ECO-system: Embracing the Change in Placement

Jarrod A. Roy and Igor L. Markov
The University of Michigan, Department of EECS
2260 Hayward Ave., Ann Arbor, MI 48109-2121
{royj,imarkov}@eecs.umich.edu

***Abstract*— In a realistic design flow, circuit and system optimizations must interact with physical aspects of the design. For example, improvements in timing and power may require replacing large modules with variants that have different power/delay trade-off, shape and connectivity. New logic may be added late in the design flow, subject to interconnect optimization. To support such flexibility in design flows we develop a robust system for performing Engineering Change Orders (ECOs). In contrast with existing stand-alone tools that offer poor interfaces to the design flow and cannot handle a full range of modern VLSI layouts, our ECO-system reliably handles fixed objects and movable macros in instances with widely varying amounts of whitespace. It detects geometric regions and sections of the netlist that require modification and applies an adequate amount of change in each case. Given a reasonable initial placement, it applies minimal changes, but is capable of re-placing large regions to handle pathological cases. ECO-system can be used in the range from high-level synthesis, to physical synthesis and detail placement.**

I. Introduction

In his keynote talk at ISPD 2006, Cadence CTO Ted Vucurevich expressed the need for "re-entrant, heterogeneous, incremental, and hierarchical" tools for EDA to handle the challenges of next-generation designs [21]. However, the importance of this problem has been realized much earlier, as Cong and Sarrafzadeh surveyed the state-of-the-art in incremental physical design techniques in 2000 and found these techniques to be largely "unfocused and incomplete" [15]. Kahng and Mantik also found disconnects between the relative strengths of incremental optimizers and perturbation techniques [26]. They conclude that CAD tools of the time "may not be correctly designed for ECO-dominated design processes" [26]. Recent work by Kahng and Reda suggests that certain types of netlist transformations are not handled well by re-placement from scratch, which also motivates incremental tools [27]. Considerable progress has been made since 2000, e.g., in incremental placement [2], [4], [8], [17], [22], [23], [30]–[34], [38], but there is no common agreement on the main tasks solved by incremental tools and how these tasks should be solved. While incremental physical design is not new, it remains a difficult, high-value goal.

We focus on incremental placement legalization and improvement in large-scale layout. The need for such legalization typically arises in two contexts. The first is the separation of placement into global and detail,

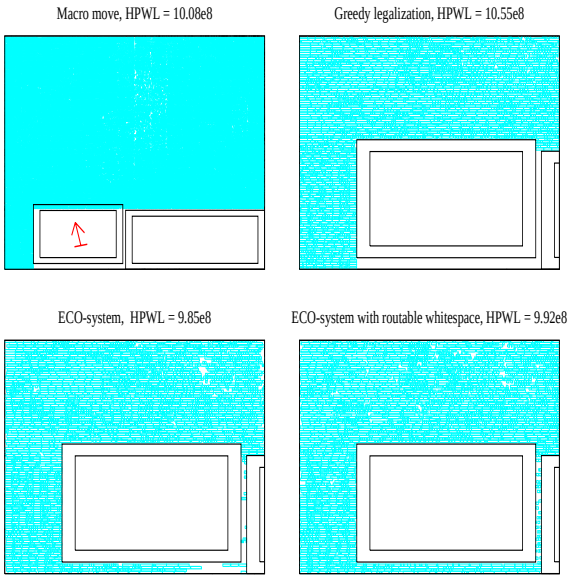

Fig. 1. Legalization of a macro move in the ICCAD'04-Faraday design DSP1 [1]. In the top-left image, the left-most macro is moved toward the north-west corner of the design. This move causes overlap with standard cells and also areas of empty space below and to the right of the macro. The remaining three images are zoomed-in legal placements of this design. In the top-right image, a greedy algorithm which tries to minimize cell movement is applied. Overlap is removed, but the empty space below and to the right of the macro remain unutilized which can be detrimental to routability. The bottom-left image shows the placement as legalized by our tool ECO-system. ECO-system improves wirelength and makes use of much of the area vacated by the macro. Lastly, the bottom-right image shows how ECO-system can distribute cells and whitespace so as to ensure routability and/or satisfy minimum whitespace constraints.

where rough placements are produced first and incrementally improved to avoid overlaps and fit into cell sites. This is common for analytical placers (APlace [28], mPL [11]) that approximate site constraints, while partitioning-driven tools (Capo [39], PolarBear [16]) and annealing-based tools (mPG [12], Parquet [3]) adopt correct-by-construction frameworks and require little post-processing.

However, the second context for legalization appears entirely unavoidable. During physical synthesis, timing-critical gates may be powered up and other gates may be powered down. These changes affect gate size and typically create overlaps [31]. Buffer insertion often leads to similar area violations, which must be resolved by legalization. The success of such legalization depends on how much the areas have changed, in what patterns, and the strength of a given legalizer. In particular, the legalization of mixed-size and block-based designs with obstacles remains very challenging [36].

Our work is focused on the design of a powerful and robust ECO tool that applies adequate amounts of replacement, in the right locations, to accommodate necessary design changes. To be useful in high-level and physical synthesis, such a tool must be able to entirely replace sections of the netlist, e.g., logic added to the design.

While practical considerations call for an interaction between global placers and legalizers, traditional work on ECO and detail placement focuses on stand-alone tools incapable of global placement. An attractive, but yet unexplored solution would be to extend an existing global placer to an incremental mode where it would automatically identify layout regions and sections of the netlist that need repair, but preserve satisfactory regions. In this work, we propose such an extension, identify and develop new components that allow a global placer to act like a powerful ECO tool, and develop a competitive implementation based on the open-source Capo tool.

As this tool can always resort to calling global placement on the entire design, it robustly handles a full range of modern designs, including those with obstacles and movable macros. Time-consuming global placement is not used when the initial placement is good.

We formulate the basic requirements for ECO placement and offer relevant algorithms. Our tool, ECO-system, is many times faster than a global placer and increases wirelength only slightly. ECO-system outperforms APlace's native legalizer on APlace global placements by over 1% in HPWL while running four times faster. ECO-system supports extensive cell resizing producing legal results that mirror the original with virtually the same HPWL while having minimal impact on timing. Unlike WSA [30], [31], we handle obstacles and displace cells an order of magnitude less.

TABLE I

A COMPARISON OF SEVERAL LEGALIZATION AND INCREMENTAL PLACEMENT TECHNIQUES. FOR EACH OF THE TECHNIQUES, ITS COMPATIBILITY WITH FIXED OBJECTS OR MACROS AS WELL AS WHAT GENERAL TECHNIQUES IT USES ARE LISTED. ECO-SYSTEM IS COMPARED WITH XDP [17] IN SECTION VI. (†) SUPPORT OF THE FEATURE BY THIS TECHNIQUE IS UNCLEAR. SEE SECTION II-A FOR MORE DETAILS. (‡) RECENT VERSIONS OF CAPO, THE BASIS OF ECO-SYSTEM, USE LINEAR PROGRAMMING AND NETWORK FLOWS IN DETAIL PLACEMENT, BUT THEY ARE BEYOND THE SCOPE OF THIS WORK.

		Capo [39]	Diffusion [33], [38]	DOMINO [18]	WSA [30], [31]	XDP [17]	**ECO-system**
Features	Fixed-object support	✓	✓			†	✓
	Macro support	✓	†		†	✓	✓
	Whitespace redistribution				✓		✓
Techniques used	Cell swapping					✓	✓
	Greedy legalization	✓	✓		✓	✓	✓
	Linear programming					✓	‡
	Network flows			✓		✓	‡
	Sliding-window optimization				✓	✓	✓

The rest of the paper is structured as follows. In Section II we review previous work. Key requirements and a likely interface are discussed in Section III. We present ECO-system in Section IV. Support for high-level and physical synthesis is discussed in Section V. In Section VI we show empirical results and conclude in Section VII.

II. PREVIOUS WORK

Below we describe existing work on incremental techniques and relevant aspects of global placement.

A. Incremental techniques

Previous work on legalization, incremental placement and detail placement can be broken into three fairly distinct stages: i) cell spreading, ii) legalization through simple end-case techniques, and iii) refinement of the legalized placement. For the first stage, several algorithmic paradigms have been applied in the literature such as network flows [8], [17], [18], [32], linear programming [17], top-down whitespace injection [30], [31] and diffusion gradients [38]. For end-case legalization, generally placers use greedy movement of cells such as in Capo [39], the Tetris legalizer [22] in FengShui [5], and greedy packing in DOMINO [18]. Lastly, placement refinement is done in sliding windows of one or more rows using optimal end-case placers based on branch-and-bound [9] or dynamic programming [23], as well as cell swapping such as in FastPlace [37].

One major theme in much of the literature is minimizing the total movement of cells in the design during legalization [8]. While our legalizer achieves remarkably small total/average movement, we point out that in general this does not always lead to minimal increase in interconnect parameters as shown in [7]. A legalization with minimal total cell displacement may cause a few cells to move a great distance. Better timing may be achieved by legalization with greater average movement, and even if the average movement is the same, there can be many alternative replacements.

