CRISP: Congestion Reduction by Iterated Spreading during Placement

Jarrod A. Roy[†‡], Natarajan Viswanathan[‡], Gi-Joon Nam[‡], Charles J. Alpert[‡] and Igor L. Markov[†]
[†]University of Michigan, Department of EECS, 2260 Hayward Ave., Ann Arbor, MI 48109
[‡]IBM Austin Research Laboratory, 11501 Burnet Road, Austin, TX 78758
{royj,imarkov}@eecs.umich.edu, {jaroy,nviswan,gnam,alpert}@ibm.com

ABSTRACT

Dramatic progress has been made in algorithms for placement and routing over the last 5 years, with improvements in both speed and quality. Combining placement and routing into a joint optimization has also been proposed. However, it remains unclear if the benefits would be significant enough to justify major changes in commercial tools. CRISP addresses this challenge and is the first tool to demonstrate tangible benefits of combined place-and-route optimization including fewer global routing detours, reduced detailed routing violations and runtime, and even shrinking the floorplan of a commercial design. We employ fast global routing to choose standard cells to temporarily inflate and iteratively spread for congestion reduction. Spreading only in congested regions, we enable die area reduction by facilitating routing with high area utilization.

1. INTRODUCTION

Physical design has been a major bottleneck in modern EDA flows, and its significance is increasing for large ICs due to the poor scaling of interconnect delay relative to transistor delay. The focus of our work is on large ASIC and SoC designs that contain millions of standard cells and thousands of macro blocks. Critical path delay in such designs is often dominated by interconnect, and cell locations affect circuit delay to a large extent, followed by the routes chosen for the longest wires. These degrees of freedom correspond to *placement* and *routing*, which have traditionally been handled by independently-developed EDA tools. These tasks have received a great amount of attention from researchers, and the contests organized by IBM at ISPD confirmed dramatic improvements in the speed and quality of placement and routing on large industry netlists by university researchers [9]. Not only have the basic algorithms changed in the last 5-10 years, but the very landscape of placement and routing has changed dramatically. Perhaps, the most visible difference from older ASICs is the presence of large amounts of *whitespace*, inserted to support net buffering, gate sizing, post-placement logic restructuring and ECOs, as well as limit power density and ease violation-free routing. Another major feature of large modern ASICs and SoCs is the presence of large fixed macros and induced routing obstacles.

Design objectives. Most work on congestion mitigation in placement cites as its primary motivation the need to facilitate violation-free routing and decrease turn-around time by reducing design iterations. Publications also point out that interconnect length must not increase too much while routability improves [1, Chapter 22]. While this trade-off is important to our work, we point out that additional motivating factors can be more critical to success in an industry environment. Indeed, total interconnect length is rarely a goal in itself, but is rather viewed as a proxy for circuit delay and dynamic power. In particular, timing-driven placement algorithms can be improved by early delay estimates, whose accuracy has a direct impact on the quality of delay optimization. Such estimates use pre-routes for individual nets to account for the capacitance of Steiner trees and other effects. When congestion estimates are available, pre-routes can be constructed to avoid congested regions, improving delay estimates. Constructing pre-routes independently per net often results in several pre-routes occupying the same track, especially near obstacles. Therefore pre-routes should be generated by a global router invoked within placement.

Chip area considerations. An important new consideration in our work is related to chip size. By working with entire ASICs and SoCs, rather than isolated partitions, one can control the area and the shape of the chip. In our case, chip floorplans are created by expert design engineers who account for many factors, including the whitespace required for routability and the placement of fixed macro blocks. Experiments discussed in Section 4 demonstrate a strong place-and-route tool that can handle high area utilization and thus requires less whitespace in the floorplan. To help extract maximum benefits from such a tool, expert design engineers working with us produced alternative floorplans with smaller area, and we demonstrate that such floorplans can be satisfied in terms of violation-free routing and timing closure. Decreasing chip area by 5% increases the number of chips in one wafer, reducing the cost of each chip [1, Part VII].

