BLUE MACAW: A DIDACTIC PLACEMENT TOOL USING SIMULATED ANNEALING

Renato Hentschke, Marcelo Johann, Ricardo Reis
Universidade Federal do Rio Grande do Sul (UFRGS) - Instituto de Informática
CP15064, CEP91501-970, Porto Alegre, Brazil email: {renato,johann, reis}@inf.ufrgs.br

Abstract: This paper presents a didactic Simulated Annealing placement tool. Simulated Annealing is a very famous generic combinatorial optimization algorithm that has been successfully applied to VLSI placement. Our tool allows any user to change several parameters of the algorithm and see their implications in the results graphically. At every algorithm step, the current placement disposition is displayed along with statistics related to the circuit connections. A plot of wirelength variation progress is also shown for visualization of the whole optimization process, and the user may play with different temperature schedules, perturbations and cost functions to see how this progress is affected by those parameters. The didactic tool can be used for teaching of both Simulated Annealing and Placement disciplines, as it significantly contributes to the understanding of the cell placement process and the Simulated Annealing method.

Keywords: Simulated Annealing, Placement, Learning, Microelectronics Education

1. INTRODUCTION

The learning process nowadays can be supported and improved by the use of multimedia instruction material, specially including the use of software that runs in real time during a class/course. The possibilities that the use of computers bring in helping the understanding of physical effects, chemistry reactions and technology processes are enormous. As a simple example, graphical simulations showing how a transistor works give a much better understanding than the use of printed book where the figures are static

and cannot represent well the component behavior. The greatest challenge is the amount of work needed to implement each tool to simulate a given process. The overhead is significant due to the fact that a didactic tool has its own requirements, although it has also some simplifications. For example, if a textbook on IC Design should have all figures substituted by tools that simulate the functioning and/or allow the student to interact by changing parameters, this would require many hours of dedicated programming to implement them. On the other hand, learning algorithms can also be easier if there are at least some representative tools to help. This paper presents one example of a didactic tool that helps understanding how an important algorithm works and it is used in the automation process of VLSI circuit design.

The algorithm explored is the Simulated Annealing [1], which is a well known optimization technique that has been applied in practice for many NP-complete and NP-hard problems. In microelectronics CAD, it has been successfully applied to many design problems, as Standard Cell Placement [2][3], Routing [2], Partitioning [3], Floorplanning [4] and many others. The algorithm is known to achieve very high quality of results and for being able to easily handle arbitrary constraints, as it is based on a generic cost function. Two known limitations of the technique are the huge amount of CPU runtime needed to handle large problems and the need for tuning. There are many good techniques to accelerate the algorithm, but none seem to solve the problem of excessive CPU runtime. Regarding the tuning part, there are many parameters that need to be appropriately set to achieve the best results on a specific application, and our tool helps the students to develop their comprehension of the algorithm behavior and its tuning.

As an analogy to the natural annealing process, Simulated Annealing begins with a high disturbance of the elements. In the natural process the elements are molecules. It happens exactly the same way in the Simulated Annealing algorithm. In high temperatures the algorithm explores the search space in a wide way, ignoring some degradation in the results and allowing to escape from a local minima. As the temperature gets lower, the current solution for the optimization problem starts improving slowly, as the search in the solution space becomes narrow. The whole process of starting in high temperatures and finishing in low temperatures (with a slow cooling procedure) combines efficient optimization with a hill-climbing capability, needed to escape from local minima. This process is very easy to understand if visualized graphically.

In this work we are specifically interested in using Simulated Annealing algorithm to solve the standard cell placement problem, although there are other algorithms that are well suited to the problem as well. The placement problem is well known and studied, and there are many academic and

commercial tools based on Simulated Annealing that are successful in solving this problem, as [2-7].

Our proposal is to aid the learning of the Simulated Annealing algorithm as well as the Standard Cell placement problem. A tool named Blue Macaw was then implemented with a simple interface and a graphical output of the current placement state. Parameters such as initial temperature, temperature variation, number of iterations, number of repetitions, perturbation functions and cost function are instantly available and can be played with by the user. By using the tool, students are able to check the effect of different parameters, understand the concepts of high and low temperatures, local minima, how these factors influence each other and how they can be tuned for an optimized placement.

