Prematurely Aborting Linear System Solver in Quadratic Placement

Guilherme Flach, Marcelo Johann, Ricardo Reis
{gaflach, johann, reis}@inf.ufrgs.br

Universidade Federal do Rio Grande do Sul (UFRGS)

Abstract

In this work we develop a new quadratic placement technique where we prematurely abort the linear system solver to save runtime. This has low impact in the placement flow due to the expand-contract phenomena intrinsic to the ICCG solver outlined in this paper. A coloring scheme for placement useful for comparing visually placement results is described. The coloring scheme then is used to create a partitioning strategy to insert more global view in our placement technique. Our placer based on the ideas developed in this paper is 25% faster for largest ISPD 2002 benchmarks than FastPlace 3 with no significant penalty in wirelength.

1. Introduction

By definition, the placement problem is easy to be stated, although solving it efficiently is commonly a challenging task. Simply put, its goal is to place circuit components evenly on the circuit area so that wirelength and other parameters are targeted.

Among several placement techniques, quadratic placement [1] has taken most of academic efforts today since it is in general faster and achieves comparable results to other state-of-the-art strategies. In this paper, we present a new standard cell placement tool inspired by quadratic placement technique.

Our main contributions are:
- a new fast placement technique able to deal with 100K cells in few seconds;
- a new coloring placement scheme to aid placement result comparisons;
- a placement result improvement through partitioning;
- an easy to parallelize placement tool.

2. The Anatomy of a Quadratic Placer

A quadratic placement tool is built around a linear system which describes the cell connectivity as well as the forces that are used to remove overlap between cells as depicted in Figure 1. The solution of the linear system provides the equilibrium cell positions where the total force acting on any cell is zero. When no force is applied besides the connectivity, the system solution is equivalent to the one which minimizes the sum of quadratic distance between connected cells, hence the name quadratic placement.

Since the linear system without overlap removal forces provides invalid solution with large amount of overlap, spreading forces must be added in order to spread cells evenly over circuit area so that overlaps are reduced. The spreading forces are added based on the current placement solution. This creates an iterative process where one interleaves the system resolution and the system modification until the cells are evenly spread over circuit area.

The differences between quadratic placers come from the way the cell connectivity is modeled and mainly from the way the spreading forces are introduced in the linear system.

3. A New Quadratic Placement Technique

As mentioned previously, quadratic placers differs from each other mainly in the way spreading forces are added to the system. Our tool, on the other hand, also adapts the quadratic placement flow by modifying the
way the linear system is solved. In fact, in our tool, the linear system is not entirely solved as the solver is interrupted in its very first iteration. At first glance, this may seem a little disruptive, but it is grounded by our experimental observation that one iteration of the solver suffices to provide a good approximation to where cells should be moving. When interrupting the solver at first iteration, the placer is able to save runtime since in general many iterations are required to converge to the solution.

To come to this idea, we set the following experiment. Using the Incomplete Cholesky Conjugate Gradient (ICCG) method [2] for solving the linear system, we have plot cells at every solver iteration instead of just plotting after the solver converges. This shown a very interesting phenomena: cell positions initially expand and then contract back to their final position. Figure 2 illustrates this phenomena.

The expansion is caused by the spreading forces added to the system whereas the contraction is due to cell connectivity. At initial solver iterations spreading forces cause cell positions to overestimate their final positions. As the system converges, the cells are contracted back until they reach their final position. Although, cell position overestimate final position, the overestimation is bounded by cell connectivity.

![Fig. 2 – Expansion-Contraction ICCG Phenomena.](image)

Looking at flow in Figure 1, after cell have contracted back we apply techniques to spread them. This indicates that we are repeating a process that is intrinsic to the ICCG in its initial iterations. Based on such observation, we develop our placer executing only the first iteration of the solver and then applying the spreading force techniques to reflect the new circuit utilization.

### 3.1. Spreading Forces

Spreading forces are added to the system to reduce overlap between cells. To compute spreading forces in our tool, first the placement area is divided hierarchically in rectangular regions. At first level, the placement area is divided in four same-sized regions. At next level, each region of the previous level is divided in four same-sized regions. The number of levels is chose such that the region area of the last level meet approximately the cell average area.

