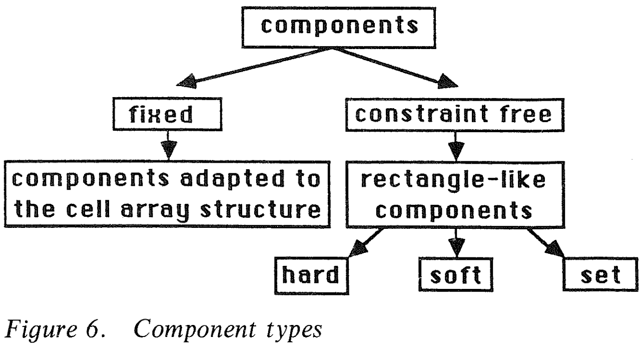

components
fixed
constraint free
components adapted to the cell array structure
rectangle-like components
hard
soft
set

Figure 6. Component types

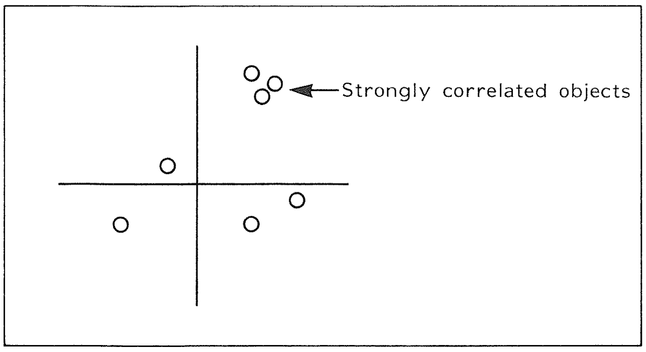


Figure 7. 2D representation of block centres

A cost function is then defined. It depends on all the quality parameters. For example: Let P be the overlapping area of the components, F the overflow area of the various components on the host structure, B the bounding rectangle area, and L the total connection length.

The cost function is given by:

$$F=\alpha P+\beta F+\gamma B+\delta L$$

The penalty parameters α, β, γ and δ are adjusted according to preferential criteria. We must note that a placement cannot be accepted if the first two constraints are not obeyed. That means $P=0$ and $F=0$. Therefore, the two parameters α and β of the cost function must be larger than the others.

The goal of automatic placement is to find a reasonably good solution within an acceptable computation time since the problem is NP-complete.

The problem initialization

Even if the simulated annealing method gives good results[9,12–16], it has been proved that a good starting point induces best results in a shorter time than a random one[17,18]. Thus, a fast and efficient initialization method must be used. Exploratory data analysis techniques yield a satisfactory solution to this problem.

Exploratory data analysis[19,20] is a set of statistical methods which allow to handle large amounts of items described by various criteria. In a first step, an Euclidean representation of the data, which preserves the similarities between the objects, is found; in a second step, this representation is embedded into a 2D space. Thus, the result is a 2D Euclidean representation of the data which is an image of the relations between the set of items which are analysed.

Two types of information have been input to our data analysis algorithms. The first is an objects X criteria matrix:

$$M=(m_{ij})$$

in which $m_{ij}=1$ if object i satisfies criterion j, i.e. if block i is connected to net j, $m_{ij}=0$ otherwise.

The second is an item-to-item dissimilarity matrix $D=(d_{ij})$, where d_{ij} has been defined as: $d_{ij}=1/c_{ij}$, where c_{ij} is the total number of nets shared by blocks i and j. If $c_{ij}=0$, d_{ij} takes an arbitrary large value.

Factor analysis yields the 2D representation of Figure 7.

At this step, each component is defined by a dot, so that centre coordinates only are given. Such a representation cannot give a final placement. Overlaps are not suppressed and spaces between the components are not optimized. However, such a result is a satisfactory starting point for an annealing process, because the relative block centre positions have been optimized according to the wire length criteria.

Application examples

To illustrate the steps of an automatic placement process, some examples are presented. The first example shows the initialization using data analysis for a typical ALU (Figure 8).

Figure 9 shows the map of the components obtained by data analysis in less than 2 CPU minutes on a VAX 780[21,22].

In the second example, a full placement process using data analysis for initialization and annealing for final improvement is described. The circuit is made of twelve hard components which must be placed on a constraint-free host structure. The connection matrix and a 2D representation given by data analysis are shown on Figure 10.