2. THE SIMULATED ANNEALING ALGORITHM

The algorithm starts from an initial solution, usually generated by a random procedure or some constructive algorithm like the ones in [8]. Simulated Annealing will iterate over this solution with random or pseudo-random modifications. At each iteration, it can accept or reject the modification. At the end of the process, the current solution will be a near optimal solution given a sufficient number of iterations.

The Simulated Annealing algorithm needs basically four functions: **cost, perturbation, temperature schedule** and **acceptance**. The **acceptance** function is based on Markov Chains theory. The **cost** function determines which solutions are considered as good and which are not. In placement, it can be based on wirelength, congestion, timing, power, area or any combination of these variables. The **perturbation** function determines the neighborhood of a solution and which kind of search will be performed by the algorithm. Random perturbations allow the search to reach the whole search space. Greedy perturbations increase convergence speed, but may get stuck in local minima. The user can mix them [7] in the user interface. The idea of using clever perturbation is also explored in [5]. Finally, the **temperature schedule** function controls the temperature initialization and variation. In the Simulated Annealing meta-heuristic, the temperature is an external parameter that controls the behavior of the **acceptance** function. When the temperature is very high, the acceptance will be relaxed and the algorithm will mix the cells randomly. Each time the temperature is decreased the solution starts to improve a little. When temperature is minimum (zero), the circuit is placed. The initial temperature determination method is based on [6], that finds a temperature for a given initial acceptance ratio. The initial acceptance ratio (probability) may be high (near 0.8)

pointing to a **high annealing method**, or low (near 0.01) pointing to a **low annealing** method. The low annealing will keep the structure from the initial placement, which might be good or bad. To find a good probability for **low annealing** is an interesting didactic activity.

3. THE BLUE MACAW TOOL

The Blue Macaw is a didactic tool developed for students and beginners in CAD development or integrated circuit design. First, we provide a graphical output showing the current placement stage. Figure 1 shows a picture of the whole interface of the tool. Figure 2 shows an example of an initial placement and the respective optimized placement found after an annealing process. Observe that the cells are represented by a filled rectangle and the connections are represented by a straight segment connecting the cells. Another output of the tool is the cost variation graphical view of the whole annealing process (figure 3). To identify the impact of the temperature schedule in the cost variation is an interesting and didactic exercise.