Cell spreading. The term "cell spreading" has been used by several authors in different contexts. In particular, several papers describe algorithms that do not take interconnect into account, while ECO-system includes interconnect optimization. Some of these publications do not describe the handling of movable and especially fixed obstacles, while ECO-system handles both, as confirmed by our experiments.

DOMINO [18] legalizes by splitting cells into pieces of identical sizes, solving a flow formulation to minimize movement, and finally reassembling the cell pieces. This limits the effectiveness of DOMINO to cells of similar sizes. Existing implementations of DOMINO do not account for obstacles and shift all cells to the left, limiting their applicability to modern placement instances, such as those from the ISPD'05 contest [35]. Flow-based legalization methods such as those used in [8], [32] divide the core area into regions and redistribute cells between neighboring regions until no region has more cell area than available site area. These techniques can handle movable macros by fixing them early in the legalization process.

In [30], [31] cells are incrementally placed by injecting whitespace in a top-down fashion. The placement region is divided into a regular grid with geometric bisection steps (based only on the size and shape of the region, not taking into account the cells, macros or fixed obstacles therein), and whitespace is injected based on some particular objective (routing congestion in [30], gate sizing and buffer insertion in [31]). Whitespace injection is done by shifting the geometric cut-lines to change the whitespace balance in regions. When cut-lines are shifted, the positions of the cells in the affected regions are scaled. Whitespace injection can cause significant overlap due to scaling, especially in the presence of fixed obstacles or movable macros as in the ISPD 2005 Contest benchmarks [35]. To remove these overlaps, a standard legalization step must be applied followed by window-based detail placement to recover HPWL. It is unclear how well this technique may work on difficult block-packing instances [36]. In addition, the most current implementation of this technique, WSA, does not support macros. The technique may also fail in cases of extreme overlap, such as global placement by analytical placers, as large areas of the placement will be essentially random. The authors of [31] report an average displacement of 2.1% of the core half-perimeter per cell, whereas the displacements observed with our technique are an order of magnitude smaller.

The diffusion technique of [38] legalizes by dividing the core area into a regular grid. Cells move from areas of high congestion to lower congestion (moving around fixed obstacles) and their directions and speeds are determined by solving equations similar to those in the process of chemical diffusion [38]. New placements are generated at each time step of the diffusion and the first solution which satisfies area constraints is taken to minimize runtime and cell movement [38]. End-case legalizers work within the grid regions to produce a final legal placement, but this may be impaired by difficult block-packing instances [36]. The work in [33] improves that in [38], but does not measure its impact on wirelength, congestion or timing.

The XDP technique [17] uses a combination of constraint graphs, network flows, linear programming and greedy cell movement for legalization of mixed-size designs. Overlaps between macros are legalized first by building constraint graphs until all macros can legally fit into the core. After the constraint graph is finalized, a linear programming instance is built and solved to remove macro overlap and move macros minimally. It is unclear if this technique supports fixed macros. Standard cells are legalized with a greedy heuristic similar to that of FengShui [5], with the addition of flow-based methods [8], [32] as necessary. After legalization, window-based detail placement techniques are used to improve HPWL. XDP is currently the detail placement technique used by the placer mPL6 [10], and is evaluated in our experiments.

Greedy legalization. FengShui [5] uses a simple packing algorithm by Hill [22] that is reminiscent of the Tetris game. Such legalization fares poorly in designs with large amounts of whitespace, as shown by the results of the ISPD 2005 Placement Contest. Capo uses two greedy legalizers for its global placements: one for macros and another for standard cells [39]. The macro overlap legalizer tries to move macros as little as possible so as not to affect neighboring standard cells. If space is available, standard cells are legalized via shifting. Otherwise cells are swapped between rows greedily until no row is overfull. Fixed obstacles are handled implicitly as they fracture rows [39].

Macro legalization. It was shown that a fixed-outline floorplanner based on Simulated Annealing with sequence pairs could be used to remove overlap [2]. Techniques in [45] improve on [2] and show how to legalize with minimal perturbation. Removal of overlap between macros can be especially difficult given hard instances of block-packing [36]. To handle

such instances, the authors of [36] modify B*-trees to account for obstacles. Recently, FLOORIST [34] has been proposed which uses constraint satisfaction to remove macro overlap.

B. Min-cut placement

ECO-system uses the top-down min-cut placement framework [5], [16], [36], [39], [40], [42]. Recent techniques for min-cut placement [13], [43] have produced some of the best placements in the ISPD 2006 contest [25] and the most routable placements on IBMv2 netlists [40]. In traditional min-cut algorithms, a placement is viewed as a series of placement bins, the first of which encompasses the core area and contains all movable cells. Based on number of cells in a placement bin, the placer either bisects the bin or places the bin's cells with an end-case placer.

When bisecting a bin, a min-cut placer proceeds by selecting a temporary cut-line for the bin based on the size and shape of the bin. Based on the amount of cell and site area in the bin, the placer determines partitioning tolerances. Given the tolerance, the placer uses a balanced min-cut partitioner to determine how to divide the cells between its child bins. Using the partitioning solution, the placer determines a final cut-line based on whitespace allocation techniques and divides the bin into child bins for further processing.

III. Requirements of Incremental Placement

Design optimizations that require incremental placement can alter a design in many ways [19] such as (see also Section V):

- Changing cell dimensions or net weights/criticalities
- Adding/Removing various constraints, such as density (to promote routability), regions (to address timing), etc.
- Inserting/Removing cells (with or w/o initial locations), nets or macros
- Adding/Moving obstacles (memories, IP blocks, RTL macros, etc.)

Generally these transformations create illegality in localized regions of a design and/or create opportunities for improving an existing placement. All of these transformations can be dealt with by performing placement from scratch, but this is undesirable: i) replacement can be slow, ii) the transformations may assume that they are applied to the current layout, and placement from scratch may invalidate them, and iii) the current layout may include *intangibles* such as designer intent, or be optimized for novel objectives not accounted for by the placement tool. Cong and Sarrafzadeh point out that incremental placers need to be able to trade off potentially several design objectives when operating on a placement [15].

```
Variables: queue of placement bins
Initialize queue with top-level placement bin
1  While(queue not empty)
2     Dequeue a bin
3     If(bin not marked to place from scratch)
4       If(bin overfull)
5         Mark bin to place from scratch, break
6       Quickly choose the cut-line which has
          the smallest net-cut considering
          cell area balance constraints
7       If(cut-line causes overfull child bin)
8         Mark bin to place from scratch, break
9       Induce partitioning of bin's cells from cut-line
10      Improve net-cut of partitioning with
          single pass of Fiduccia-Mattheyses
11      If(% of improvement > threshold)
12        Mark bin to place from scratch, break
13      Create child bins using cut-line and partitioning
14      Enqueue each child bin
15    If(bin marked to place from scratch)
16        If(bin small enough)
17          Process end case
18        Else
19          Bi-partition the bin into child bins
20             Mark child bins to place from scratch
21          Enqueue each child bin
```

Fig. 2. Our ECO algorithm. Lines 3-15 and 20 are different from traditional min-cut placement.

In addition to preserving the original placement, a legalizer must also be able to completely replace sections of the placement that are deemed too suboptimal after design alterations. For example, if all of the cells are moved on top of one another at the center of the placement area, the legalizer should have the ability to replace all of the cells as the initial placement gives little useful information about a legal placement of the design. While this example is not typical of legalization as a whole, it is quite possibly the case for small sections of an illegal placement. This pathological case is not considered by most legalization techniques (such as those described in Section II).

Take for example the case when new cells are added to a design. If the new cells are added to isolated regions of the design, such as during buffer insertion, traditional techniques that perturb the design only slightly are most likely appropriate. Yet, timing optimization may call for pipelining of a multiplier or changing an adder to a different type. Adding a significant amount of new logic to an already placed and optimized design will require the functionality of a full-blown placer rather than just cell spreading to avoid degrading the design's wirelength and timing characteristics.

IV. Top-Down Legalization

To develop a strong ECO tool, we build upon an existing global placement framework and must choose between analytical and top-down. The main considerations include robustness, the handling of movable macros and fixed obstacles, as well as consistent routability of placements and the handling of density

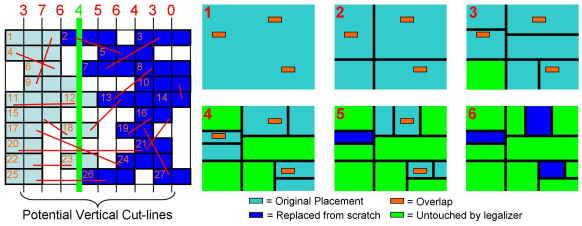

Fig. 3. Fast legalization by ECO-system. The image on the left illustrates choosing a vertical cut-line from an existing placement. Nets are illustrated as red lines. Cells are individually numbered and take 2 or 3 sites each. Cut-lines are evaluated by a left-to-right sweep (net cuts are shown above each line). A cut-line that satisfies partitioning tolerances and minimizes cut is found (thick green line). Cells are assigned to "left" and "right" according to the center locations. On the right, placement bins are subdivided using derived cut-lines until i) a bin contains no overlap and is ignored for the remainder of the legalization process or, ii) the placement in the bin is considered too poor to be kept and is replaced from scratch using min-cut or analytical techniques.

constraints. Based on recent empirical evidence [36], [40], [42], the top-down framework appears a somewhat better choice. Indeed the 2 out of 9 contestants in the ISPD 2006 Competition that satisfied density constraints were top-down placers. However, analytical algorithms can also be integrated into our ECO-system when particularly extensive changes are required. We base ECO-system on the open-source min-cut placer Capo [39] and plan to distribute it with Capo as well.