The decrease of the cost function during the annealing

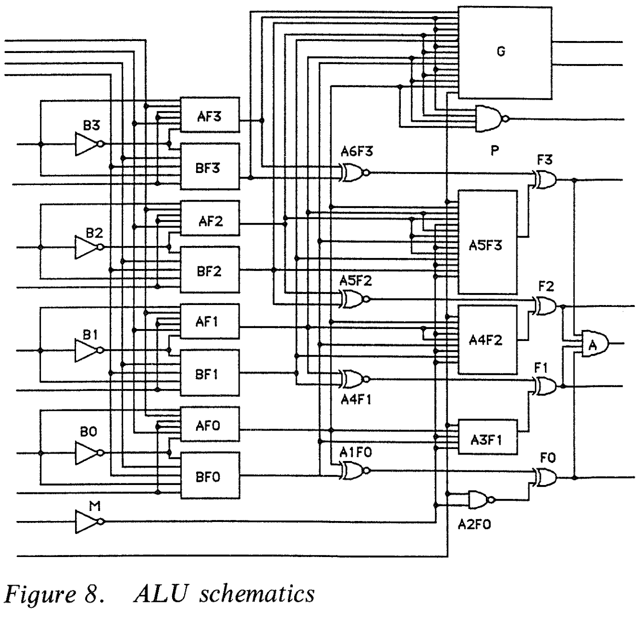


Figure 8. ALU schematics

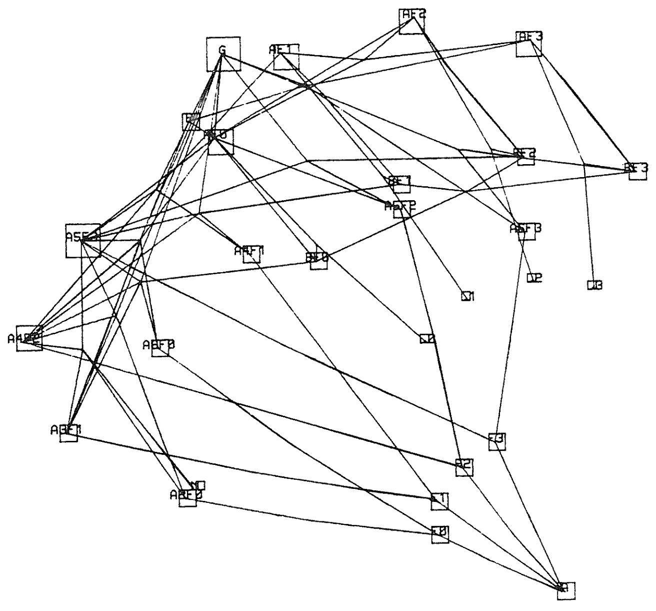
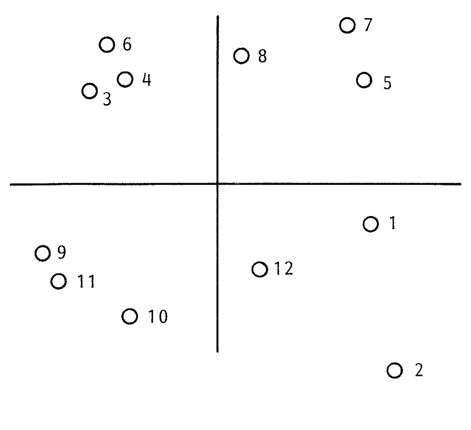


Figure 9. Representation obtained by data analysis

	1	2	3	4	5	6	7	8	9	10	11	12
1	-											
2	7	-										
3	3	0	-									
4	5	0	4	-								
5	14	0	5	5	-							
6	3	0	7	4	6	-						
7	18	0	9	9	31	11	-					
8	1	0	1	2	6	2	11	-				
9	3	0	3	5	3	2	6	3	-			
10	5	0	4	3	2	4	6	1	8	-		
11	3	0	4	3	3	4	7	2	9	9	-	
12	9	0	2	2	12	2	2	1	4	10	5	-

