DPlace2.0: A Stable and Efficient Analytical Placement Based on Diffusion *

Tao Luo and David Z. Pan
Department of Electrical and Computer Engineering
The University of Texas at Austin, Austin, TX 78712
tluo@ece.utexas.edu, dpan@ece.utexas.edu

ABSTRACT

Nowadays a placement problem often involves multi-million objects and excessive fixed blockages. We present a new global placement algorithm that scales well to the modern large-scale circuit placement problems. We simulate the natural diffusion process to spread cells smoothly over the placement region, and use both analytical and discrete techniques to improve the wire length. Although any analytical wire length technique can be used in our new framework, by using the quadratic wire length model, the hessian of our formulation is extremely sparse compared with conventional formulations, which brings 24x speed up on quadratic solver. We also propose a wire linearization technique that transform quadratic star model into HPWL exactly. The overall runtime of our tool is close to the fastest placement tool in existing literature and significantly better than others. And meanwhile, we obtain competitive wire length results to the best known ones. The average total wire length is 2.2% higher than mPL6, 0.2%, 3.1%, and 9.1% better than FastPlace3.0, APlace2.0, and Capo10.2 respectively.

1. INTRODUCTION

Although circuit placement has been studied for decades, it continuously attracts research attentions. The placement problems grow rapidly in both problem size and complexity. Some industry placement problems contain multi-million gates and excessive number of blockages [1, 2]. In this paper, we introduce a new diffusion based placer that scales well to large scale placement problems.

Historically, existing circuit placement algorithms can be roughly classified into three major categories, the simulated annealing based approaches [3], the iterative partitioning based approaches [4, 5, 6], and analytical placement approaches [7, 8, 9, 10, 11, 12, 13, 14, 15]. In placement, the Half Parameter Wire Length (HPWL) is a common estimation of the routed wire length. Since HPWL model is not smooth and derivable, quadratic placement optimizes the quadratic form of HPWL [7, 8, 9, 10, 11, 13], and nonlinear model placement [16, 12, 14, 15] adopts a nonlinear estimation of HPWL model, typically the log-sum-exponential wire length approximation patented by Naylor et al. [17].

All 4 existing academic placers [16, 12, 14, 15] that use the log-sum-exponential wire model have achieved excellent wire length results. It is widely agreed that placement uses log-sum-exponential wire model approximates the HPWL much closer than the quadratic estimation. However, although still controversial, some researchers believe that the quadratic placement potentially has advantages for timing driven placement, as the quadratic approximation of the HPWL gives larger penalty on longer wires.

Diffusion is the flow of particles from a region of higher concentration to a region with lower concentration, until the concentration on both regions are equal. The cell spreading in placement shares similar philosophy as the natural diffusion process, where cells are driven from high density areas to low density areas. Diffusion based technique has been successfully applied to incremental placement optimization [18], and here we use diffusion to move cells in global placement.

In this paper, we present DPlace2.0: a stable and efficient diffusion based analytical placement. The DPlace starts with a seed placement, and interleaves two major optimization steps iteratively, 1) the diffusion based cell spreading step to even out the density distribution, 2) the wire length improvement step, which adopts both analytical and discrete wire length improvement techniques. The initial work of DPlace1.0 is presented in a technical report[19]. The following are a few characteristics of our approach, which differentiates our approach from previous analytical placement works.

- We propose a new placement framework based on, but not limited to, diffusion spreading and quadratic placement. Any smooth spreading and wire length optimization technique can be inserted in our framework. Most importantly, we show that it is possible to deal with the overlap removing and wire length improvement tasks separately and achieve fairly good results. Such an approach provides flexibility to plug in additional optimizations/constraints that are non-trivial to integrate in conventional analytical placement framework.
- The *anchor cell* concept is introduced as the bridge integrating different optimization techniques for a stable and efficient global placement, which also significantly reduces the complexity of large scale placement. In each row of the hessian of the quadratic formulation, the anchors inserted change the number of non-zero entries of each row from the number of pins on the corresponding net to the number of pins on the corresponding cell. Such a change helps to speed up the linear system solver by 24 times for ISPD2005 benchmark suite.
- In our placement, to improve the wire length of a given placement by using quadratic formulation, we present a net weight linearization strategy that transforms the star model [20, 10] based quadratic objective into HPWL objective exactly.

In the following, we introduce the analytical placement models in section 2. The details of our global placement are described in section 3. Section 4 is about the legalization and detailed placement. We give the overall algorithm in section 5 and show the experimental results in section 6, which followed by the conclusion in section 7.

2. ANALYTICAL PLACEMENT MODEL

Among existing placement works, analytical placement has been successful in recent years and achieved impressive results on wire length, scalability and the speed of convergence. According to the reported results of ISPD 2005 and 2006 placement contest [21, 2], most of the top ranked placers are analytical placers.