A. General framework

The goal of ECO-system is to reconstruct the internal state of a min-cut placer that could have produced a given placement **without the expense of global placement**. Given this state, we can choose to accept or reject previous decisions based on our own criteria and build a new placement for the design. If many of the decisions of the placer were good, we can achieve a considerable runtime savings. If many of the decisions are determined to be bad, we can do no worse in terms of solution quality than placement from scratch. After this modified global placement, we use a subset of Capo's detail placement to guarantee legality. An overview of the application of ECO-system to an illegal placement is depicted in Figure 3. See the algorithm in Figure 2.

To rebuild the state of a min-cut placer, we must reconstruct a series of cut-lines and partitioning solutions efficiently. To extract a cut-line and partitioning solution from a given placement bin, we examine all possible cut-lines as well as the partitions they induce. We start at one edge of the placement bin (left edge for a vertical cut and bottom edge for a horizontal cut) and move towards the opposite edge. For each potential cut-line encountered, we maintain the cell area on either side of the cut-line, the partition induced by the cut-line and the net cut.

B. Fast cut-line selection

For simplicity, assume that we are making a vertical cut and are moving the cut-line from the left to the right edge of the placement bin (the techniques necessary for a horizontal cut are analogous). Pseudo-code for choosing the cut-line is shown in Figure 4. To find the net cut for each possible cut-line efficiently, we first calculate the bounding box of each net contained in the placement bin from the original placement. We create two lists with the left and right x-coordinates of the bounding boxes of the nets and sort them in increasing x-order. While sliding the cut-line from left to right (in the direction of increasing x-coordinates), we incrementally update the net-cut and amortize the amount of time used to a constant number of operations per net over the entire bin. We do the same with the centers of the cells in the bin to incrementally update the cell areas on either side of the cut-line as well as the induced partitioning. While processing each cut-line, we save the cut-line with smallest cut that is legal given partitioning tolerances. An example of finding the cut-line for a partitioning bin is shown in Figure 3.

Once a partitioning has been chosen, we accept or reject it based on how much it can be improved by **a single pass of a Fiduccia-Mattheyses partitioner with early termination** (which takes only several seconds even on the largest ISPD'05 circuit). [1] The intuition is that if the constructed partitioning is not worthy of reuse, a single Fiduccia-Mattheyses pass could improve its cut non-trivially. If the Fiduccia-Mattheyses pass improves the cut beyond a certain threshold, we discard the solution and bisect the entire bin from scratch. If this test passes, we check legality: if a child bin is overfull, we discard the cut-line and bisect from scratch.

C. Scalability

Pseudo-code for the cut-line location process used by ECO-system is shown in Figure 4. The runtime of the algorithm is linear in the number of pins incident to the bin, cells incident contained in the bin, and possible cut-lines for the bin. Since a single Fiduccia-Mattheyses pass takes also takes linear time [20], the asymptotic complexity of our algorithm is linear. If we let P represent the number of pins incident to the bin, C represent the number of cells in the bin and L represent the number of potential cut-lines in the bin, the cut-line selection process runs in $O(P+C+L)$ time. In the vast majority of cases, $P > C$ and $P > L$, so the runtime estimate simplifies to $O(P)$.

The number of bins may double at each hierarchy layer, until bins are small enough for end-case

[1] We do not assume that the initial placement was produced by a min-cut algorithm.

```
Input: placement bin, balance constraint
Output: x-coord of best cut-line
1  numCutlines =
    1+⌊(rightBinEdgeX−leftBinEdgeX)/cellSpacing⌋
2  Create three arrays of size numCutlines:
    LEFT, RIGHT, AREA
3  Set all elements of LEFT, RIGHT, and AREA to 0
4  Foreach net
5    Calculate x-coord of left- and right-most pins
6    leftCutlineIndex =
      max(0,⌈(leftPinX−leftBinEdgeX)/cellSpacing⌉)
7    rightCutlineIndex =
      max(0,⌈(rightPinX−leftBinEdgeX)/cellSpacing⌉)
8    if(leftCutlineIndex < numCutlines)
9      LEFT[leftCutlineIndex]+=1
10   if(rightCutlineIndex < numCutlines)
11     RIGHT[rightCutlineIndex]+=1
12 Foreach cell
13   Calculate x-coord of the center of the cell
14   cutlineIndex =
      max(0,⌈(centerX−leftBinEdgeX)/cellSpacing⌉)
15   if(cutlineIndex < numCutlines)
16     AREA[cutlineIndex]+=cellArea
17 Set X = leftBinEdge, CURCUT = 0, BESTCUT = ∞
   BESTX = ∞, LEFTPARTAREA = 0
18 For(I = 0;I < numCutlines;I+=1,X+=cellSpacing)
19   CURCUT+=LEFT[I]
20   CURCUT−=RIGHT[I]
21   LEFTPARTAREA+=AREA[I]
22   If(CURCUT < BESTCUT and
        LEFTPARTAREA satisfies balance constraint)
23     BESTCUT = CURCUT
24     BESTX = X
25 Return BESTX
```

Fig. 4. Algorithm for finding the best vertical cut-line from a placement bin. Finding the best horizontal cut-line is largely the same process. Note that the runtime of the algorithm is linear in the number of pins incident to the bin, cells contained in the bin, and possible cut-lines for the bin.

placement. End-case placement is generally a constant amount of runtime for each bin, so it does not affect asymptotic calculations. Assume that ECO-system is able to reuse all of the original placement. Since ECO-system performs bisection, it will have $O(\log C)$ layers of bisection before end-case placement. At layer i, there will be $O(2^i)$ bins, each taking $O\left(\frac{P}{2^i}\right)$ time. This gives a total time per layer of $O(P)$. Combining all layers gives $O(P \log C)$. Empirically, the runtime of the cut-line selection procedure (which includes a single pass of a Fiduccia-Mattheyses partitioner) is much smaller than partitioning from scratch. On large benchmarks, cut-line selection requires 5% of ECO-system runtime time whereas min-cut partitioning generally requires 50% or more of ECO-system runtime.

D. Handling macros and obstacles

With the addition of macros, the flow of top-down placement becomes more complex. We adopt the technique of "floorplacement" which proceeds as traditional placement until a bin satisfies criteria for block-packing [36], [39]. If the criteria suggest that the bin should be packed rather than partitioned, a fixed-outline floorplanning instance is induced from the bin where macros are treated as hard blocks and standard cells are clustered into soft blocks. The floorplanning instance is given to a Simulated Annealing-based floorplanner to be solved. If macros are placed legally and without overlap, they are considered fixed. Otherwise, the placement bin is merged with its sibling bin in the top-down hierarchy and the merged bin is floorplanned. Merging and re-floorplanning continues until the solution is legal.

We add a new floorplanning criterion for our legalization technique. If no macros in a placement bin overlap each other, we generate a placement solution for the macros of the bin to be exactly their placements in the initial solution. If some of the macros overlap with each other, we let other criteria for floorplanning decide. If block-packing is invoked, we must discard the placement of all cells and macros in the bin and proceed as described in [39].

During the cut-line selection process, some cut-line locations are considered invalid — namely those that are too close to obstacle boundaries but do not cross the obstacles. This is done to prevent long and narrow slivers of space between cut-lines and obstacle boundaries. Ties for cut-lines are broken based on the number of macros they intersect. This helps to reduce overfullness in child bins allowing deeper partitioning, which reduces runtime.

E. Controlling overlaps, whitespace and congestion

We introduce techniques and user controls for ECO-system and show how they can be used for reallocation of whitespace and congestion improvement in the original placement.

Relaxing overfullness constraints. One of the primary objectives of ECO-system is to reuse as much relevant placement information as possible from a given placement. As described above, it is possible to find a cut-line which has a good cut but is not legal due to space constraints. In these cases, ECO-system must discard these good solutions and partition from scratch.

In order to make better use of the given placement, we propose the following addition to ECO-system. In these situations, we allow ECO-system to shift the cut-line to legalize the derived partition with respect to area. Cut-line shifting is a technique commonly used in the top-down min-cut placement for allocation of whitespace [4], [30], [31], [40], [42]. The cut-line is shifted as little as possible to make the derived partitioning legal with respect to area. If it is impossible to find an area-legal cutline, the derived partitioning must be discarded and ECO-system proceeds normally.