a



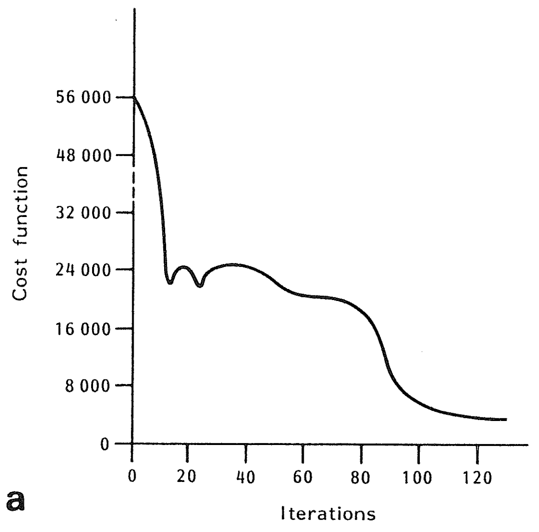
b

Figure 10. (a) Connection matrix and (b) block embedding after data analysis

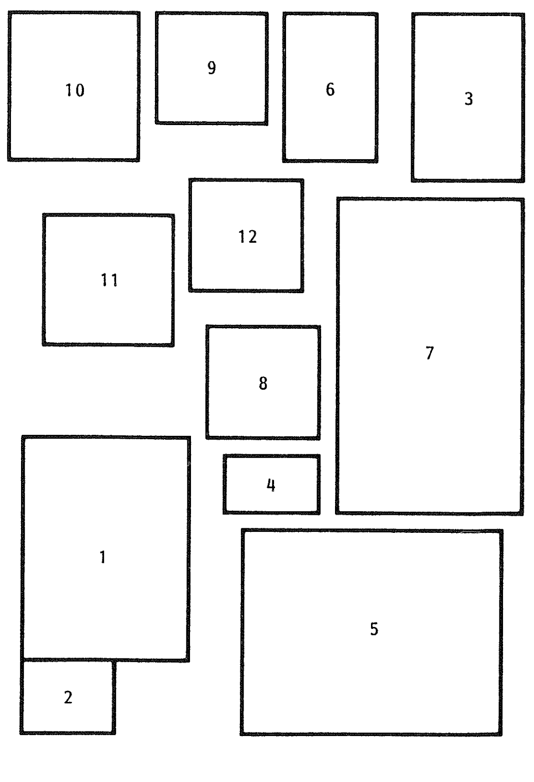
process is shown on Figure 11a. The cost function is given by:

$$F = L + \alpha P + \beta E$$

where L, P, E are respectively the total connection length, the overlapping area and the bounding rectangle area. Parameters α and β have been assigned values of 50 and 5 respectively. The initial temperature is 10 units. The



a



b

Figure 11. (a) The annealing process and (b) its final result temperature is lowered as follows:

$$T_{n+1} = 0.9 T_n$$

The average number of iterations at each temperature is about $100N$, where N is the number of blocks (modules). The number depends on the current temperature value and it is gradually increased as the temperature decreases. An iteration consists in at most three transformations (rotation, translation, deformation, mirror symmetry) selected randomly.

The whole process is performed in 3 CPU minutes on a 4 MIPS computer (see Figure 11b for the final result).

THE AUTOMATIC ROUTING

The routing problem

Routing an integrated circuit consists in determining the physical connections between blocks, once the placement has been achieved. Let us consider the case for which all the blocks have a rectangular shape and the connections cannot pass through the blocks; therefore, the pins are necessarily located on the periphery of the blocks. Under these conditions, the routing problem can be expressed in terms of a discrete geometry. Each of the n blocks of the circuit is defined by its dimensions and by the locations of its pins on the boundary. The whole set of the pins of the circuit is partitioned into m disjoint classes which define the *logical nets* to be realized, during the routing process, by wires, also called *physical nets*. The technological rules related to the manufacturing process of circuits and the design rules are taken into account through the choice of a routing grid which guarantees that the pins of the blocks coincide with the vertices of the grid, so that the layout rules can be expressed through it. In the case of a usual technology allowing two dedicated levels of interconnections, it is possible to find a grid step such that the routing constraints can be summarized by two simple rules:

- every physical net coincides with a tree which is a partial subgraph of the grid,
- every vertex and every edge of the grid cannot coincide with more than one physical net.

Of course, this simplification of the real problem must not lead to solutions with an unacceptable quality. Since the effective density of wires and the area that they occupy depend directly on the grid step, we may have to decompose the circuits into several grids with different steps; at that time, the routing process has to incorporate a step whereby the layouts are connected.

The routing probelm *per se* consists in associating to each logical net a physical net which obeys the layout rules and optimizes some global properties of the circuit layout, such as the total wire length or the number of vias for the wires. In the proposed model, the area reserved for routing corresponds to a partial subgraph of the grid. The physical nets are completely defined by a forest of disjoint trees going through the terminals of the corresponding logical nets; these are the so-called Steiner forests. The objective function associated to the optimization problem is defined through a valuation of the grid's edges. The problem is to find a minimum cost Steiner forest. Since the associated decision problem, namely the existence of a Steiner forest having a given cost, is an NP-complete problem, we use a strategy which decomposes the resolution process into four stages:

- *partitioning* the initial grid into subgrids related to a cutting of the block free area into rectangular cells called paving-blocks;
- *global routing* of nets; i.e. assignment of logical nets to the paving-blocks crossed by the associated physical nets;
- *ordering* of the paving-blocks for the effective layout of the nets;
- *full routing* of nets in each paving-block, consistent with the routing of neighbouring paving-blocks.

Such a strategy aims at decomposing the initial problem into subproblems of the same kind with as little interaction as possible on their common borders. It becomes really efficient only if the routing of the paving-blocks is much simpler, and if they can be ordered so as to avoid conflicts. The most interesting case arises when the paving-blocks are channels. A channel is a paving-block having all the pins needed for the nets on two opposite sides. This preferential direction has the effect of 'linearizing' the problem. Usually, nets are also allowed to go beyond the sides of the channel which are free of pins. One can show that this extension of the problem can be reduced to the initial case without any increase of the complexity of the problem.

The routing region of a circuit is usually partitioned into rectangular subregions by drawing lines from corners of blocks. One can exhibit an efficient draining strategy constructing rectangular subregions which are all channels for most circuits[23]. The structure of the circuits which are not decomposable into channels is such that a real circuit is very unlikely to have it.

The full routing of a channel

In the present section, we give an illustration of the simulated annealing strategy for the specific case of full routing of channels.