*This work is supported in part by NSF, SRC, IBM Faculty Award, and equipment donations from Intel.

978-1-4244-1922-7/08/$25.00 ©2008 IEEE

Two major tasks in a typical analytical placement iteration are to remove cell overlaps and to reduce the wire length. The conventional analytical placement formulates the wire length and density constraints into a mathematical problem. However, it is possible to address one issue at a time, which gives the flexibility to integrate additional optimizations or constraints during the global placement. In the proposed approach, we iteratively use diffusion to spread out cells, and repair the wire length afterwards.

2.1 Diffusion spreading

In general, an analytical placement tool starts with an initial placement with good wire length. Such an initial placement solution has excessive overlap among cells. Force directed placer adds "spreading force" or density constraints into the original wire length formulation to perturb the placement.

As diffusion has the advantage of smoothness in cell spreading and it preserves the cell relative ordering naturally, we use diffusion to spread cell out directly. Unfortunately, wire length objective is not modeled in the diffusion formulation. We then apply wire length reduction technique on the diffusion solution, such as the quadratic placement formulation, while preserving the diffusion improved density distribution.

2.2 Quadratic placement

In conventional quadratic placement [7], each multi-pin net is transformed into multiple two pin connections with proper weights by either the clique or the star model [20, 10]. Clique model may increase the number of non-zero entries in the connectivity matrix significantly, as the example in Figure 2, which may slow down the quadratic solver. The combination of the clique and star transformation is referred as the hybid model in FastPlace [10].

Assuming there are n movable objects in the netlist. The Hessian matrix of the quadratic system $\mathbf{A}$ is symmetric and positive definite, which is essentially the n x n connectivity matrix of the netlist. Let $\mathbf{x}$ denotes the vector of x coordinates of all cells. $\mathbf{b}$ is the vector encoding all connectivity information between movable and fixed objects, and the pin offsets are captured in $\mathbf{b}$ as well. The minimizer of $\Phi(x)$ can be obtained by taking the gradient of the cost function to zero, $\partial(\Phi(x))/\partial x = \mathbf{0}$, which is determined by the following system of linear equations

$$\mathbf{Ax} = \mathbf{b} \tag{1}$$

Solving the unconstrained wire length minimization problem results a placement with significant overlap among cells. In conventional quadratic placement, overlap is reduce by either partitioning or cell spreading. In our approach, the quadratic solver is used to recover wire length increased during the diffusion spreading.

2.3 The proposed approach

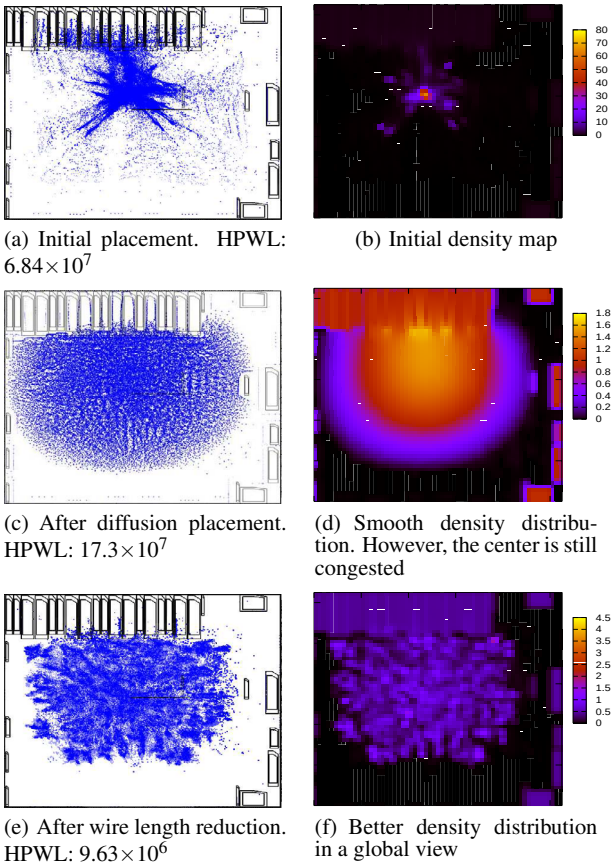
We propose to interleave diffusion with analytical and discrete wire length minimization techniques to improve the wire length, and the quadratic placement formulation is used. Meanwhile, any smooth spreading technique and analytical placement technique can be plugged in our two steps general placement framework.

The anchor cells are used to bridge the overlap reduction and the wire length reduction steps. We use the nets connecting anchor cells and real cells to formulate an unconstrained wire length minimization problem. Those nets are all "real" nets and we do not need to use any artificial nets in our formulation.