If cut-line shifting is successful in correcting the illegality, the original placement must be modified for purposes of consistency. To do so, cells are scaled proportionately within the placement bin based on their original positions, the position of the originally chosen cut-line and the position of the shifted cut-line in a

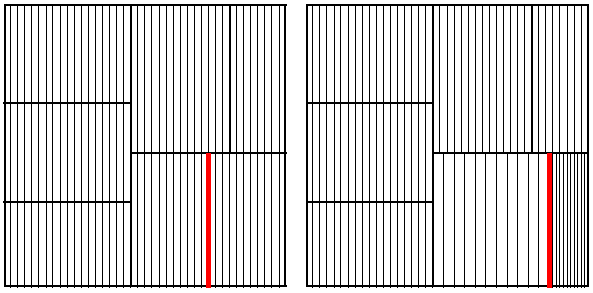

Fig. 5. Shifting a cut-line chosen during ECO cut-line selection. Unlike the WSA technique [30], [31], cut-line shifting during ECO is not done on geometric cut-lines but instead on those cut-lines which are chosen during fast cut-line selection. The image on the left shows a placement that has been divided into bins during the course of ECO-system. In the image on the right, the chosen cut-line of the bottom-right bin is shifted to the right. The density of vertical lines represents the initial placement and its scaling around the moving cutline (shown in red).

manner similar to that in the WSA technique [30], [31]. As the centers of cells are used to determine in what partitions cells belong during fast cut-line selection, we shift cell locations based on center locations as well to ensure that cut-line shifting will not change derived partitions. We seek to shift cell locations and maintain the following property: the relative position between cells before and after shifting is maintained. Also, if a cell were in the middle of a partition before shifting, it should remain in the middle of a partition after shifting. Let x_L and x_R represent the x-coordinates of the left and right sides of the placement bin, x^{cut}_{orig} and x^{cut}_{new} the x-coordinates of the original and new cuts, and, lastly, x^{cell}_{orig} and x^{cell}_{new} the x-coordinates of the center of a particular cell before and after shifting. We wish to maintain the following ratios (for vertical partitioning):

$$\frac{x^{cell}_{orig} - x_L}{x^{cut}_{orig} - x_L} = \frac{x^{cell}_{new} - x_L}{x^{cut}_{new} - x_L}, \; x^{cell}_{orig} \leq x^{cut}_{orig}$$

$$\frac{x_R - x^{cell}_{orig}}{x_R - x^{cut}_{orig}} = \frac{x_R - x^{cell}_{new}}{x_R - x^{cut}_{new}}, \; x^{cell}_{orig} > x^{cut}_{orig}$$

Solving for x^{cell}_{new} :

$$x^{cell}_{new} = \begin{cases} x_L + \left(x^{cell}_{orig} - x_L\right) \frac{x^{cut}_{new} - x_L}{x^{cut}_{orig} - x_L}, & x^{cell}_{orig} \leq x^{cut}_{orig} \\ x_R - \left(x_R - x^{cell}_{orig}\right) \frac{x_R - x^{cut}_{new}}{x_R - x^{cut}_{orig}}, & x^{cell}_{orig} > x^{cut}_{orig} \end{cases}$$

The new y-coordinates of cells shifted during horizontal partitioning are calculated analogously.

Figure 5 illustrates the scaling involved when a cut-line is shifted. In the figure, the cut-line of the bottom-right bin is shifted to the right. All objects to the left and right of the cut-line are scaled appropriately. Objects that were to the left of the original cut-line remain to the left and are spread out and objects on the right are packed closer together.

Shifting proportionately in this way maintains the relative ordering of all the cells within the current placement bin. Also the partitioning induced by the cutline remains unchanged so ECO-system can proceed as normal. Shifting the cut-line in this manner can allow deeper ECO partitioning which can reduce both runtime and cell displacement.

Satisfying density constraints. A common method for increasing the routability of a design is to inject whitespace into regions that are congested [4], [30]. One can also require a minimum amount of whitespace (equivalent to a maximum cell density) in local regions of the design to achieve a similar effect [42]. As one of ECO-system's legality checks is essentially a density constraint (checking to see if a child bin has more cell area assigned to it than it can physically fit), this legality check is easy to generalize. The new criterion for switching from using the initial placement and partitioning from scratch is based on a child bin having less than a threshold percent of relative whitespace, which is controlled by the user.

The cut-line shifting feature of ECO-system can also be used to satisfy density constraints. As ECO-system proceeds, cut-lines can be shifted as described above to implement a variety of whitespace allocation schemes [30], [31], [40], [42]. Specifically, ECO-system can implement the hierarchical whitespace injection of WSA [30], [31]. WSA chooses cut-lines based only on the geometry of a placement bin and shifts these cut-lines from the top down. ECO-system chooses cut-lines that are more natural to the original placement, shifts cut-lines top-down, and also supports fixed objects and movable macros. Figure 1 shows the power of the cut-line shifting technique in redistributing whitespace for routability after making a change to a placement that causes significant overlap.

V. Using ECO-system in High-level and Physical Synthesis

We extend the proposed framework to offer users efficient access to the features of incremental placement described in Sections III and IV as well as provide greater user control and flexibility.

A. Additional user controls

We present further controls over ECO-system to vary how much it is allowed to modify a given placement as well as what regions of a placement are allowed to be changed, which can both be beneficial to a designer. We also illustrate how ECO-system can be used to re-optimize placements based on changes to net weights. This control can be extremely useful when critical nets in a design change, for example.

Tunable aggressiveness. ECO-system accepts or rejects derived partitioning solutions based on how much a single pass of a Fiduccia-Mattheyses partitioner can improve them. If the partitioner improves the net cut by more than a threshold percentage, the

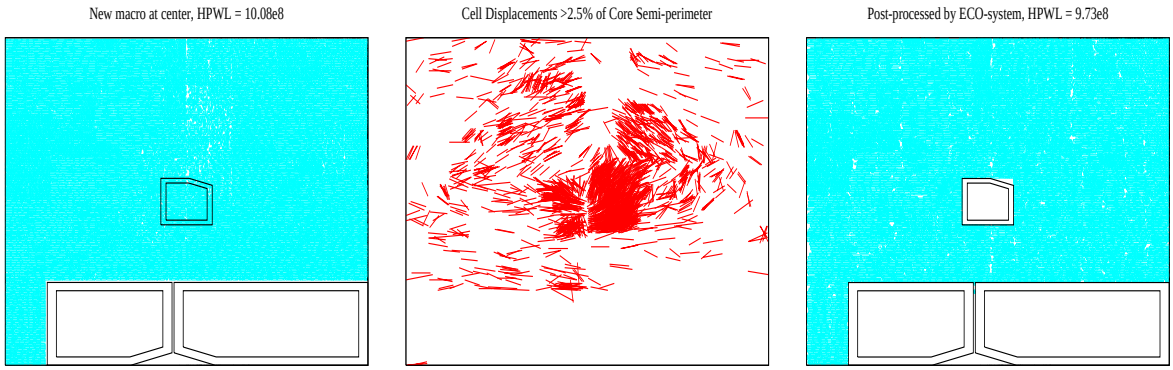

Fig. 6. Legalizing the placement of a new fixed obstacle at the center of the ICCAD'04-Faraday design DSP1 [1]. The picture in the middle shows the movement of standard cells to make room for the obstacle. Many standard cells must move in order to accommodate the obstacle, but ECO-system moves these cells on average only a short distance (1.27% of the core half-perimeter) and is able to improve total HPWL.

partitioning solution is rejected. This threshold can be adjusted by the user so as to prevent ECO-system from performing large changes. If a designer wants ECO-system to change the placement as little as possible, the improvement threshold can be given as 100%. Tunable aggressiveness also allows one to adjust the strength of ECO-system legalization to better correlate with the magnitude of design modifications [26].

User-defined locality. ECO-system operates automatically on the given placement and quickly focuses on sections of overlap. It may be the case that a designer has performed optimization on only a small portion of the design. Having our algorithm run over the entire design to find this small area is potentially wasteful. Thus we allow the user or a physical synthesis tool to specify one or more regions of the placement area to apply legalization. Combined with whitespace control techniques described above, this allows a designer to re-tune whitespace allocation to reduce congestion in localized regions of the design.

While this control can be useful to designers to ensure that certain regions of a design remain untouched, it is not a replacement for the automatic techniques of ECO-system. Changes made to one region of a design can affect the quality of the placement in a separate area of the design. Patch-based replacement of a design does not handle this situation well because the patches must be supplied but may not be well-defined. Also, the processing of given patches in a particular order can make the legalization within the first patch inconsistent with that in subsequent patches. However, ECO-system can automatically narrow down the regions that require extra work, partition them, and simultaneously perform top-down legalization in all regions to ensure consistency. Cut-line shifting in ECO-system is truly hierarchical and allows ECO-system to subsume other hierarchical techniques such as WSA [30], [31] while also supporting fixed objects and movable macros.

Changing net weights. Having a legal placement facilitates more precise static timing analysis and finding timing-critical nets. To improve timing, weights can be increased for nets with worst slack, and decreased for non-critical nets. As ECO-system checks if the cut of an induced partitioning solution can be improved significantly, net weights are naturally integrated into this test. With weighted cut, ECO-system recognizes instances when the initial placement can be improved.