A channel is defined as a rectangular grid of height H and length L, composed of vertices $\{(i,j); 1 \leqslant i \leqslant H, 1 \leqslant j \leqslant L\}$. We assume that the sides of length L are the North and South sides and that they carry the terminals of the nets; the vertex $(1,1)$ corresponds to the North-West corner of the grid. We call respectively lines (or tracks), and columns the paths whose vertices have the same first (resp. second) component; in this model, we assume that the lines carrying terminals can be used as the other lines for the layout; one can see that adding a new line on the North and a new line on the South allows to recover the usual model.

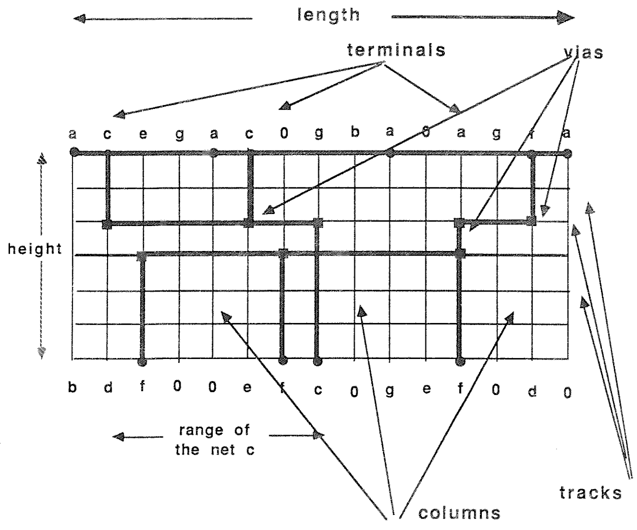


Figure 12. A two-layer channel. Each channel is labelled with the name of its net

There are two independent classes of constraints on the routing problem: the constraints which are inherent to the nets — i.e. independent from the layout rules — and the constraints which are induced by the properties of the grid (Figure 12).

Relational constraints between nets

The range of a net is defined as the segment starting on the index of the column carrying its leftmost terminal and finishing on the index of the column carrying its rightmost terminal.

Two nets overlap iff their ranges have a nonvoid intersection. The cliques of the graph associated to this relation, which is usually called the horizontal constraint graph, are used to evaluate the congestion of the channel.

Two nets with opposite terminals have a definite relative position in the neighbourhood of this column. These local conditions can be expressed in a sufficient condition which gives a global relative position of the nets. A net e is above a net e' iff there exists a column carrying a northern terminal of e and a southern terminal of e'. The existence of circuits in the graph associated to this relation, usually called the vertical constraint graph, points out conflicts for the layout which must be solved by cutting some nets into pieces.

Valuations of grid edges

Each grid edge is evaluated by a positive or null integer which represents its wiring capacity; the capacity of an edge is decreased by one each time it is used by a Steiner tree; thus, a null capacity indicates a forbidden edge and all the Steiner trees are automatically disjoint if the capacity of grid edges belongs to the set $\{0,1\}$. The relaxation of the disjunction constraint for the trees is obtained by allowing edges to have a capacity larger than one. In any strategy taking into account the constraints gradually, such as the simulated annealing strategy, the edge capacity is an increasing integer function of a parameter which is equivalent to a temperature, although it takes only discrete values; the temperature is supposed to decrease during the optimization process; we denote this function by $c(a, T)$ where a is a grid edge and T the temperature.

The quality of the layout is evaluated through three criteria:

- number of tracks necessary for the layout,
- characteristics of vias occurring in wires,
- length of wires.

The minimization of the number of tracks is the first goal to achieve. We know that the minimum number of tracks necessary for the layout of a set of nets has a lower bound given by the maximum number of nets having terminals in two subgrids defined by North–South cuts of the initial grid. One can check that this bound is reached within a few units in all the practical cases.

The number of vias necessary for the layout of a net is at least equal to the number of cells carrying terminals of the net. In most cases, the minimization of the total number of vias is not the only criterion which is used; we look also for a minimization of contiguous vias in a same column.

Each grid edge is also weighted by a cost representing the conductivity properties of the material used for the corresponding wire; we denote the value of this function for the edge a by the positive integer $p(a)$.

Thus, we get an objective function which is composed of two elements, the first one expressing the wireability and the second the quality of the routing. We denote this function by $F(t)$:

$$F(t)=\sum_{a\in\{\text{tree edges}\}} \alpha r(a,T)+\beta s(a)$$

If $n(a, t)$ is the number of trees going through grid edge a at temperature T, one can take:

$$r(a,T)=\text{if } n(a,T)<c(a,T) \text{ then } 0 \text{ else } (n(a,T)-c(a,T))$$

When we need to penalize the capacity overflows of vertical edges more heavily than horizontal ones, we can introduce a new weight. The function $s(a)$ is either equal to $p(a)$, or to a weighted function of $p(a)$ and of the number of vias associated to the edge a.