Compatibility with existing EDA infrastructure. On the algorithmic side, we are working with a state-of-the-art timing-driven force-directed placement framework and enhance it by accounting for congestion and routability. Unlike previous published work on combined placement and routing [5], we evaluate these steps within a complete industrial physical-synthesis flow. Our contributions include a new technique for interconnect estimation, the use of incremental cell inflation, and the use of dedicated legalization and detail placement. Dedicated steps are needed due to the requirement to preserve timing and the validity of neighboring optimizations.

Key contributions of our work include:

- We present CRISP, a technique which improves the routability of a given placement through highly accurate congestion modeling and a novel measure-and-improve placement flow.
- We formulate placement pin-density constraints and propose a separate placement-spreading pass to enforce them, applied between global and detailed routing. This pass improves

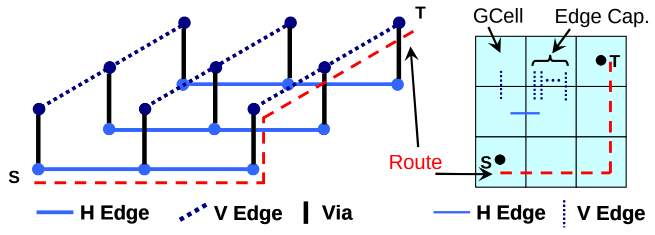

Figure 1: 3-d and 2-d routing grids with grid cells, also known as *GCells*, routing edges and routing tracks labeled.

detail routing in our experiments while preserving global routes. This work is the first to separately measure and optimize global and detail routability in placement.

- We apply CRISP to layouts produced by mPL6 [3], and improve NTHU-Route 2.0 [4] via counts by 8.7%, global routed wirelength by 6.5% and detouring by 5.3%.
- We use CRISP in an industrial physical-synthesis flow to make previously unroutable designs routable, reducing detail routing runtime, detours and violations.
- We illustrate that effective congestion reduction can be applied to shrink the die size of a commercial design by 5%. Thus, our algorithms lead to savings in manufacturing cost.

The remainder of the paper is organized as follows. Section 2 provides relevant background. Section 3 describes our proposed techniques: Congestion Reduction by Iterated Spreading during Placement (CRISP). Section 4 validates CRISP in experiments on publicly available benchmarks with academic tools as well as large commercial ICs. Section 5 summarizes our work and concludes.

2. BACKGROUND & PREVIOUS WORK

Global and detailed routing. Routing is traditionally divided into two independent steps: *global* and *detailed routing*. At the beginning of global routing, the design is abstracted into a regular 3-d *routing grid* with one layer for each layer of metal in the design. Grid cells, or *GCells*, on the same layer are connected with *routing edges*. An example routing grid is shown in Figure 1. Each GCell is connected to at most four neighbors on the same layer in the four cardinal directions. GCells on different layers are connected through *via* edges. A GCell may be connected by vias to at most two neighbors, one layer above and one layer below. The router assigns a capacity to each routing edge which represents the amount of wire or blockage metal which is allowed to pass between one GCell and its neighbor. We define the *congestion* of a routing edge as the ratio of the metal assigned to it to its capacity. A routing edge is *congested* if more metal that its capacity is assigned to it. *Overflow* is defined as difference between routing edge usage and capacity summed over all congested routing edges.

During global routing, the pins of each net are binned into GCells, and the global router is must connect all nets with metal through routing and via edges while satisfying routing edge capacity constraints. Nets which are subsumed by a GCell are ignored by academic global routers and the ISPD routing contests, but are generally accounted for by industrial tools. Detail routing begins with a global routing solution, which may not satisfy all capacity constraints, as a guide, and assigns wires for all nets to actual routing tracks. Usually an industrial detail router can handle a small number of global routing capacity constraint violations and produce a routing solution with all wires connected (having no *opens*) and no metal from different nets connected (no *shorts*).