B. Placing new cells and macros

The addition of macros, IP blocks and embedded memories to an already placed netlist can introduce significant overlap as can be seen in Figure 6. Large modules may need to be fixed due to alignment constraints and will appear as obstacles. Buffer insertion is also a concern as numerous buffers may need to be inserted. There are typically few legal locations for buffer insertion, and, compounding the problem, buffers must be placed precisely to be effective.

Our current technique can accommodate newly added modules for which tentative initial placements are given. All a designer would need to do is place new modules roughly where they should go in the core, and ECO-system will find legal positions for them automatically. If new module locations are not known, they can be found with simple analytical techniques. Specifically, if an unplaced module is connected to several placed modules, an initial location for the module could be the average location of its neighbors. This does not work well, however, when a cluster of new logic is added to a design, especially in the presence of macros and obstacles. For this reason, we develop a technique to place unplaced modules within ECO-system.

To handle new modules separately, one must be able to detect them easily in a design. Some input formats allow the user to specify modules which are new with the keyword UNPLACED. For other input formats without such a keyword, ECO-system checks for modules that are placed outside of the core and

marks them as being unplaced. ECO-system also tests to see if several modules are placed at exactly the same location which could indicate a cluster of new logic. Modules placed in exactly the same location, such as a default location like (0,0), are also treated as unplaced.

In each bin, if a cut-line and partitioning are derived, unplaced modules are partitioned with a separate partitioning call to assign them to child bins. If the derived partitioning is not accepted, unplaced modules are combined with the old modules, and placement continues from scratch. In this way, unplaced modules will migrate to good legal locations automatically. As the locations for unplaced modules are chosen based on current locations of all the modules in the design, the final locations of unplaced modules will likely be better than ones that were chosen based on the initial placement.

If new modules are introduced into a design and a user defines a region of the placement to work in, there is some ambiguity in what ECO-system should do with unplaced modules. All unplaced modules could be placed inside the user-specified region, or ECO-system could determine which of the unplaced modules would best be placed in the region. Determining which of the unplaced modules belong in a user-specified rectangular region requires at most four calls to a partitioner (since the region can be carved out with four geometric cut-lines), so this will still be efficient. To avoid uncertainty, the user is allowed to specify which behavior is desired.

VI. Empirical Results

We implemented ECO-system in C++ and ran it on 3.2GHz Pentium Xeon machines. In this section we present results dealing with the legalization of benchmarks altered due to cell resizing, the effect of ECO-system on the timing of resized benchmarks, and using ECO-system to legalize various analytically-generated global placements.

A. Legalization of Resized Netlists

For testing we use three suites of benchmarks. The first suite of benchmarks are the ICCAD 2004 IBM-MSwPins benchmarks: mixed-size netlists with non-trivial macro sizes, aspect ratios and pin offsets [1]. We placed all of the benchmarks with Capo 10 [42] and chose the best of 2 runs. Next we randomly resized the standard cells of the benchmark to simulate cell sizing such that the total area of cells would remain relatively constant. Each standard cell of the design was randomly increased or decreased in size, but no cell was decreased below the minimum cell size or increased beyond the largest cell size.

The change in cell area and amount of overlap introduced by the resizing is shown in Table II. The resized benchmarks should have legal placements with HPWL near that of the original benchmarks since total cell area does not change appreciably. Discussions with colleagues in the industry point out that cell resizing is affected by a variety of factors, which are not as random as in our experiments. The IBM-MSwPins benchmarks do not contain enough information to perform more intelligent resizing, so this experiment is used primarily to evaluate ECO-system in the presence of many movable macros.

We compare ECO-system to the legalizer of Capo 10, and the results are summarized in Table II. The Capo legalizer runs quickly and produces legal placements, but it increases HPWL by 3.93% on average. ECO-system takes less than 16% of the original placement time, and only increases HPWL by 0.61% on average. By adding cut-line shifting to ECO-system runtime is largely unaffected but the HPWL increase is further reduced to 0.24%. We have also varied the amount of overlap introduced into these benchmarks by reducing the number of cells affected by our sizing. We find that HPWL is relatively unaffected (HPWL generally changes by less than 0.5%) by increasing amounts of overlap for these designs.

The second set of benchmarks are from the ISPD 2005 Placement Contest [35]. They are a standard cell benchmark suite with non-trivial fixed obstacles throughout the placement area [35]. We placed all of the benchmarks with APlace 2.04 [28] (the winning placer of the contest) and randomly resized the standard cells of the benchmark in the same way as the IBM-MSwPins benchmarks as the ISPD 2005 benchmarks do not contain necessary information for more intelligent resizing. As a result, the focus of this experiment is to see how ECO-system performs on very large-scale placement instances in the presence fixed obstacles.

The change in cell area and amount of overlap introduced by the resizing is shown in Table II. A comparison of ECO-system to the legalizer of Capo 10 is summarized in Table II. Full data for the ISPD'05 benchmarks can be found in [41]. The Capo legalizer runs 40% faster than ECO-system, but increases HPWL by 4.28% on average. ECO-system takes 14% of the original placement time, and *decreases* HPWL by 1.00%. Figure 7 depicts the benchmark adaptec3 before cell resizing and after legalization with ECO-system. ECO-system's placement is similar to the original APlace 2.04 placement and does not move the majority of cells far from their original locations. The average displacement per cell is 0.28% of the half-perimeter of the design which is an order of magnitude less than WSA's displacements [30], [31] and those reported in [4]. Only 1.98% of the cells have non-trivial displacements.

The third set of benchmarks on which we perform

TABLE II

OVERLAP LEGALIZATION ON THE IBM-MSWPINS [1] AND ISPD'05 CONTEST BENCHMARKS [35]. "AREA RATIO" REPRESENTS THE CHANGE IN TOTAL CELL AREA AFTER RESIZING. OVERLAP IS MEASURED AS % OF THE TOTAL MOVABLE CELL AND MACRO AREA. FULL DATA FOR THE ISPD'05 BENCHMARKS CAN BE FOUND IN [41]. ECO-SYSTEM REQUIRES SIGNIFICANTLY MORE RUNTIME THAN THE CAPO 10 LEGALIZER [39], AND APPROXIMATELY 16% OF THE ORIGINAL PLACEMENT TIME. ECO-SYSTEM INCREASES HPWL BY 0.61% ON AVERAGE WHILE THE CAPO 10 LEGALIZER INCREASES HPWL BY 3.93% ON THE IBM-MSWPINS BENCHMARKS. ON THE ISPD'05 CONTEST BENCHMARKS ECO-SYSTEM *decreases* HPWL BY 1.00% ON AVERAGE WHILE THE CAPO 10 LEGALIZER INCREASES HPWL BY 4.28%.

IBM-MSwPins	Area	Orig.	Orig.	Overlap	Capo 10 Legalizer [39]			ECO-system			ECO-system /w shifting		
Benchmarks	Ratio	Time (s)	HPWL		Time (s)	HPWL	Ratio	Time (s)	HPWL	Ratio	Time (s)	HPWL	Ratio
ibm01	0.9982	248	2.48	7.35%	1.27	2.57	1.0371	44.4	2.48	0.9995	45.2	**2.47**	0.9957
ibm02	1.0008	463	5.12	5.56%	2.15	5.28	1.0328	77.3	5.13	1.0024	81.0	**5.11**	0.9980
ibm03	1.0011	661	7.58	5.83%	15.9	7.99	1.0543	128	7.54	0.9951	127	**7.53**	0.9934
ibm04	0.9990	728	8.61	8.13%	11.3	9.03	1.0482	149	8.67	1.0070	147	**8.66**	1.0055
ibm05	1.0017	593	10.14	13.54%	0.13	10.25	1.0114	141	10.32	1.0177	149	**10.28**	1.0139
ibm06	1.0018	846	6.78	7.36%	10.5	7.10	1.0469	152	6.82	1.0054	155	**6.79**	1.0019
ibm07	0.9997	1213	11.63	9.61%	16.4	12.16	1.0455	201	11.72	1.0081	210	**11.69**	1.0052
ibm08	1.0029	1492	13.42	8.50%	7.36	13.73	1.0232	211	13.54	1.0090	223	**13.49**	1.0054
ibm09	1.0025	1492	14.96	8.14%	14.8	16.06	1.0732	288	14.89	0.9954	296	**14.82**	0.9907
ibm10	0.9997	2476	31.79	4.53%	119	32.62	1.0260	387	31.54	0.9922	390	**31.48**	0.9903
ibm11	0.9993	2067	21.43	8.48%	26.3	22.56	1.0529	384	21.63	1.0092	411	**21.44**	1.0005
ibm12	0.9996	2903	38.52	5.91%	50.6	39.20	1.0175	379	37.95	0.9851	393	**37.82**	0.9819
ibm13	1.0014	2667	27.30	7.94%	55.3	28.61	1.0478	586	27.57	1.0101	587	**27.31**	1.0004
ibm14	1.0002	4954	40.00	13.49%	38.3	41.67	1.0417	734	40.70	1.0174	744	**40.58**	1.0144
ibm15	1.0016	6241	53.72	10.85%	63.1	56.48	1.0514	1127	54.68	1.0178	996	**54.68**	1.0178
ibm16	0.9997	7232	61.12	9.19%	36.2	62.74	1.0264	890	61.42	1.0050	907	**61.20**	1.0014
ibm17	0.9987	7558	70.52	14.09%	36.0	73.09	1.0365	983	71.65	1.0160	1009	**71.45**	1.0132
ibm18	1.0017	6897	46.46	15.91%	13.7	48.11	1.0354	1006	47.30	1.0182	1032	**47.13**	1.0145
Average	1.0005	1.0000			0.0102		1.0393	0.1551		1.0061	0.1558		1.0024
Data for ISPD'05 Benchmarks from [41]													
Average	1.0004	1.0000			0.0415		1.0428	0.1234		0.9899	0.1138		0.9920

TABLE III

OVERLAP LEGALIZATION ON THE IWLS 2005 BENCHMARKS [24]. "AREA RATIO" REPRESENTS THE CHANGE IN TOTAL CELL AREA AFTER RESIZING. OVERLAP IS MEASURED AS % OF THE TOTAL MOVABLE CELL AREA. ECO-SYSTEM DECREASES HPWL BY 1.81% ON AVERAGE WHILE THE CAPO 10 LEGALIZER INCREASES HPWL BY 1.85%.