Divide and conquer strategy

The divide and conquer strategy for channel routing consists in reducing the layout of a family of n logical nets in a channel of height h to the routing of two families of n subnets in two channels, a North channel and a South channel, with respective heights h_N and h_S with $h=$

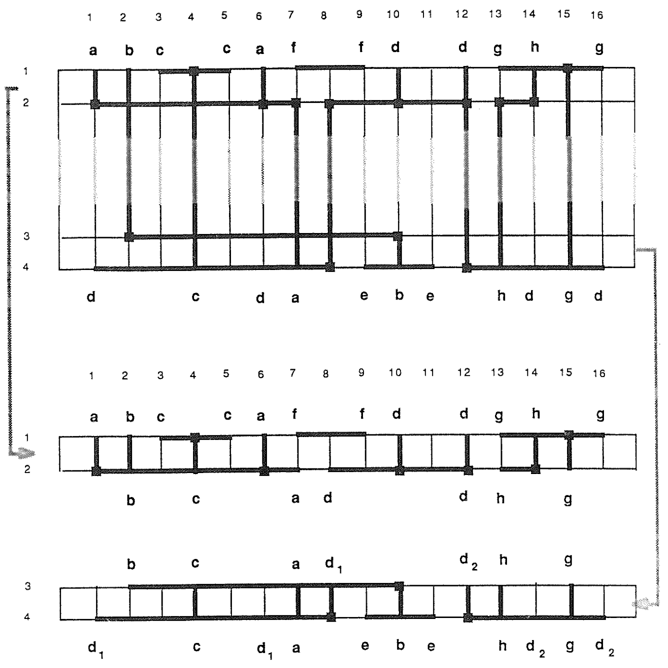


Figure 13. Subdivision of a channel routing into two subchannels routing

$h_N + h_S$, the southern terminals of the North channel being identical to the northern terminals of the South chennel. It is important to notice that some nets can share a terminal when the column capacity is larger that 1.

Such a strategy is safe if optimal solutions are still reachable; thus, one can easily check that every optimal layout can be decomposed into two optimal layouts for subchannels North and South as they are defined. Moreover, it has the advantage of reducing the general problem to the case of grids of height 2 with edges of capacity greater than 1 and to require a synthesis stage which is limited to the merging of layouts obtained in previous stages (Figure 13).

Linear algorithm for routing of channels of height 2

The trees of a partial subgraph of a grid G of height 2 can be constructed gradually by a West–East exploration of the grid without backtracking. We denote the subgrid consisting of the m first columns of the grid G by G_m; every tree of G_m extendable to a tree of G_{m+1} contains at least one of the vertices $(1,m)$ or $(2,m)$. The extendable trees of G_m are partitioned into four classes:

- those having only $(1,m)$ as a vertex,
- those having only $(2,m)$ as a vertex,
- those having both $(1,m)$ and $(2,m)$ as vertices, without the edge joining them,
- those having the edge $[(1,m), (2,m)]$.

Every tree of G passing through a given set S of grid vertices, i.e. every Steiner tree of G for S, is also constructible by this process. If the grid edges are weighted by a cost function, one can obtain optimal Steiner trees for a set S of vertices by constructing a sequence of optimal extendable trees for each of the four classes, following the dynamic programming principle; if L is the length of the channel, this construction can be done in a time proportional to $4L$. The algorithm, which is an extension of the results of Aho, Garey, Hwang[24] and of the Burstein-Pelavin algorithm[25–27], is described by the automaton represented on Figure 14.

This strategy does not remain polynomial when one tries to construct n optimal Steiner trees for n families of vertices in a grid with edges labelled at the same time by a capacity and a cost. Thus, a family of 4^n trees has to be constructed gradually. One can verify that this problem has solutions only under some conditions (cf. relational constraints for nets) and that it is, in fact, an NP-complete problem.

Controlled relaxation strategies and simulated annealing

At the moment, the routing of a family of n nets in a channel of height 2 is only achieved by general algorithmic

Name	Operation
a	Insertion of the edge $[(1,m)\ (1,m+1)]$
b	Insertion of the edge $[(2,m)\ (2,m+1)]$
c	Insertion of the edges $[(1,m)\ (1,m+1)]$ and $[(2,m)\ (2,m+1)]$
d	Insertion of the edges $[(1,m)\ (1,m+1)]$ and $[(1,m+1)\ (2,m+1)]$
e	Insertion of the edges $[(2,m)\ (2,m+1)]$ and $[(1,m+1)\ (2,m+1)]$
f	Deletion of the edges $[(1,m)\ (2,m)]$ and Insertion of the edges $[(1,m)\ (1,m+1)]$, $[(2,m)\ (2,m+1)]$ and $[(1,m+1)\ (2,m+1)]$