After placement area is divided, gradient forces are computed at corner of each region in each level. Gradient forces point to high density regions to low density ones. Finally, to compute the spreading force acting on a cell, we interpolates the gradient forces of the four corners at the center of the cell as shown in Figure 3a. This process is done in each level and the final gradient force acting on the cell is the average gradient force for all levels.

![Fig. 3 – Linear system modification: (a) spreading forces; (b) HPWL forces.](image)

### 3.2. HPWL Forces

A common reported drawback of quadratic placement technique is that it only indirectly accounts for half-perimeter wirelength (HPWL) since the system tends to reduce quadratic wirelength. Our placer applies HPWL forces to aim more accurately the HPWL.
HPWL forces affect only cells on the boundary of the bounding-box. For such cells, a force is added in order to push it to the opposite border. This force is fixed, i.e., does not vary on the net cardinality neither the bounding-box area. The computation of HPWL forces is exemplified in Figure 3b.

4. Improving Placement Results Using Partitioning

During the development of our placer, we noticed that our approach is faster, but was not capable to meet wirelength results from other state-of-the-art placer. To try to discover why our placer could not beat other placer in wirelength, we developed a technique for placement coloring to compare visually different placement results. It is known that placement and partitioning are in some extent correlated [3]. In fact, many placers use partitioning as their main algorithm or as a heuristic to improve placement results. Supported by this correlation, our coloring technique use partitioning to color each cell. Basically cells are partitioned into \( n \) groups and to each group a color is set. Due to the correlation between placement and partitioning, it is expected that same-colored cells will be clustered together.

Finally, we compared our placement results with the FastPlace 3 [5] results visually. The comparison for the \( ibm18 \) benchmark [4] is shown in Figure 4 where we can notice that our placer (a), in fact, cluster same-colored cells, but clusters are more messy spread than the result from FastPlace (b). This fact indicates that our placer was not able to deal with the global view of the problem, although it was doing a good job in the local view.

![Fig. 4 – Placement coloring.](image)

Using the results of this coloring techniques, we used partitioning to impose a initial relative order between cells of different color. We partitioned cells in four groups and placed each group in the center of the four placement area quadrants instead of putting all cells in the middle of the circuit. This increased the total run-time of the placement, but allowed our placer to provide state-of-the-art results. The result after partitioning is presented in Figure 4c.

5. Results

To check our placer performance we run it over the ISPD 02 benchmark set [4] and compare it to the state-of-the-art placer FastPlace 3 [5]. FastPlace 3 is currently one of the faster academic placers providing comparable results with other state-of-the-art placers. Our placer receive as input the 4-way partitioning result provide by the hMetis [6]. The run-time of the hMetis is not accounted in the results, however, we point out that for the largest benchmark, \( ibm18 \), it took only 9s to run the partitioning.

![Fig. 5 – Results on the ISPD 02 benchmark set.](image)

In the Table 1 we see that our placer is in average 13% faster while providing the same average wirelength. It is important to highlight that as the benchmark size increases, our placer achieve larger runtime gains, although a little increase in the wirelength is noticed. If we average only the nine largest benchmark w.r.t the number of nodes, the average runtime reach up 25%. This effect can be seen in the Figure 5.
6. Conclusions and Future Work

In this work, we presented a new quadratic placement technique where we prematurely abort the linear system solver at the very first iteration. This has low impact in the placement flow due to the expand-contract phenomena intrinsic to the ICCG solver outlined in this paper. Prematurely aborting the solver save runtime allowing our placement tool run faster than state-of-the-art placers.

We also presented the use of partitioning to insert global view in quadratic placement. The lack of global view in our prior placer was detected thanks to a new scheme for placement coloring developed in this work. The coloring scheme set the cell color based on the partition that cell belongs to and can be used for researches to visually compare their results.

Besides being fast, most of the pieces of our placer are easily to parallelize as the linear algebra operations and gradient computation. This allows our placer to scale well as the circuit size increases.

7. References