Congestion-driven placement. University research achieved spectacular breakthroughs due to its focus on point-tools. However, this approach also has limitations. For example, it is well-known that improving the half-perimeter wirelength (HPWL) targeted by most placers sometimes complicates the work of routers and results in unroutable placements [2, 8, 14]. To this end, the placers that performed best at the ISPD 2006 contest — NTUplace and KraftWerk2 — have recently been enhanced to mitigate congestion [7, 11] blending quick congestion estimates into the objective function. The unnamed technique presented in [12] estimates wiring density without using a router (and thus does not estimate the impact of detouring) and incorporates minimization of these estimates within an analytical placement engine. A related extension to FastPlace [5] goes further and embeds a fast global router into the placement loop. It demonstrates that the same router produces shorter routes starting from enhanced FastPlace placements. BonnPlace [2] presented temporary standard cell inflation for use during partitioning-based placement. Each cell was inflated based on a combination of the magnitude of the congestion estimated at its location, the number of pins on the cell and a user-specified constant. After each call to partitioning, BonnPlace estimates congestion, inflates cells in all over-congested regions, and then re-partitions the design [2]. BonnPlace techniques can only be applied during top-down placement rather than after; it is unclear how to apply them during analytical placement, the dominant placement paradigm.

These approaches are quite different, and the use of congestion estimates is much easier to implement. Also, the evidence in [2, 7, 11, 12] is more convincing because it involves complete commercial routers and confirms violation-free completion. However, our experience with large industry ICs suggests that congestion estimates around obstacles and blockages are often very inaccurate. This observation does not conflict with results in [2, 7, 11, 12] because the benchmarks used there do not contain significant blockages and have artificially-generated routing information. While routing congestion is known to impact circuit timing, these effects were not discussed in previous academic publications.

Placement spreading. Spreading techniques are common in the literature for analytical placement algorithms. One such technique is iterative local refinement (ILR) used by FastPlace [13]. ILR creates a regular grid for a given placement and performs many rounds of movement for every cell in a design. During each round, each movable cell is examined once. A cell may move from its current grid tile to one of its eight neighboring grid tiles. The choice of destination for each cell is based on a cost function which is a linear combination of the change in wirelength and area balance between the source and destination tiles caused by the move.

3. CRISP TECHNIQUES

We seek an incremental technique which can be applied to any placement in a physical-design flow to reduce congestion as well as preserve timing characteristics. To this end, we present CRISP, a technique which reduces routing congestion by incremental placement changes based on highly accurate congestion metrics. CRISP assembles state-of-the-art placement and global routing techniques into a new incremental placement flow, adding several missing pieces. CRISP carefully spreads standard cells located in congestion hot-spots while preserving the placement of uncongested areas where possible. If spreading creates new congested areas, CRISP iterations identify and eliminate them. In this section, we describe the techniques used by CRISP to model routing congestion, spread the placement and dissolve congested spots.

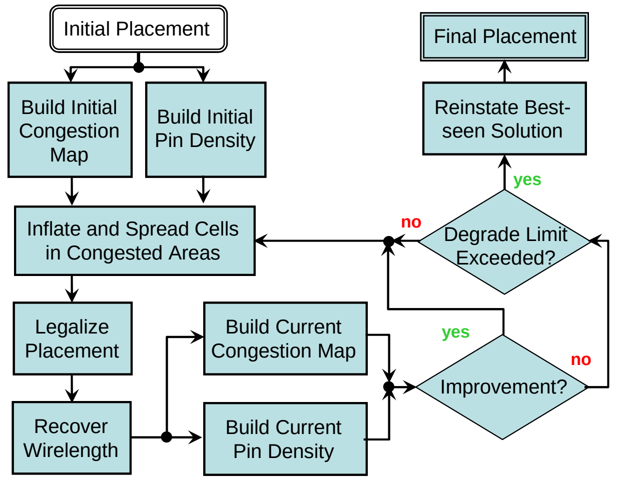
Initial Placement
Final Placement
Build Initial Congestion Map
Build Initial Pin Density
Reinstate Best-seen Solution
yes
no
Inflate and Spread Cells in Congested Areas
Degrade Limit Exceeded?
Legalize Placement
Build Current Congestion Map
yes
no
Improvement?
Recover Wirelength
Build Current Pin Density

Figure 2: The CRISP incremental placement flow.