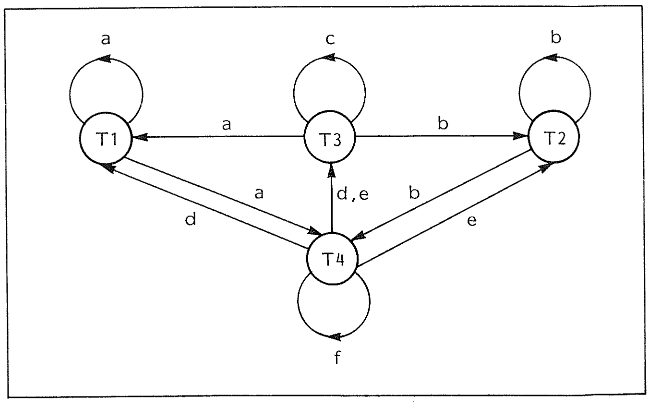


Figure 14. The automation for routing chennels of height 2

strategies used to solve any combinatorial optimization problems. The class of such problems can be defined by the following sketch plan: given a finite set of configurations and an objective function which defines a cost for each configuration, find a configuration with the lowest cost among the subset of allowed configurations. The allowed configurations are those which represent solutions satisfying the constraints of the problem to solve.

An optimization strategy can be described as an exploration process of the set of configurations. Such a strategy partitions the set of configurations into three classes: initial, intermediate and final configurations. Any implementation of it constructs a path from an initial configuration to a final configuration. It can be assumed that each final configuration can be reached from at least one initial configuration.

A strategy is *optimal* iff the final configurations are exactly the allowed configurations with the lowest cost. A strategy is *safe* iff all the subset of final configurations contains only allowed configurations and at least one such configuration with the lowest cost.

Since for such problems there is no polynomial time optimal strategy, one has to try to get polynomial time safe strategies. Unfortunately, the two most usual strategies for channel routing are unsafe.

As a matter of fact, the strategy which routes each net independently of the others by using the same capacity for the grid edges, constructs a solution as soon as each net is routable, but the solution is very likely to violate some constraints. In the same way, the strategy which routes nets in sequence with updating of the edge capacities, provides a solution which satisfies the constraints, but it may happen that the routing of the last nets is impossible because of new constraints arising from the routing of the first ones. Thus, these two strategies are unsafe.

One way to get closer to a safe strategy is to accept that, under certain conditions, the exploration process visits configurations which do not satisfy the constraints of the problem. Simulated annealing is one of these *controlled relaxation strategies*.

First, their efficiency lies in the possibility of defining a topology on the set of all configurations for which there exists an efficient algorithm to enumerate the configurations forming the neighbourhood of a given configuration. Usually, an operative definition is set in terms of elementary transformations which define the path from one configuration to another. An especially favourable case, from the algorithmic point of view, arises when there exists a coding of the configurations such that all the elementary transforms are implemented through a local modification of the code.

Secondly, the transforms satisfy some conditions to get a safe strategy. A set T of transforms is safe iff it is generated by a minimal family of elementary transforms acting in a reversible way (if there exists an elementary transform acting on a configuration c to get a configuration c', there exists also an elementary transform acting on c' to get back c) and transitively on the configurations (every configuration is accessible from any configuration through a finite sequence of elementary transforms).

As the topology introduced to define the resolution strategy is not necessarily intrinsic to the problem, artefacts of the topological landscape which are able to obstruct the exploration process must be removed.

Several authors have proposed to carry out the global wiring by simulated annealing[28,29]. In this paper, a relaxation strategy for detailed routing based on simulated annealing is proposed. The nets are wired all together by an annealing strategy the objective function of which is designed to take gradually into account the capacity constraints. An objective function, such as the function mentioned previously in the section on edge valuations, is perfectly suitable. In the case of the problem studied in this paper, one can state that the configurations can be coded in a local language[30] whose structure is directly deducible from the linear algorithm used to construct a Steiner tree in a channel of height 2. A safe family of local transformations related with the finite automaton recognizing this language[30,31] has been investigated to define the topology.

Experimentation

A test program has been implemented in the French dialect of Lisp, Le_Lisp and its companion object language Ceyx to validate the approach. The experiment was focused on the feasibility of an annealing process in a recursive context such as the context of the Burstein–Pelavin algorithm. The program follows the most usual cooling schedule, i.e. a constant decrease rate for the temperature and a decrease of temperature when a given constant number of unsuccessful trials has been performed to decrease the cost of the current configuration. Two classes of starting configurations have been investigated: configurations which are trivial to construct, like U-shape trees, and configurations constructed by an algorithm of the Burstein–Pelavin type.

Figure 15 presents a set of 63 nets routed by the algorithm in a channel of height 32 and length 174. In the present stage of the experiment[32], wireability can be reasonably guaranteed; however, it is necessary to improve the algorithm in two directions. First, the exploration of the configurations induced by the transformations at a given temperature leads to long loops which must be cut down to speed up the process. Secondly, the transformations defined through the actual coding of the Steiner trees are not always consistent with the natural routing heuristics to guarantee the suitable quality of the routing. Therefore, other regular or local languages must be considered to code the configurations.