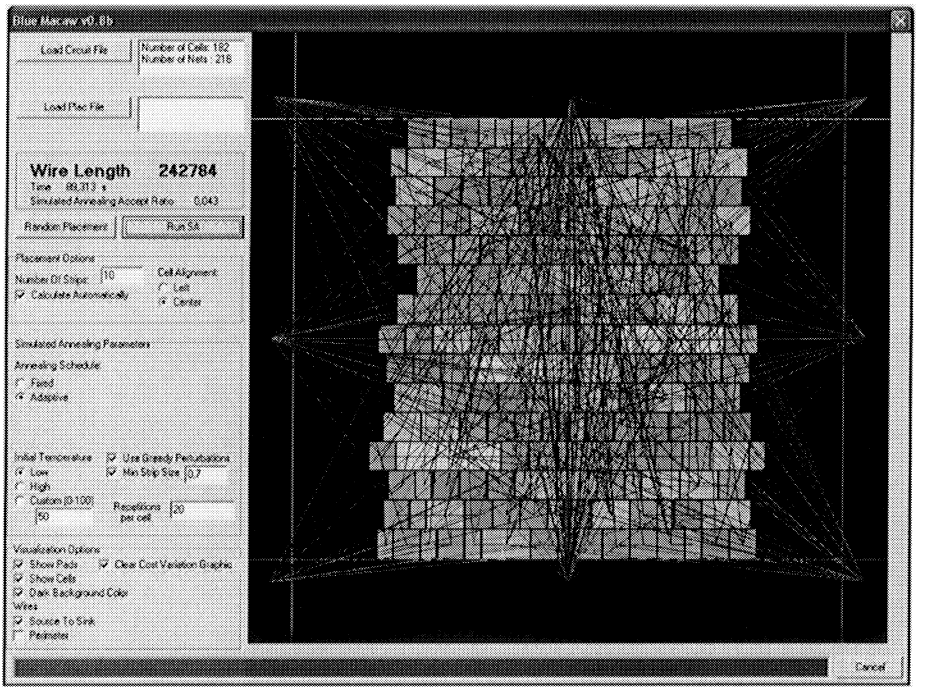

Figure 1 – Blue Macaw Tool interface

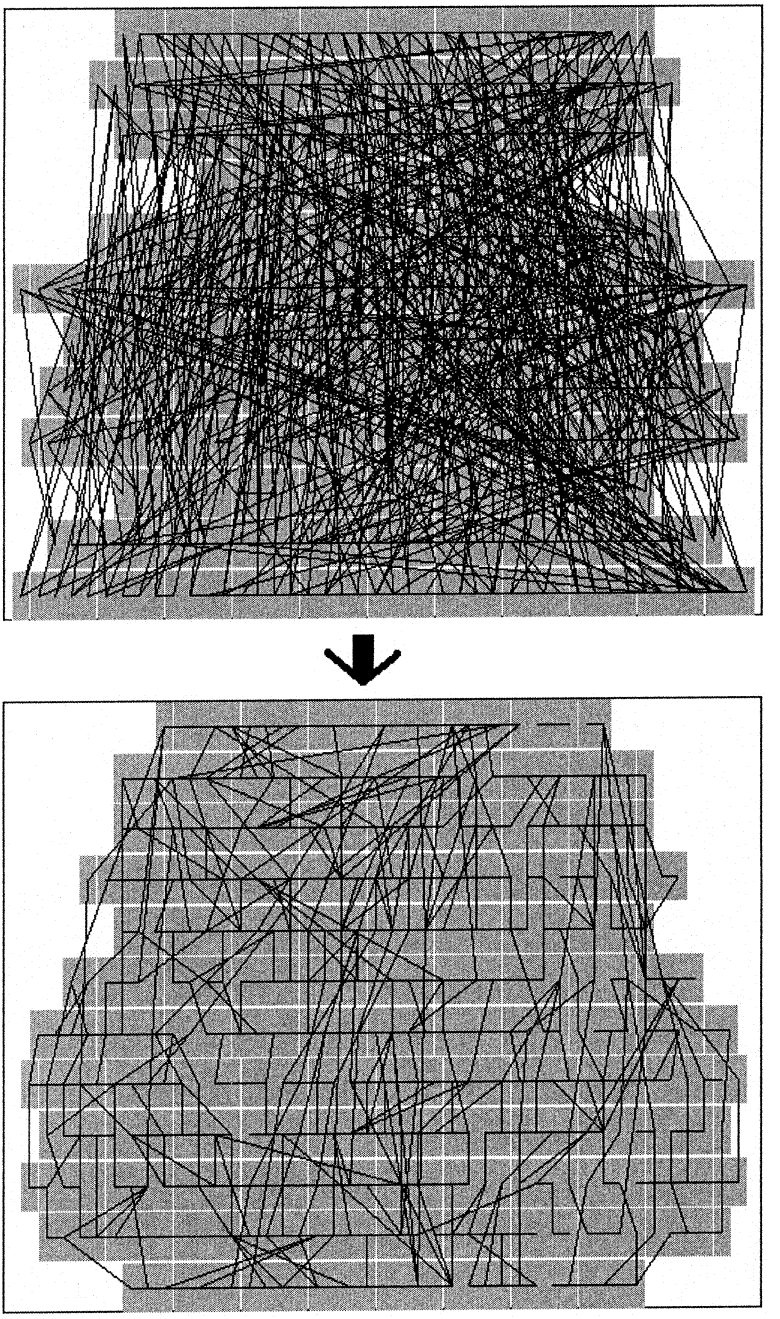

Figure 2 – Placement Optimization

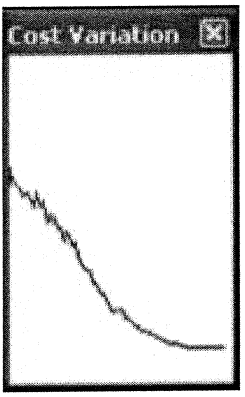

Figure 3 – Cost Variation Curve of a Simulation

As input, our tool receives a logic description of a circuit. We set up a web site, where users are able to download some circuits as examples. The circuits are described in a pseudo **spice** format, without the details of the cell's internal components. Blue Macaw will estimate the cell size based on their transistor counts.