IWLS	Area	Orig.	Orig.	Overlap	Capo 10 Legalizer [39]			ECO-system		
Benchmarks	Ratio	Time (s)	HPWL		Time (s)	HPWL	Ratio	Time (s)	HPWL	Ratio
aes_core	1.0278	519	23.70	14.30%	0.2	23.91	1.0089	64.4	**22.94**	0.9679
ethernet	1.1122	3666	105.71	13.34%	0.5	108.73	1.0286	284	**104.78**	0.9912
mem_ctrl	1.0508	404	16.29	13.24%	0.1	16.63	1.0209	32.6	**15.95**	0.9791
pci_bridge32	0.9724	550	19.61	11.27%	0.2	20.09	1.0245	55.8	**19.21**	0.9796
usb_funct	1.0901	346	15.93	13.82%	0.1	16.34	1.0257	39.3	**15.72**	0.9868
vga_lcd	0.9841	15686	370.79	9.06%	1.1	371.76	1.0026	819	**365.87**	0.9867
Average	1.0383	1.0000			0.0001		1.0185	0.0612		0.9819

experiments with resizing is the IWLS 2005 suite of benchmarks [24]. These benchmarks contain information such as the signal directions of pins, so we were able to resize cells in a more realistic way based on wire load. The benchmarks were first placed using Capo 10 in ROOSTER mode [40] for routability. Next, for each cell we calculated the Steiner length of wires the cell drives. According to the theory of logic effort, longer wires should be driven by larger cells [44], so we increased the sizes of cells whose driven lengths were longer than the median driven length and decreased the size of cells whose driven lengths were shorter than the median. The amount of overlap introduced by this resizing method is shown in Table III. We compare ECO-system to the Capo 10 legalizer and find again that the Capo 10 legalizer is extremely fast, but increases HPWL significantly (1.85%) while ECO-system is able to *reduce* HPWL by 1.81% on average. For this experiment we did not use ECO-system's cut-line shifting feature in order to preserve Capo's routability-driven whitespace allocation.

B. ECO-system's Impact on Timing

One of the most important goals of an incremental placer is to preserve the timing characteristics of a design after timing optimizations have been performed on the design. Recall that cell sizing and buffer insertion decisions are based on circuit timing. If an incremental placer moves cells too drastically, popular timing optimizations can be less effective and eventu-

TABLE IV

OVERLAP LEGALIZATION OF APLACE 2.04'S [28] GLOBAL PLACEMENTS OF THE ISPD'05 CONTEST BENCHMARKS [35]. OVERLAP IS MEASURED AS % OF THE TOTAL MOVABLE CELL AREA. ECO-SYSTEM PRODUCES LEGAL SOLUTIONS WITH NEARLY THE SAME OR BETTER HPWL THAN APLACE 2.04'S LEGALIZER. APLACE'S LEGALIZER INCREASES HPWL BY 4.91% WHILE ECO-SYSTEM INCREASES HPWL BY 3.68% AND ONLY 2.35% WHEN USING SHIFTING. ECO-SYSTEM WITH SHIFTING IS FASTER ON 7 OF THE 8 BENCHMARKS AND FOUR TIMES FASTER THAN APLACE'S LEGALIZER OVERALL.

Benchmark	Orig.	Illegal	Overlap	APlace 2.04 Legalizer [28]			ECO-system			ECO-system /w shifting		
	Time (s)	HPWL		Time (s)	HPWL	Ratio	Time (s)	HPWL	Ratio	Time (s)	HPWL	Ratio
adaptec1	7569	81.05	34.74%	1346	83.87	1.0348	1656	85.17	1.0508	1386	**82.23**	1.0146
adaptec2	6062	94.22	47.25%	2543	101.64	1.0788	2037	101.10	1.0730	1684	**97.85**	1.0385
adaptec3	15849	211.13	47.12%	11495	231.17	1.0949	4245	227.25	1.0763	3672	**222.24**	1.0526
adaptec4	15404	197.24	36.78%	15271	206.23	1.0456	3805	202.26	1.0255	3505	**200.80**	1.0180
bigblue1	8265	100.51	28.53%	2486	**101.96**	1.0144	1607	104.22	1.0369	1262	102.50	1.0198
bigblue2	13650	154.51	30.15%	14252	159.08	1.0296	3882	156.35	1.0119	3840	**155.83**	1.0086
bigblue3	30624	385.40	41.06%	38873	414.29	1.0750	12546	**386.99**	1.0041	10080	395.11	1.0252
bigblue4	61932	865.03	32.01%	56809	884.39	1.0224	11552	880.58	1.0180	10451	**874.90**	1.0114
Average	1.0000			0.8978		1.0491	0.2594		1.0368	0.2252		1.0235

ally degrade timing rather than improve it. Therefore, we evaluate the impact of ECO-system on circuit timing. For these experiments we resized the 20 of the OpenCores designs that were part of the IWLS 2005 benchmark suite [24] in a realistic manner (as described above) and evaluate timing characteristics of the resized netlists before and after legalization by ECO-system.

Circuit timing was evaluated using a Static Timing Analysis engine which uses the D2M net delay model (more accurate then Elmore) [6] for each net based on Steiner trees produced by the FLUTE package [14]. The worst change in circuit delay for these designs was an increase of 8.07%. The average change was 1.00% while the best was a decrease of 7.37% of maximum delay. Thus ECO-system is effective in preserving the timing of a netlist by minimally impacting maximum delay during legalization and in some cases can further improve maximum delay. In this experiment ECO-system is completely independent of the timing analyzer used and therefore our results are likely to hold for other STA engines as well.

C. Legalizing Analytical Global Placements

Analytical placements generally contain a significant amount of overlap after global placement, especially so on the ISPD'05 Contest benchmarks given their numerous fixed obstacles in the core region. As such, we compare ECO-system to the APlace 2.04 legalizer on APlace 2.04 global placements on the ISPD'05 Contest benchmarks. Table IV shows that APlace 2.04 global placements have overlap of approximately 30% or more. APlace 2.04's legalizer generally increases HPWL by 4.91% while ECO-system increases HPWL only 3.68% on average. ECO-system is also three times faster than APlace's legalizer. Adding cut-line shifting improves ECO-system's results, increasing HPWL by only 2.35% while running four times faster than APlace's legalizer.

To illustrate the effectiveness of ECO-system in redistributing whitespace to improve routability, we placed the IBMv2 benchmark suite [47] with the analytical placer mPL6 [10]. mPL6 global placements were refined by ECO-system and then routed using Cadence WarpRoute. In Table V we compare the placements refined by ECO-system to those produced by the detail placer of mPL6 (XDP [17]) in terms of routed wirelength (Rt WL), via counts, violations, and routing time (Rt Time). ECO-system improves mPL6 global placements to the point where the router completes in all cases, reducing routed wirelength by 1.1%, via counts by 7.8% and routing time by more than half on average.

Lastly, to test the ECO-system's routability improvements in the presence of fixed obstacles, we test on the ICCAD'04-Faraday benchmarks [1]. The Faraday benchmarks are a suite of mixed-size benchmarks with routing information based on netlists released by the Faraday Corporation [1]. For our experiments, we fix macros to their original locations as determined by Silicon Ensemble Ultra v5.4.126 (details on the construction of the benchmarks can be found in Appendix A of [1]). We run mPL6 on the four benchmarks with macros and produce global and detail placements of each. As in the previous experiment, we compare mPL6 detail placements to mPL6 global placements refined by ECO-system. Results for this experiment are shown in Table VI. ECO-system placements are mostly routable with a few violations, but the mPL6 placements are completely unroutable. We were unable to run the WSA technique on the mPL6 placements as WSA does not support fixed obstacles, which we have confirmed with the authors of WSA.

VII. CONCLUSIONS

Below we summarize our work, outline several additional applications and articulate our contributions to shared infrastructure for research in placement.