CONCLUSION

In this paper, we have shown that simulated annealing has turned out to be an efficient tool for optimizing the

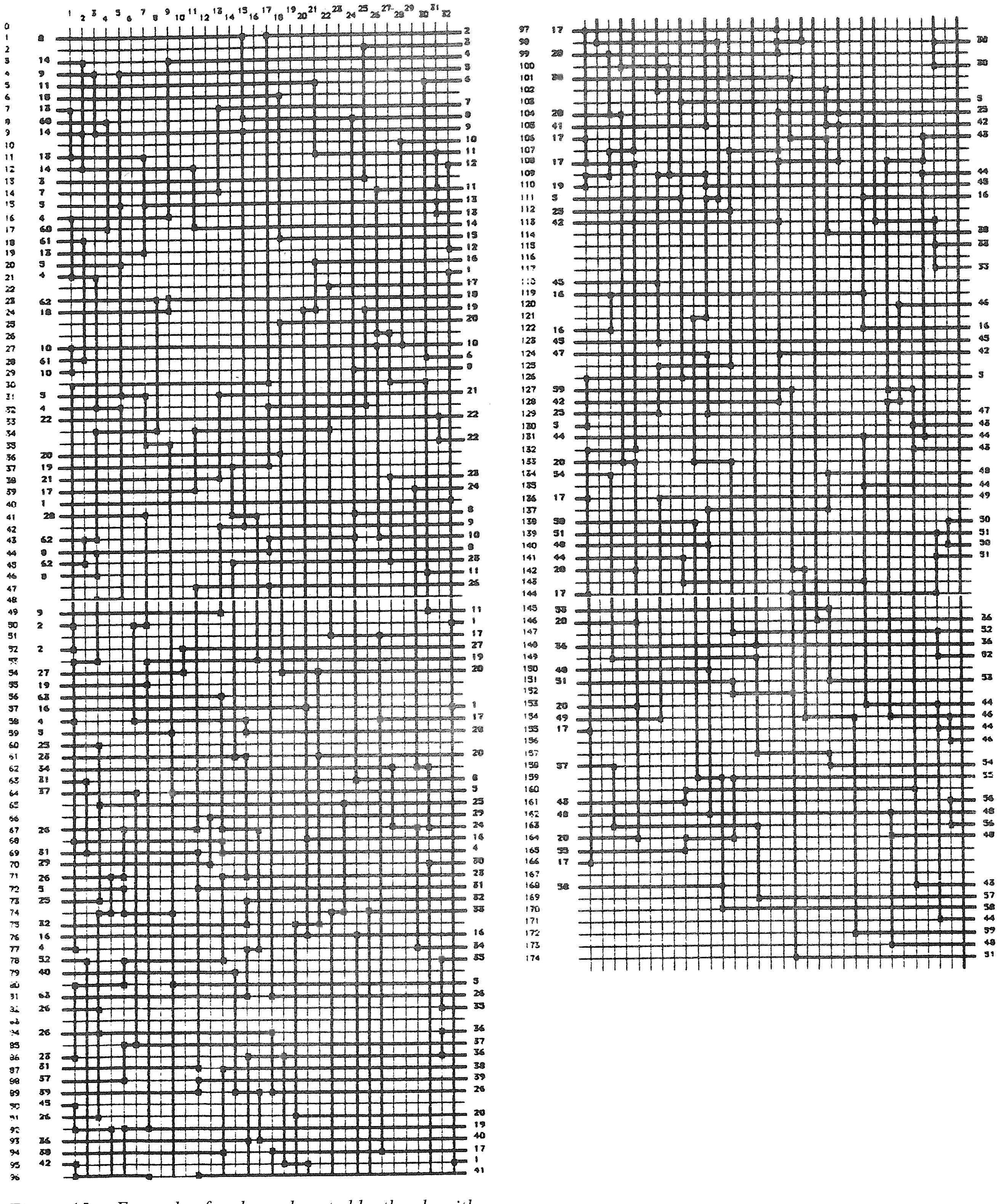


Figure 15. Example of a channel routed by the algorithm

layout of electronic circuits. The algorithm can be easily adapted to the special cases of the automatic placement and of the full routing of channels. From a practical standpoint, developments may be expected in several directions: first, there is a clear need for problem-independent results on the convergence of the method and the annealing schedule. Such results may be obtained from mathematical considerations about Markov chains as well as from progress in understanding of spin glasses. Secondly, a large amount of computation time could be saved by using the method interactively. Finally, the parallelization of the algorithm may be another possibility for complex problems which require a large computing power. In the present state of the art, however, simulated annealing is more and more routinely used and is already a part of the designer's toolbox.