Also, as output, the tool provides detailed online information of the annealing process. The tool interface shows a drawing of the temperature schedule. The interface also shows:

- **Total wirelength**: the total wirelength of the circuit estimated by the half-perimeter of the net's bounding-boxes
- **Maximum congestion**: the maximum wire congestion focus of the circuit, estimated with bounding-boxes.
- **Rejected Moves**: the number of random perturbations discarded by the algorithm. The greater this number is, more CPU time is needed to undo the perturbations that are not accepted.
- **Mean Probability of Acceptance**: This number indicates the mean acceptance ratio computed at each moment. The user can see that it is directly related to the current temperature.
- **Longest Connection**, **Mean Connection** and **Standard Deviation**: all estimated with half-perimeter. It is not included in the cost function.

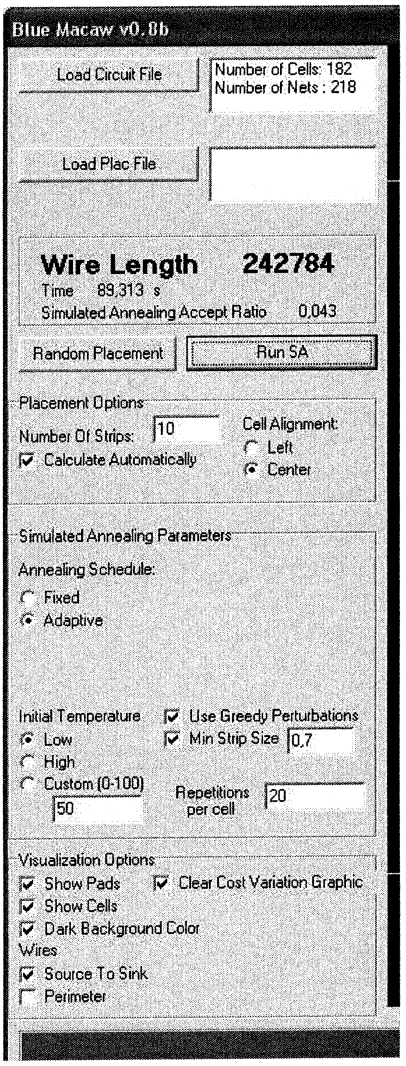

Figure 4 – Detail of the Tool User Interface

4. TUNING THE TOOL

The user may change some parameters of the annealing process, using the tool interface (figure 4):

- **Number of iterations**: the number of variations in the temperature. At the end of each iteration the temperature is decreased. Each iteration is composed of many repetitions.
- **Number of repetitions**: the number of steps that the algorithm will execute at each temperature.
- The **initial temperature** determines the starting point of the optimization process. If it is set as **very high**, the initial placement will be completely destroyed (considering that it might be good at the beginning), as many wide moves in the solution space will be made. If it is set as **low**, the initial placement may be some how preserved. We call these processes as **high annealing** and **low annealing** respectively**.** The effects of high and low annealing can be seen in the graphical output as well as in the cost variation plot. The initial temperature is computed in two ways: **Initial Probability** (selection of a desired probability of acceptance - 0.01 or less for low annealing and 0.6-0.8 for high annealing) and the program automatically computes the temperature (as in [6]). If the user prefers, she/he can set the initial temperature manually at the specific box in the graphical interface.
- The student can also control the **temperature variation** (annealing schedule). There are two basic algorithms that control the temperature variation: **fixed** and **adaptive**. The adaptive schedule is fully automatic. The idea is to decrease the temperature according to the acceptance ratio of the algorithm in the current temperature. The fixed schedule will follow a curve (or a straight line), as show in figure 5. By experiments, the user will notice that the variation (a) is appropriate for high annealing, while the others are better for low annealing processes. The student using the tool can set the curve graphically by adjusting the inflexion point by dragging it with the mouse.
- The perturbation function can be a combination of the following functions:
    - **Single Move** (a random move of one cell);
    - **Double Exchange** (an exchange of positions of two randomly selected cells);
    - **Force Directed Methods** (greedy variations of the two functions above, which may increase the convergence of the algorithm) [7]. The force directed method does not implement random perturbations, in fact, but moves each sorted cell to the point of the resulting forces for its

connections. Blue Macaw allows the use of the four techniques together, for faster convergence.