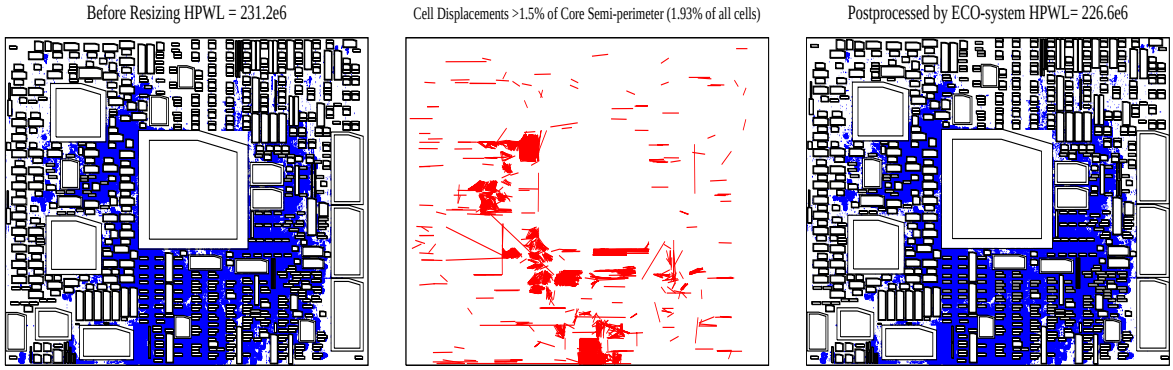

Fig. 7. When applied to resized netlist, ECO-system produces a placement (right) similar to the original placement (left). Fixed objects are outlined in double black lines. The largest cell displacements are shown in red (center). Only displacements larger than 1.5% of the half-perimeter of the design are shown. Average displacement is 0.28% of the design half-perimeter. The majority of the large displacements form around the corners of the large, fixed obstacles. Many of these large displacements appear to be clustered, indicating small groups of modules transported to another region of the core or spread to accommodate area increases.

TABLE V

IMPROVING THE ROUTABILITY OF ANALYTICAL PLACEMENTS USING ECO-SYSTEM. WE COMPARE THE ROUTABILITY OF MPL6 [10] GLOBAL PLACEMENTS WHEN USING MPL6'S DETAIL PLACER (XDP [17]) VS. ECO-SYSTEM WITH CUT-LINE SHIFTING FOR DETAIL PLACEMENT ON THE IBMV2 BENCHMARK SUITE [47]. BEST LEGAL ROUTED WIRELENGTH (RT WL) AND VIA COUNTS ARE HIGHLIGHTED IN BOLD. ECO-SYSTEM PRODUCES ROUTABLE PLACEMENTS IN ALL CASES, REDUCES ROUTED WIRELENGTH BY 1.1%, REDUCES VIA COUNTS BY 7.8%, AND CUTS ROUTING RUNTIME BY MORE THAN HALF ON AVERAGE.

Benchmark	XDP [17]				ECO-system			
	Rt WL	Vias	Viols.	Rt Time (m)	Rt WL	Vias	Viols.	Rt Time (m)
ibm01e	723961	150166	806	1052	**745660**	**125177**	0	22
ibm01h	735409	156414	348	654	**701959**	**122995**	0	70
ibm02e	1937102	261495	0	27	**1822638**	**247396**	0	13
ibm02h	2004969	324609	108	133	**1933310**	**255647**	0	18
ibm07e	3817994	497500	0	54	**3555210**	**468105**	0	22
ibm07h	3814735	569897	49	91	**3658097**	**479911**	0	25
ibm08e	3999658	587627	0	31	**3970074**	**561636**	0	24
ibm08h	3948739	591744	0	35	**3914580**	**574135**	0	28
ibm09e	**2891305**	483046	0	17	2956856	**472863**	0	17
ibm09h	**2935006**	490682	0	19	2965823	**480363**	0	18
ibm10e	**5753519**	773695	0	36	5888185	**750270**	0	30
ibm10h	**5742241**	778756	0	35	5762900	**759962**	0	31
ibm11e	**4399838**	637627	0	26	4438438	**615691**	0	23
ibm11h	4670094	645872	0	31	**4634023**	**630791**	0	25
ibm12e	**8640070**	972714	0	66	8697654	**908164**	0	42
ibm12h	**8695922**	977498	0	69	8726583	**926119**	0	53
Ratio	1.000	1.000		1.000	0.989	0.922		0.446

A. *Summary of our work*

Our main contribution is ECO-system — an algorithmic framework designed to interface a wide variety of circuit optimizations with their physical environment. This framework offers, for the first time in the literature, a strong and robust legalizer that can handle a broad range of modern placement instances with movable macros, fixed obstacles, etc. ECO-system automatically focuses on regions of the layout and sections of the netlist that require changes, and performs optimization of adequate strength in each case. ECO-system can be combined with an external global placer invoked when particularly large changes are required. It can also be used in incremental re-synthesis, in high-level and physical synthesis optimizations, and several other contexts.

ECO-system includes all detail placement methods implemented in Capo [36], [39], [40], [42], and can similarly be grafted onto other top-down placers, such as BonnPlace [46], PolarBear [16] or NTUPlace [25], by performing a one-pass Fiduccia-Mattheyses test. ECO-system can act like the WSA technique [30], and can invoke any black-box global placement algorithm when it decides that a particular bin must be replaced from scratch.

The definitive success of ECO-system in legalizing APlace and mPL6 global placements (Tables II, IV, V and VI) allows one to answer a long-standing question in placement — whether the slicing structure of min-cut placements costs them HPWL. Given that the placements produced by ECO-system are largely slicing, the answer appears negative for standard-cell

TABLE VI

IMPROVING THE ROUTABILITY OF ANALYTICAL PLACEMENTS IN THE PRESENCE OF FIXED OBSTACLES IN THE ISPD'04-FARADAY BENCHMARK SUITE [1]. WE POSTPROCESS MPL6 [10] GLOBAL PLACEMENTS USING MPL6'S DETAIL PLACER AND, SEPARATELY, OUR ECO-SYSTEM (WITH CUT-LINE SHIFTING). THE MPL6 DETAIL PLACER XDP [17] PRODUCES LARGELY UNROUTABLE PLACEMENTS.

Benchmark	XDP [17]				ECO-system			
	Rt WL	Vias	Viols.	Rt Time (m)	Rt WL	Vias	Viols.	Rt Time (m)
dsp1	1041556	233408	112883	12	1162096	202700	0	6
dsp2	-	-	-	>24 hrs.	1117349	201598	0	6
risc1	2042695	342856	373088	71	2066426	344258	10	10
risc2	-	-	-	>24 hrs.	1906434	337809	11	11

placement, but is likely to be positive when large macros are involved, as suggested by results in [36].

We have analyzed requirements for an ECO placement tool and implemented an interface based on ECO-system applicable to high-level and physical synthesis, allowing the designer to add and remove nets and cells from a design, reallocate whitespace and large macros (Figure 1), resize cells and re-weight nets while retaining control of the amount of change performed by ECO-system.

B. Additional applications

As ECO-system subsumes and generalizes the WSA technique [30], [31] and outperforms the technique from [4], ECO-system can also be used for the same applications. In addition to our experiments that demonstrate improvement in routability and support for gate sizing, ECO-system can be used to support buffer insertion in physical synthesis and floorplan resizing during chip planning [31].

Another relevant application of ECO-system lies in fault-tolerant reconfigurable computing. In this paradigm, the digital system periodically invokes built-in self-test and may identify components that recently failed. To avoid using faulty components, the system can be reconfigured to use only those resources that remain operational.

ECO-system could be used to quickly reprogram faulty chips in the following way. Obstacles are placed in those areas of a circuit that have been determined to have errors. ECO-system can be run on this modified design to remove all overlaps between the old placement and the new fixed objects. The legalized placement would then be free of errors as none of the faulty parts would be used in the replacement. ECO-system uses as much of the original placement as possible so timing and other relevant circuit properties would likely be preserved.

Algorithms used in ECO-system can also be used to geometrically partition a layout so as to minimize interconnect between partitions, as shown in Figure 3. With minimal communication between partitions, physical design algorithms that are generally run after placement (such as cell sizing, routing or buffer insertion) can be parallelized, improving runtime on multiprocessor systems. In particular, it has been shown that post-placement optimizations for timing can be parallelized [29]. Empirical results show that runtime can be decreased by up to five times when running on a parallel machine with eight processors [29].

C. Our contributions to shared research infrastructure

All algorithms reported in this work will be available to the research community in source code form, integrated into the Capo placer — an established software distribution used by over two hundred people and requested several times during an average week in Fall 2006. The availability of ECO-system in this work will significantly lower barriers for entry in two research directions: (a) global placement, (b) physical synthesis. Indeed, work in global placement has always been complicated by the need to produce legal, routable placements, but with the availability of a fast and reliable legalizer it becomes easy to evaluate new global placement techniques without a significant infrastructure investment. Similarly, our software allows one to experiment with physical synthesis optimizations (e.g., sizing, buffering) and placement-driven logic transformations (e.g., fanout optimization) while delegating legalization to ECO-system.