If the placement stability is not considered, once a netlist is changed, the placement solutions before and after the change could be completely different. In our approach, we may add additional constraints for placement stability. For example, we may "force" a placement to evolve in a desired way, which potentially provides flexibility for ECO placement.

3. GLOBAL PLACEMENT IN DPLACE

Similar as other analytical placements, our placer starts with a seed placement, which has a fair good initial wire length. In the beginning of each iteration, the diffusion algorithm is called and cells flow away from congested areas.



(a) Initial placement. HPWL: 6.84×10^7

(b) Initial density map

(c) After diffusion placement. HPWL: 17.3×10^7

(d) Smooth density distribution. However, the center is still congested

(e) After wire length reduction. HPWL: 9.63×10^6

(f) Better density distribution in a global view

Figure 1: The diffusion and wire length reduction iteration in bigblue1(from ISPD 2005 placement benchmark suite)

3.1 Diffusion based placement

Diffusion in placement is driven by the density gradient, i.e. the steepness of the density difference. Mathematically, the diffusion process is characterized by the following differential equation.

$$\frac{\partial d_{x,y}(t)}{\partial t} = D\nabla^2 d_{x,y}(t) \tag{2}$$

In the context of placement, $d_{x,y}(t)$ is the cell density at position (x,y) at time t. D is the diffusivity constant, which determines the speed of the diffusion process. The discrete approximation method in [18] can be used to solve the diffusion equation.

In diffusion based placement, the placement region is cut into equal size bins. The bin density is computed as the total cell area enclosed in the bin divided the bin area. The discrete solver we use to solve the diffusion equation evens out the densities between neighboring bins as time proceeds.

In every global placement iteration, cells are diffused from high density area to low density area. The diffusion based pre-placement takes k substeps, where k is relatively small in earlier placement iterations and becomes larger in the later iterations.

Figure 1 shows the diffusion based placement in circuit bigblue1 from ISPD2005 placement benchmark suite. The bin size used is 64x64. Figure 1(a) plots the initial placement seed, which is generated by the quadratic placement. Not surprisingly, the initial placement is extremely poor on density distribution. The highest density is 80 times of the bin capacity and cells are highly congested in the middle of the placement region. Once applying a few iterations of the diffusion spreading, in Figure 1(c), we see how smoothly cells are moved and how effectively the density distribution is improved. The highest density in the placement region is reduced from 80 to less than 2 times of the bin capacity. Unfortunately, the wire length of the diffusion solution increases from 6.84 to 17.3. Figure 1(e) shows that once the wire length reduction techniques are applied, the wire length is reduced to 9.63. Although the densities in a few bins increase, we have a better density distribution in a global view.

The discrete diffusion algorithm is applied on a hierarchical bin structure. Our experiments suggest that a fixed bin size from the beginning to the end do not work well in global placement. If a small bin size is used, cells will spread smoothly, but slowly, which affects the convergence of the algorithm. Furthermore, as shown in Figure 1(e), it is difficult to reduce the density in the central area of congestion if the bin size is too small. We use a large bin size in the beginning of the placement and reduce the bin size down gradually to resolve the local congestions. In addition, we adjust the weight of the density gradients according to local bin density distribution to improve the speed of convergence. i.e. let cells in highly congested area move faster.

3.2 Anchor cells

To preserve the diffusion improved density distribution in the wire length repairing step, there are several choices to prevent cells collapsing back. For example, we can fix a small percentage of cells, or attach some virtual cells to restrict the movement of real cells. And then, the quadratic engine is used to pull free cells toward a better location for improved wire length. In above scenarios, the fixed real or virtual cells are used as anchors to control the movement of real cells, and we name them "anchor cells".

We can either use one anchor per cell or one anchor per net in our framework. An efficient way is to use the star model to transform a portion of multi-pin nets into two-pin connections and use the star as the anchor of real cells. Compared with the method to use one anchor per cell, using stars as anchors will have less impact to the original wire objective and imposes less constraint on cell movements. And we can apply additional HPWL linearization technique by attaching anchor cells to the nets, as shown in later sections.

In the hybrid model based wire length transformation, multi-pin nets are decomposed into two-pin connections by star and clique model. All stars are treated as movable objects and added back into the Hessian matrix **A**, which may increase the dimension of the matrix significantly. In ISPD 2005 benchmark, by using star model with a pin threshold as 5 will increase the dimension of the matrix up to 40 percent. Under conventional formulation, solving one iteration of the system of linear equations with a dimension over 2 million will take several minutes.

Figure 2 shows the quadratic placement formulation of a toy circuit using the clique model. We see how dense is the hessian by using clique transformation. Figure 3 is the formulation by using the star model. The dimension of the Hessian will increase. Unlike stars, anchor cells are fixed objects. Therefore, anchor cells will not increase the dimension of the Hessian **A**. Most importantly, in the anchor cell based quadratic formulation in Figure 4, we see that the Hessian matrix is extremely sparse compared with that by using either the star or clique models.