REFERENCES

1 **Kirkpatrick, S, Gelatt Jr, C D and Vecchi, M P** 'Optimization by simulated annealing' *Science* Vol 220 No 4598 (1983) pp 671

2 **Cerny, V** 'Thermodynamic approach to the travelling salesman problem: an efficient simulation algorithm' *J. Optimization Theory and Applications* Vol 45 No 1 (1985) pp 41

3 **Siarry, P and Dreyfus, G** 'An application of physical methods to the computer aided design of electronic circuits' *J. Physique Lett.* Vol 45 (1984) pp 39

4 **Siarry, P** 'La méthode du recuit simulé: application à la conception de circuits électroniques' thèse, Univ. P. et M. Curie, Paris (1986)

5 **Toulouse, G** 'Applications de la physique statistique aux problèmes complexes' unpublished (1985)

6 **Guyon, I, Personnaz, L, Siarry, P and Dreyfus, G** 'Engineering applications of spin-glass concepts' Heidelberg Colloquium on Glassy Dynamics and Optimization, Springer (1986)

7 **Wille, L T** 'Optimization by simulated annealing — an overview and some case studies' to be published (1986)

8 **Aarts, E H L and Van Laarhoven, P J M** 'Simulated annealing: a pedestrian review of the theory and some applications' to be published (1986)

9 **Siarry, P, Bergonzi, L and Dreyfus, G** 'Thermodynamic optimization of block placement' *IEEE Trans. on CAD* Vol CAD 6 (1987) pp 211

10 **Toda, M, Kubo, R and Saitô, N** *Statistical Physics I* Springer Series in Solid-State Sciences, Vol 30, Springer, Berlin (1983)

11 **Metropolis, N, Rosenbluth, A W, Rosenbluth, M N, Teller, A H and Teller, E** 'Equation of state calculations by fast computing machines' *J. Chem. Phys.* Vol 21 (1953) pp 1087

12 **Jepsen, D W and Gelatt Jr, C D** 'Macroplacement by Monte-Carlo annealing' *IEEE Proc. Int. Conf. on Computer Design* Port Chester (1983)

13 **Otten, R H J M and Van Ginneken, L P P P** 'Floorplan design using simulated annealing' *Proc. Int. Conf. on CAD* Santa Clara (1984) p 96

14 **Siarry, P, Bergonzi, L and Dreyfus, G** 'Optimisation de placement de blocs par la méthode thermodynamique' 1er Colloque National Conception de circuits à la demande sur réseaux prédiffusés et précaractérisés, Grenoble (1985)

15 **Ouldali, B, Siarry, P, Dreyfus, G, Fauvin, I and Darnal, M** 'Application d'une nouvelle méthode d'optimisation à l'implantation automatisée des circuits hybrides' Conférence internationale sur le montage en surface des composants électroniques, Paris (November 1986)

16 **Wong, D F and Liu, C L** 'A new algorithm for floorplan design' *Proc. IEEE/ACM 23rd DAC* (1986)

17 **Barra, J R, Becker, M, Kouka, E F M and Tricot, M** 'An application of data analysis methods and simulated annealing for automatic layout of circuits' *Comput. Systems Sci. Eng.* Vol 2 No 1 (1987) pp 3–15

18 **Grover, L K** 'Standard cell placement using simulated sintering' *Proc. IEEE/ACM 24th DAC* (1987)

19 **Kouka, E F M and Saucier, G** 'An application of exploratory data analysis techniques to floorplan design' *Proc. IEEE/ACM 24th DAC* (1987)

20 **Benzécri, J P** 'L'Analyse des Données' Dunod (1984)

21 **Chaisemartin, P** 'FADDA2, a new interactive floorplan design system' *Proc. Integrated Circuit Technology Conference* Limerick (1986) pp 361–372

22 **Chaisemartin, P** 'Contribution à la génération automatique de plan de masse' PhD INPG, Grenoble (1986)

23 **Jalbert, D** 'ROUT2D: un routeur à deux couches d'interconnexion pour circuits VLSI' 1er Colloque National Conception de circuits à la demande sur réseaux prédiffusés et précaractérisés, Grenoble (1985)

24 **Aho, A V, Garey, M R and Hwang, F K** 'Rectilinear Steiner trees: efficient special-case algorithms' *Networks* Vol 7 (1977) pp 37–58

25 **Burstein, M and Pelavin, R** 'Hierarchical channel router' *Proc. DAC* (1983)

26 **Burstein, M and Pelavin, R** 'Hierarchical channel router' *IEEE Trans. CAD* Vol 2 No 4 (1983) pp 223–230

27 **Burstein, M and Hong, S J** 'Hierarchical VLSI layout: simultaneous placement and wiring of gate arrays' *VLSI 83* (1983) pp 45–60

28 **Leong, H W, Wong, D F and Liu, C L** 'A simulated annealing channel router' *Proc. Int. Conf. CAD* Santa Clara (1985) pp 226

29 **Vecchi, M P and Kirkpatrick, S** 'Global wiring by simulated annealing' *IEEE Trans. CAD* Vol CAD 2 (1983) pp 215

30 **Loubières, P** 'Un algorithme de routage canalisé' 1er Colloque National Conception de circuits à la demande sur réseaux prédiffusés et précaractérisés, Grenoble (1985)

31 **Loubières, P** thèse de 3ème cycle, to appear

32 **Oysel, P** 'Application de la méthode du recuit à un problème de routage canalisé' Mémoire DEA (1986)