Acknowledgements. This work was partially supported by the Gigascale Silicon Research Center (GSRC), the National Science Foundation (NSF), and the Horace H. Rackham School of Graduate Studies at the University of Michigan.

REFERENCES

[1] S. N. Adya, S. Chaturvedi, J. A. Roy, D. A. Papa and I. L. Markov, "Unification of Partitioning, Placement and Floorplanning," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 550-557, 2004.

[2] S. N. Adya and I. L. Markov, "Consistent Placement of Macro-blocks Using Floorplanning and Standard-Cell Placement", in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 12-17, 2002.

[3] S. N. Adya and I. L. Markov,"Fixed-outline Floorplanning: Enabling Hierarchical Design", *IEEE Trans. Very Large Scale Integr.*, vol. 11, no. 6, pp. 1120-1135, 2003. See also (*Proc. Int. Conf. Computer Design (ICCD)*, pp. 328-334, 2001).

[4] S. N. Adya, I. L. Markov and P. G. Villarrubia, "On Whitespace and Stability in Mixed-Size Placement," *Integration: the VLSI Journal*, vol. 39, no. 4, pp. 340-362, 2006. See also (*Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 311-318, 2003).

[5] A. Agnihotri et al., "Mixed Block Placement via Fractional Cut Recursive Bisection," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 5, pp 748-761, 2005. See also (*Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 307-310, 2003).

[6] C. J. Alpert, A. Devgan and C. Kashyap, "A Two Moment RC Delay Metric for Performance Optimization," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 69-74, 2000.

[7] C. J. Alpert, G.-J. Nam, P. Villarrubia and M. C. Yildiz, "Placement Stability Metrics," in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, pp. 1144-1147, 2005.

[8] U. Brenner and J. Vygen, "Legalizing a Placement With Minimum Total Movement," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 12, pp. 1597-1613, 2004. See also (*Proc. Int. Symp. Physical Design (ISPD)*, pp. 2-9, 2004).

[9] A. E. Caldwell, A. B. Kahng and I. L. Markov, "Optimal Partitioners and End-case Placers for Standard-cell Layout," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 19, no. 11, pp. 1304-1314, 2000. See also (*Proc. Int. Symp. Physical Design (ISPD)*, pp. 90-96, 1999).

[10] T. F. Chan, J. Cong, J. R. Shinnerl, K. Sze and M. Xie, "mPL6: Enhanced Multilevel Mixed-size Placement," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 212-214, 2006.

[11] C.-C. Chang, J. Cong, D. Pan and X. Yuan, "Multilevel Global Placement with Congestion Control," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 22, no. 4, pp. 395-409, 2003.

[12] C.-C. Chang, J. Cong and X. Yuan, "Multi-Level Placement for Large-Scale Mixed-Size IC Designs," in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, pp. 325-330, 2003.

[13] T. C. Chen, Y. W. Chang and S. C. Lin, "IMF: Interconnect-Driven Multilevel Floorplanning for Large-Scale Building-Module Designs," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 159-164, 2005.

[14] C. Chu and Y.-C. Wong, "Fast and Accurate Rectilinear Steiner Minimal Tree Algorithm for VLSI Design," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 28-35, 2005. `http://class.ee.iastate.edu/cnchu/flute.html`

[15] J. Cong and M. Sarrafzadeh, "Incremental Physical Design", in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 84-92, 2000.

[16] J. Cong, M. Romesis and J. Shinnerl, "Robust Mixed-Size Placement Under Tight White-Space Constraints," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 165-173, 2005.

[17] J. Cong and M. Xie, "A Robust Detailed Placement for Mixed-Size IC Designs," in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, pp. 188-194, 2006.

[18] K. Doll, F. M. Johannes and K. J. Antreich, "Iterative Placement Improvement By Network Flow Methods," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 10, pp. 1189-1200, Oct. 1994.

[19] W. Donath et al., "Transformational Placement and Synthesis", in *Proc. Design and Test in Europe (DATE)*, pp. 194-201, 2000.

[20] C. M. Fiduccia and R. M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," in *Proc. Design Automation Conf. (DAC)*, pp. 175-181, June 1982.

[21] R. Goering, "Cadence CTO: CAD 'Foundations' Must Change," EETimes, April 11, 2006, `http://www.eetimes.com/showArticle.jhtml?articleID=185300099`

[22] D. Hill, "Method and System for High Speed Detailed Placement of Cells Within an Integrated Circuit Design," *U.S. Patent 6370673*, April 2002.

[23] S. W. Hur and J. Lillis, "Mongrel: Hybrid Techniques for Standard Cell Placement," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 165-170, 2000.

[24] IWLS 2005 Benchmarks, `http://iwls.org/iwls2005/benchmarks.html`

[25] Z.-W. Jiang et al., "NTUPlace2: A Hybrid Placer Using Partitioning and Analytical Techniques," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 215-217, 2006.

[26] A. B. Kahng and S. Mantik, "On Mismatches Between Incremental Optimizers and Instance Perturbations in Physical Design Tools," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 17-22, 2000.

[27] A. B. Kahng and S. Reda, "Evaluation of Placer Suboptimality Via Zero-Change Netlist Transformations," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 208-215, 2005.

[28] A. B. Kahng and Q. Wang, "Implementation and Extensibility of an Analytic Placer," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 25, no. 5, pp. 734-747, May 2005.

[29] J. Kim, M. C. Papaefthymiou and J. L. Neves, "Parallelizing Post-Placement Timing Optimization," in *Proc IEEE Int. Parallel and Distributed Processing Symp. (IPDPS)* 2006.

[30] C. Li, M. Xie, C. K. Koh, J. Cong and P. H. Madden, "Routability-driven Placement and White Space Allocation," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 394-401, 2004.

[31] C. Li, C.-K. Koh and P. H. Madden, "Floorplan Management: Incremental Placement for Gate Sizing and Buffer Insertion," in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, pp. 349-354, January 2005.

[32] L. Luo, Q. Zhou, X. Hong and H. Zhou, "Multi-stage Detailed Placement Algorithm for Large-Scale Mixed-Mode Layout Design," in *Proc. Int. Conf. Computational Science and Apps. (ICCSA)*, pp. 896-905, 2005.

[33] T. Luo, H. Ren, C. J. Alpert and D. Pan, "Computational Geometry Based Placement Migration," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 41-47, 2005.

[34] M. D. Moffitt, A. N. Ng, I. L. Markov and M. E. Pollack, "Constraint-driven Floorplan Repair," in *Proc. Design Automation Conf. (DAC)*, pp. 1103-1108, 2006.

[35] G.-J. Nam, C. J. Alpert, P. Villarrubia, B. Winter and M. Yildiz, "The ISPD2005 Placement Contest and Benchmark Suite," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 216-220, 2005.

[36] A. N. Ng, I. Markov, R. Aggarwal and V. Ramachandran, "Solving Hard Instances of Floorplacement," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 170-177, April 2006.

[37] M. Pan, N. Viswanathan and C. Chu, "An Efficient and Effective Detailed Placement Algorithm," in *Proc. Int. Conf. Computer-Aided Design (ICCAD)*, pp. 48-55, 2005.

[38] H. Ren, D. Z. Pan, C. J. Alpert and P.Villarrubia, "Diffusion-based Placement Migration," in *Proc. Design Automation Conf. (DAC)*, pp. 515-520, 2005.

[39] J. A. Roy, S. N. Adya, D. A. Papa and I. L. Markov, "Min-cut Floorplacement," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 25, no. 7, pp. 1313-1326, 2006.

[40] J. A. Roy, J. F. Lu and I. L. Markov, "Seeing the Forest and the Trees: Steiner Wirelength Optimization in Placement," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 78-85, 2006.

[41] J. A. Roy and I. L. Markov, "ECO-system: Embracing the Change in Placement," To appear in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, 2007.

[42] J. A. Roy, D. A. Papa, A. N. Ng, I. L Markov, "Satisfying Whitespace Requirements in Top-down Placement," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 206-208, 2006.

[43] N. Selvakkumaran and G. Karypis, "Theto - A Fast, Scalable and High-quality Partitioning Driven Placement Tool," Technical report, Univ. of Minnesota, 2004.

[44] P. V. Srinivas, M. Borah and P. Buch, "System and Method for Estimating Capacitance of Wires Based on Congestion Information," *U.S. Patent 6519745*, Feb. 11, 2003.

[45] N. Viswanathan, M. Pan and C. Chu, "FastPlace 2.0: An Efficient Analytical Placer for Mixed-Mode Designs," in *Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC)*, pp. 195-200, 2006.

[46] J. Vygen, "Algorithms for Large-Scale Flat Placement," in *Proc. Design Automation Conf. (DAC)*, pp. 746-751, 1997.

[47] X. Yang, B. K. Choi, and M. Sarrafzadeh, "Routability Driven White Space Allocation for Fixed-die Standard-cell Placement," in *Proc. Int. Symp. Physical Design (ISPD)*, pp. 42-49, 2002.