Let $\mathbf{A}'$ denotes the Hessian matrix in our new formulation. Anchor cells are not movable objects, thus will not appear in $\mathbf{A}'$. Matrix $\mathbf{A}'$ has the dimension as the number of movable objects in the netlist. Once cells are diffused, anchor cells are inserted at the gravity centers of their connected cells and locked. In such a way, anchor cells will pull free cells around in the subsequent wire length minimization.

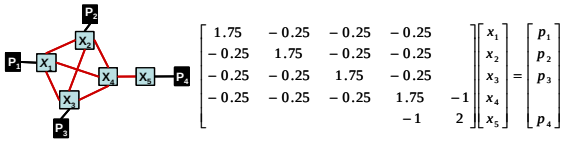


Figure 2: The quadratic placement formulation by using clique model. For simplicity, we assume the weight of each transformed two-pin net is 0.25.

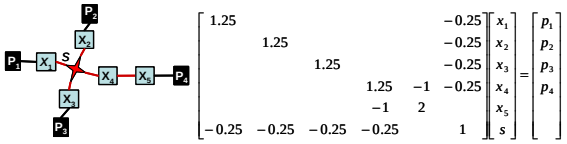


Figure 3: The quadratic placement formulation by using star model. S is the x coordinate of the star, which is a moveable object in the placement. For simplicity, we assume the weight of each transformed two-pin net is 0.25. The dimension of the Hessian matrix A is equal to the number of cells plus the number of stars

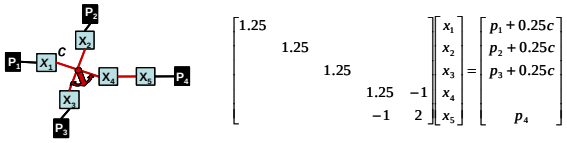


Figure 4: The quadratic placement formulation after the anchor cell insertion. C is the x coordinate of the anchor cell, which is a constant. The new Hessian matrix A is extremely sparse compared with that by using the star or clique formulation.

Figure 5 shows the statistics of the number of non-zero entries in old Hessian **A** and new Hessian $\mathbf{A}'$ for circuit *adaptec*2 in ISPD 2005 benchmark. The dimension of the Hessian **A** is 354K, while only 254K for the new Hessian $\mathbf{A}'$. In most of rows, the number of non-zero entries in **A** are around 3-6, and 1-2 in new Hessian $\mathbf{A}'$. Circuit *bigblue*4 in ISPD 2005 benchmark contains 2 million objects. By using the quadratic solver in [22], in our experiments for bigblue4, it takes 200 seconds for pre-conditioning and 75 seconds for solving using the conventional quadratic formulation, while only 11 seconds for preconditioning and 4 seconds for solving using our anchor cells based formulation.

Note that the anchor cell is different from the fixed point used in mFar [9] and FastPlace [10]. Anchor cell is the bridge connecting the overlap removing and wire length improvement stages. Fixed point is used to add the spreading forces back to the quadratic system and perturb the exiting placement.

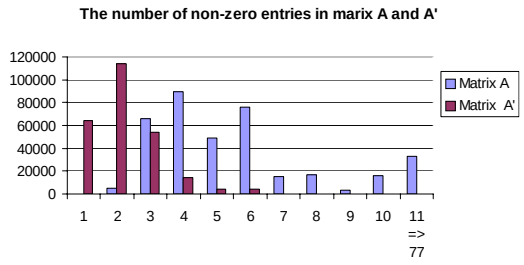


Figure 5: The comparison of non-zero entries in all rows in the sparse matrix A and A$'$. The x-axis is the number of non-zero entries, the y-axis is the row counts. In hessian of the quadratic formulation, each row corresponds to the connected movable objects of each movable cell in the netlist. The insertion of anchor cells change the number of non-zero entries in each row from the pin degree of the corresponding net to the pin degree of the corresponding cell. Therefore, most of rows in matrix A$'$ has only 2-3 non-zero entries. Such linear system takes very short time to solve.

3.3 HPWL transformation in a quadratic system

A major weakness of the quadratic wire formulation is that the quadratic objective is an approximation of HPWL for a two pin nets. Transforming a multi-pin net into multiple two-pin nets may enlarge the gap between HPWL and the actual objective to optimize. To alleviate such a problem, existing techniques iteratively linearize the quadratic wire length objective [23][24]. Here we propose a new liearization technique to transform the quadratic objective into HPWL exactly in our framework, which helps to reduce the gap between quadratic wire length and HPWL.