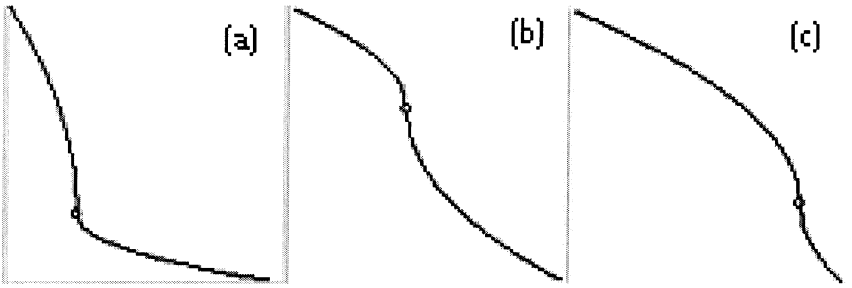

Figure 5 – Temperature Variations

4.1 Cost Function Options

The cost function plays a very important role in the optimization process. Its use is to qualify the current placement and define what is an optimized solution. Also, the cost function should be simple, to ease the optimization process. If it is complex and dependent on many variables, Simulated Annealing will have a hard time to find an optimized state. Additionally, the cost function is usually the performance bottleneck of the whole process, and should be carefully designed to save CPU time as much as possible.

In placement, the most common cost function is wire length. In the Blue Macaw tool, it is computed by the sum of the half-perimeter of the nets. However, we offer additional features to improve the placement quality:

- **Equalize Band Size:** the algorithm will try to maintain the rows with the same size. The user can use just the pure variance, which improves the equality, but may produce problems with wirelength. To avoid this, click on SqRt.
- **Min Strip width:** Selecting this option will limit the iterative placement to conform to a minimum strip width.

5. CONCLUSIONS

This paper has presented a didactic Simulated Annealing VLSI Placement tool that allows the user to try and to modify several parameters of the annealing process. The tool displays graphically the current placement, a variety of statistics about the process, including temperature schedule and

cost variation, along the placement. With the Blue Macaw tool, a student may learn easily the basics of the Simulated Annealing algorithm and its effects when applied for cell placement, while developing its ability to tune the method for a particular application.

The development of similar tools can help the understanding of other algorithms. The main difficulty is that a large taskforce is needed to implement a whole set of tools to teach a representative set of algorithms used in CAD tools and in computer science in general. However, we strongly believe that this is a path to take, as the human knowledge advances and the set of techniques that a student need to be aware of in a short period of time is ever increasing. Only graphical help and personal experiences with real tools can leverage the efficiency of the learning process. Another factor that is worth to consider is that any class or course that uses computer tools displaying functioning methods will let the students much more interested and involved, as compared to traditional speaks and teaching methods. This is not a novelty, but can be well explored by the current availability of hardware and software resources.

The BlueMacaw didactic placement tool can be downloaded freely from www.inf.ufrgs.br/~renato/bluemacaw.

REFERENCES

[1] S. Kirkpatrick, C. D. Gelatt Jr, M. P. Vecchi, **Optimization by Simulated Annealing**, In: *Science*, 220, 4598, 671-680, 1983.

[2] C. Sechen, A. Sangiovanni-Vincentelli. **The Timberwolf Placement and Routing Package**. In: IEEE J. Solid-State Circuits, vol. SSC-20, Apr. 1985.

[3] T. Taghavi, X. Yang, B. Choi. **Dragon2005: Large-Scale Mixed-size Placement tool.** In: Proceedings of International Symposium on Physical Design, ISPD, 2005.

[4] J. Cong, J. Wei, Y. Zhang. **A Thermal-driven Floorplanning Algorithm for 3D ICs.** In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design, ICCAD, 2004.

[5] L. Su, W. Buntine, R. Newton. **Learning as Applied To Stochastic Optimization for Standard-Cell Placement**. In IEEE Transactions on CAD, Vol. 20, nº 4, April 2001.

[6] E.H.L Aarts and P.J.M. Laarhovn, **A New Polynomial-Time Cooling Schedule**. In: Proceedings of International Conference on Computer-Aided Design, ICCAD, 1985.

[7] R. Hentschke And R, Reis, **Improving Simulated Annealing Placement by Applying Random and Greedy Mixed Perturbations.** In: Proceedings of Brazilian Symposium on Circuits and Systems, SBCCI, 2003.

[8] R. Hentschke, M. Johann, R. Reis, **A Study on the Performance of Fast Initial Placement Algorithms.** In: Proceedings of VLSI-SOC 2003 conference, Darmstadt, Germany.