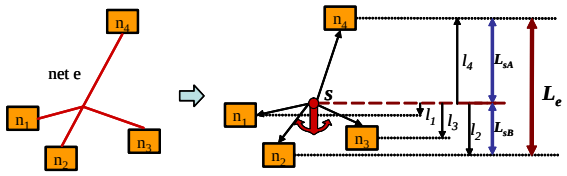


Figure 6: Net weights computation. $A = \{n_4\}, B = \{n_1, n_2, n_3\}$ in this example

Assuming net e is connected with n cells, and HPWL in direction y is L_e. s is added to decompose the net e into n two-pin connections. Let l_i denotes the distance between s and cell i and let w_i denote the weight of each two pin connection. We assign all cells into two sets based on if the cell n_i has a y coordinate large than that of star s. As a result, we have two sets, set $A = \{n_i : y_i > y_s\}$ and set $B = \{n_i : y_i < y_s\}$ for each star model transformation. We define the weight of each two pin net as follows.

$$
\begin{aligned}
w_i &= \frac{L_{sA}}{S_{AB} \times |y_i - y_s|}, \forall n_i \in A \\
w_i &= \frac{L_{sB}}{S_{AB} \times |y_i - y_s|}, \forall n_i \in B \\
&\textit{where} \\
S_{AB} &= 0.5 \sum_{n_i} |y_i - y_s| \\
L_{sA} &= max\{y_i\} - y_s \\
L_{sB} &= y_s - max\{y_i\} \\
L_e &= L_{sA} + L_{sB} \qquad (3)
\end{aligned}
$$

The anchor cell s is placed at the gravity center of all cells on net e, and S_{AB} is defined as the half of the sum of all distances from cell i to the star. Star s splits the length L_e into two parts, L_{sA} and L_{sB}, as shown in Figure 6.

In the following, we show that the above net weighting strategy transforms the quadratic wire length objective into HPWL objective exactly.

$$
\begin{aligned}
\sum_{i=1}^{n} w_i (y_i - y_s)^2 &= \sum_{i \in A} \frac{(y_i - y_s)^2 \times L_{sA}}{|y_i - y_s| \times S_{AB}} + \sum_{i \in B} \frac{(y_i - y_s)^2 \times L_{sB}}{|y_i - y_s| \times S_{AB}} \\
&= \frac{L_{sA}}{S_{AB}} \sum_{i \in A} |y_i - y_s| + \frac{L_{sB}}{S_{AB}} \sum_{i \in B} |y_i - y_s| \\
&= L_{sA} + L_{sB} = L_e \qquad (4)
\end{aligned}
$$

Figure 6 shows an example of 4-pin nets transformation.

3.4 Fixed blockages

Fixed blockages are obstacles to cell spreading. Modern designs may contain a large number of fixed-blockages, which disrupt the cells from smooth spreading. Cells are often placed on top of the fixed-blockages in initial placement, and fixed blockages are density obstacles to prevent cells to pass over. If not properly handled, the wire length may grow dramatically by forcing cells moving out of blockages.

We use a contour-based density smoothing technique to alleviate the density obstacles. First, we identify large blockages, which are those fixed macros with width and height larger than a certain threshold, such as 1% size of the chip size. In the beginning of the global placement, we adjust the density on bins covered by blockages, the adjusted density distribution is contour based. For a bin covered by a big blockage, the bin density is set to be proportional to the distance of the bin to the blockage boundary. Therefore the highest density is in the bin lying in the middle of the blockage.

In the earlier stages of the global placement, the adjusted fixed blockage density is set to a very small value to allow cells to flow over. As the cells spreading stabilizes, the adjusted density increases gradually. The density in the middle of the fixed blockage rises to push overlapping cells out of blockages smoothly. The diffusion based pre-placement pushes cells over blockage easily according to the adjusted density distribution.

3.5 Wire length improvement heuristics

Beside the core techniques proposed above, there are more issues that will affect the final wire length quality. Pushing cells away from a region of congestion often contradicts with wire length optimization objective. Furthermore, Equation 1 optimizes the quadratic wire length, which is an indirect estimation of the linear wire length. The discrenpcy btween the quadratic approximation and HPWL is magnified in large-scale benchmarks, such as the ISPD2005 benchmarks.

In quadratic placement, wire length improving heuristics are crucial for the final HPWL results. The poor initial wire length implies that the initial ordering among cells are not ideal for HPWL, we interleave the medium improvement heuristic [11] and the anchor insertion based technique to generate a better initial cell ordering.

During the placement, wire length improving heuristics are employed between each iteration, which strongly affect the quality of the final HPWL. In DPlace, the quadratic optimization step is very fast and most of the CPU time is on wire length improving heuristics. In our experiments, the medium improvement heuristics used in FDP [11] was found effective in the earlier stages of the global placement. However, the medium improvement heuristic tends to create a lot of overlap in later iterations. The iterative local refinement technique [10] was found effective during the later spreading

Table 1: Statistics on new Hessian A′ and the Hessian A for conventional formulation, and the quadratic solver runtime comparisons

	Matrix **A**				Matrix **A′**				Solver
	Size	Entries	Precon(s)	Solve (s)	Size	Entries	Precon(s)	Solve(s)	speedup
adaptec1	243K	196K	15.85	4.65	211K	430K	0.53	0.19	24.5x
adaptec2	355K	2099K	25.61	7.38	254K	557K	0.90	0.30	24.6x
adaptec3	674K	3713K	38.18	15.61	494K	1131K	1.74	0.58	26.9x
adaptec4	508K	3676K	38.42	15.51	451K	997K	1.97	0.49	31.7x
bigblue1	392K	2287K	29.78	6.87	278K	603K	1.16	0.36	19.1x
bigblue2	729K	3937K	47.79	22.78	535K	1178K	2.29	0.82	27.8x
bigblue3	1389K	7290K	103.93	39.32	1096K	2714K	4.54	1.70	23.1x
bigblue4	2831K	16850K	221.47	75.70	2169K	5190K	10.66	3.91	19.4x
									24.6x

stages. At the point cells stop to spread, iterative local refinment can also be tuned to improve the density distributions. We use the iterative local refinement [10] to improve the density distribution and further reduce the wire length during the later stages of global placement.

4. LEGALIZATION AND DETAILED PLACEMENT

Legalization and detailed placement are non-trivial for the final wire length quality of the placer. Before legalization, we divide the placement region into regular bin structures and analyze the density overflow in each bin. We swap cells out of the overflowed bins and swap cells between bins if such a swap helps to further reducing the wire length. Once the bin density overflow is below a threshold, we run a Tetris [25] like legalization flow. We first legalize all movable macros such that no overlap exist between macros. Blockages/macros will split the placement region into row segments. We identify all row segments, sort cells and pack cells into the closet row segment with the minimum cost.

In this work, we use the FastDP [26] as the detailed placement engine to improve the wire length further.

5. OVERALL ALGORITHM

The overall algorithm of our placer is summarized in **algorithm** 1. In every global placement iteration, cells are diffused to reach a specified density distribution, and the anchor cell based wire length optimization is performed m times to reduce the wire length. The larger m, the shorter the wire length, and the worse the density distribution. Therefore, m is less than 3. We legalize the placement before using FastDP as the detailed placement for final wire length improvement.

Algorithm 1 The Overall Algorithm

```
1:  The global placement
2:      Build matrix A, and matrix A'
3:      Generate an initial quadratic placement with matrix A
4:      Improve the initial cell ordering A
5:      Repeat
6:          Do diffusion based placement for k iterations
7:          Do m iterations
8:              Generate anchor cells and lock them at the gravity centers
9:              Compute HPWL net weights, update A'x = b
10:             Solve x = A'^{-1}b
11:         end
12:         if (In first a few iterations)
13:             Use medium improvement heuristic to repair wire length
14:         else if (Cells are roughly spread)
15:             Use iterative local refinement to repair wire length
16:     Until (reaches a desired density distribution)
17:     Further diffuse cells to remove remaining overlap
18: The legalization
19:     Legalize the macros, then legalize the standard cells
20: The detailed placement
21:     Use FastDP [26] as the detail placer
```

6. EXPERIMENTS

We implement our placer in C++ and run the experiments on a Linux machine with 3.4 GHz 64-bit Xeon processors and 8G memory. We used the Hybrid solver [22] as our quadratic system solver, and use the FastDP [26] as the detailed placer to further improve the wire length. Our current focus is to obtain a good wire length efficiently. We give the wire length and runtime results on ISPD 2005 benchmarks [1].

The anchor cell based formulation in DPlace gives significant advantage on the solving speed of the quadratic solver. Table 1 shows the statistics of the new Hessian matrix $\mathbf{A}'$ used in our placer, versus the Hessian matrix $\mathbf{A}$ in conventional formulation. Column *Size* shows the dimension of the Hessian, and column *Entries* shows the non-zero entries in the Hessian. Column *Precon*. shows the CPU time to preconditioning each Hessian matrix. Same preconditioning quality targets are used for the comparison. Column *Solve* shows the CPU time to solve one iteration of the quadratic system. Comparing with the conventional Hessian $\mathbf{A}$, the new Hessian $\mathbf{A}'$ is about 30% smaller on the dimension of the matrix. Furthermore, because $\mathbf{A}'$ is extremely sparse (Figure 5 and Table 1), the runtime to precondition and solve the new quadratic system are improved significantly. The quadratic solver achieved a 24x times speed up on solving time.

In Table 2, we compare DPlace with some of state of art academic placers, including the FastPlace3.0 [27], mPL6 [12, 28], Capo10.2 [29] and APlace2.0 [30, 16] on the ISPD 2005 placement benchmark. Our placer and FastPlace3.0 are tested on the same machine and compared directly. The HPWL and runtime of mPL6, Cap10.2 and APlace2.0 are derived from the FastPlace3.0 paper [27], in which they are directly compared with that of FastPlace3.0. Although not as accurate as running all placers on the same machine, we can still roughly tell the relative runtime among all placers. For ISPD 2005 benchmark suite, the average HPWL result of DPlace (using the FastDP as the detailed placer) is 0.2% better than FastPlace3.0, and DPlace wins 5 out of 8 circuits. The average HPWL of DPlace is 2.2% higher than mPL6, 9.1% and 3.1% better than Capo10.2 and APlace2.0 respectively. The total runtime of DPlace+FastDP is about 33% longer than FastPlace3.0, 3.4 times faster than mPL6, 7.56 times faster than Capo10.2 and 11.32 times faster than APlace2.0.

We compare the HPWL results of DPlace with that of other placers in ISPD 2005 placement contest in Table 3. It is to be noted that the results in Table 3 were the best possible results generated by each placer, with no runtime limitation, for the ISPD 2005 placement contest. The ISPD2005 results of APlace1.0 are 3.4% better than our placer on average. However, the total runtime of APlace1.0 to finish all circuits are much longer, 113.2 hours for 6 circuits, on a 1.6GHz computer[16], compared with 8.3 hours for 8 circuits in our case, on a 3.4GHz computer. Other than Aplace, DPlace generates the best results among all other placers for their ISPD2005 placement contest versions.

Table 2: Wire length and runtime comparison with FastPlace3.0, mPL6, Capo10.2, and APlace2.0 on ISPD2005 benchmark

Circuits	HPWL $\times 10^6$					Runtime(s)				
	DPlace + fastDP	FastPl3	mPL6	Capo10	APlace2	**DPlace + fastDP**	FastPl3	mPL6	Capo10	Aplace2
adaptec1	78.534	1.011	0.991	1.162	1.001	606	0.69	5.13	10.46	14.99
adaptec2	89.415	1.041	1.031	1.124	1.072	842	0.74	3.56	8.93	14.49
adaptec3	221.817	0.982	0.962	1.031	0.982	1874	0.82	3.12	5.50	9.68
adaptec4	198.506	1.014	0.974	1.045	1.055	1628	0.81	4.68	7.97	17.39
bigblue1	95.339	1.004	1.014	1.144	1.054	903	0.75	4.08	9.98	12.69
bigblue2	160.149	0.962	0.943	1.011	0.953	4656	0.43	2.93	4.99	7.52
bigblue3	363.073	1.046	0.952	1.099	1.130	5409	0.71	1.94	7.01	6.95
bigblue4	869.804	0.958	0.958	1.111	1.005	14026	0.41	1.72	5.61	6.85
Average	**1**	1.002	0.978	1.091	1.031	**1**	0.67x	3.40x	7.56x	11.32x

Table 3: Wire length ($\times 10^6$) comparison with other placers in ISPD 2005 placement contest

Placers	adaptec1	adaptec2	adaptec3	adaptec4	bigblue1	bigblue2	bigblue3	bigblue4	ratio
DPlace + fastDP	78.53	89.41	221.82	198.51	95.34	161.15	363.07	869.80	**1.034**
Aplace	79.50	87.31	218.00	187.65	94.64	143.82	357.89	833.21	1.00
mFAR	-	91.53	-	190.84	97.70	168.70	379.95	876.28	1.06
dragon	-	94.72	-	200.88	102.39	159.71	380.45	903.96	1.08
mPL	-	97.11	-	200.94	98.31	173.22	369.66	904.19	1.09
FastPlace	-	107.86	-	204.48	101.56	169.89	458.49	889.87	1.16
Capo	-	99.71	-	211.25	108.21	172.30	382.83	1098.76	1.17
NTUP	-	100.31	-	206.45	106.54	190.66	411.81	1154.15	1.21
fs50	-	122.89	-	337.22	114.57	285.43	471.15	1040.05	1.50
K&D	-	157.65	-	352.01	149.44	322.22	656.19	1403.79	1.84

7. CONCLUSIONS

In this paper, we present DPlace, a new analytical placement tool for large scale placement. DPlace is based on the diffusion placement technique to spread cells smoothly, which generates a golden placement for improved density distribution. Then DPlace uses the anchor cells based formulation as well as wire length improvement heuristics to reduce the wire length.

In the wire length reduction stage of DPlace, the Hessian matrix of the anchor cells based quadratic formulation is extremely sparse. In our framework, since it is possible to affix explicit cell movement control in the diffusion stage, our new formulation has the potential advantages for ECO and timing driven placement, in which precise cell movement control is required. By using the FastDP as the detailed placement, the HPWL results DPlace are the best among published quadratic placement works, and close to the best reported results on ISPD 2005 placement benchmark suite. Also, the runtime of DPlace is much better than most of state-of-art placers.

8. REFERENCES

[1] G.-J. Nam and et. al., "The ispd2005 placement contest and benchmark suite," in *Proc. ISPD*, 2005.

[2] G.-J. Nam, "Ispd 2006 placement contest: Benchmark suite and results," in *Proc. ISPD*, 2006.

[3] TimberWolf Systems, Inc., "Timberwolf placement & global routing software package," in http://www2.twolf.com/benchmark.html.

[4] A. E. Caldwell and et. al., "Can recursive bisection alone produce routable, placements?," in *Proc. DAC*, 2000.

[5] M. Wang and et. al., "Dragon2000: Standard-cell placement tool for large industry circuits," in *Proc. ICCAD*, pp. 260–263, 2000.

[6] M. C. Yildiz and P. H. Madden, "Improved cut sequences for partitioning based placement," in *Proc. DAC*, 2001.

[7] J. Kleinhans and et.al., "GORDIAN: VLSI placement by quadratic programming and slicing optimization," *IEEE TCAD*, vol. CAD-10, pp. 356–365, Mar. 1991.

[8] H. Eisenmann and F. M. Johannes, "Generic global placement and floorplanning," in *Proc. DAC*, pp. 269–274, 1998.

[9] B. Hu and M. Marek-Sadowska, "Far: fixed-points addition & relaxation based placement," in *Proc. ISPD*, 2002.

[10] N. Viswanathan and C. C. N. Chu, "Fastplace: Efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model," in *Proc. ISPD*, pp. 26–33, 2004.

[11] K. Vorwerk, A. Kennings, and A. Vannelli, "Engineering details of a stable force-directed placer," in *Proc. ICCAD*, 2004.

[12] T. Chan, J. Cong, and K. Sze, "Multilevel generalized force-directed method for circuit placement," in *Proc. ISPD*, 2005.

[13] B. Yao and et.al., "Unified quadratic programming approach for mixed mode placement," in *Proc. ISPD*, 2005.

[14] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, and Y.-W. Chang, "A high quality analytical placer considering preplaced blocks and density constraint," in *Proc. ICCAD*, 2006.

[15] A. R. Agnihotri and P. H. Madden, "Fast analytic placement using minimum cost flow," in *Proc. ASPDAC*, 2006.

[16] A. B. Kahng, S. Reda, and Q. Wang, "Architecture and details of a high quality, large-scale analytical placer," in *Proc. ICCAD*, November 2005.

[17] W. C. Naylor, R. Donelly, and L. Sha", "Non-linear optimization system and method for wire length and dealy optimization for an automatic electric circuit placer," US patent 6,301,693, 2001.

[18] H. Ren, D. Z. Pan, C. J. Alpert, and P. Villarrubia, "Diffusion-based placement migration," in *Proc. DAC*, June, 2005.

[19] T. Luo and D. Z. Pan, "Large scale placement with explicit cell movement control," in *Technical Report UT-CERC-06-01*, April 2006.

[20] F. Mo, A. Tabbara, and R. K. Brayton, "A force-directed macro-cell placer," in *Proc. ICCAD*, p. 4, EECS, UC Berkeley, November 2000.

[21] ISPD_2005_Placement_Contest, "http://www.sigda.org/ispd2005/ispd05/slides/10-1-placement-contest-ispd05.ppt,"

[22] H. Qian and S. S. Sapatnekar, "A hybrid linear equation solver and its application in quadratic placement," in *Proc. ICCAD*, 2005.

[23] G. Sigl, K. Doll, and F. M. Johannes, "Analytical placement: A linear or quadratic objective function?," in *Proc. DAC*, 1991.

[24] P. Spindler and F. M. Johannes, "Fast and robust quadratic placement combined with an exact linear net model," in *Proc. ICCAD*, 2006.

[25] D. Hill, "Method and system for high speed detailed placement of cells within an integrated circuit design," US patent 6,370,673, 2002.

[26] M. Pan, N. Viswanathan, and C. C. N. Chu, "An efficient and effective. detailed placement algorithm," in *Proc. ICCAD*, 2005.

[27] N. Viswanathan and C. C. N. Chu, "Fastplace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control," in *Proc. ASPDAC*, 2007.

[28] T. Chan, J. Cong, J. Shinnerl, K. Sze, and M. Xie, "mpl6: enhanced multilevel mixed-size placement," in *Proc. ISPD*, 2006.

[29] J. Roy, S. Adya, D. Papa, and I. Markov, "Min-cut floorplacement," *IEEE TCAD*, 2006.

[30] A. B. Kahng and Q. Wang, "Implementation and extensibility of an analytic placer," in *IEEE TCAD*, 2005.