3.1 Modeling routing congestion

Rather than build a probabilistic congestion map, CRISP creates a global routing instance from the current placement and uses a global router to generate a full set of routes. Prior to the ISPD 2007, 2008 routing contests [9], most publications assumed that global routing was too expensive to invoke during placement. However, we demonstrate that recent advances in global routing algorithms make them affordable as estimators. To keep routing runtime practical, CRISP limits the amount of detouring the global router is allowed to perform. This allows CRISP to capture the areas of the design which have actual rather than estimated routing congestion as well as identify areas of congestion which can be caused by detouring. An example of a congestion map derived from an academic global router is shown in Figure 5 (center), and a congestion map from an industry global router is shown in Figure 6.

Accounting for pin density. Local peaks of pin density often cause routing congestion, but are overlooked as a source of congestion by many algorithms. Global routing accurately captures the wires that pass between routing edges, but does not focus on congestion internal to global routing grid cells. In fact, this source of congestion is ignored completely by academic global routing formulations, and, in our experience is underestimated by industry global routers. A design may appear to be easily globally routable, but may fail detail routing leaving several shorts and opens. We propose to handle pin density directly with a separate pass of placement-spreading driven by pin density applied before detailed routing.

A naive method to mitigate the inaccuracy of global routing with respect to pin density is to shrink GCell sizes so that fewer nets become subsumed by GCells, but this can make global routing too slow for practical use. Our solution assimilates and extends ideas from previous work [2, 10], which injects whitespace into areas of high pin density or routing congestion. Novel elements include (i) exact requirements for determining when a region has too many pins and (ii) a separate placement-spreading pass driven by pin density, applied before detail routing. The latter seeks to ease detail routing while preserving global routes.

3.2 Temporary cell inflation

The technique of cell inflation is used by experienced designers to alleviate routing congestion, and it proves to be very effective in practice [2, 10]. For this reason, we develop algorithms for cell inflation in the context of incremental placement. During each iteration of CRISP, we determine areas of congestion and inflate cells in the most congested areas preferentially. Thus cells which are

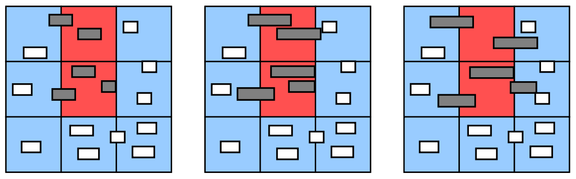

Figure 3: Placement with congested areas (left). CRISP inflates cells in these areas (center), and spreads them (right).

consistently found to be in heavily congested regions grow in size more quickly over time than those in light congestion, which is illustrated in Figure 5 (right).

We inflate cells in proportion to their pin counts in order to reduce pin density in congested regions. Empirically this step improves detailed routability (see Section 4). The width of cell c during iteration i, width(c,i), is

$$\max(\text{width}(c,i-1)+1, \lceil(1+\alpha T)\text{numPins}(c)\rceil)$$

where T is the number of times c has been in a congested region, α is the width increment and width$(c,0)$ is the initial width of c. An example of CRISP inflating cells in shown in Figure 3. We use the same $\alpha = 0.2$ for all of our experiments in Section 4.

3.3 Incremental spreading

The FastPlace iterative local refinement (ILR) framework [13] is insufficient for use in CRISP because any cell can be moved at any iteration. We enhance ILR for use in CRISP and remove this limitation. For each grid tile, we define a target density and a multiplier describing the relative importance of area requirements versus wirelength for the tile. At the beginning of each round, tiles which are above their target density have their multiplier increased so that satisfying their density constraint becomes more important than change in wirelength; tiles which meet their density constraint have their multipliers reduced. During each round, only movable cells contained within tiles that do not meet their density constraint are examined. Additionally, we impose a greedy ordering on cells so that those with better gain in cost function are moved preferentially. When we call this modified ILR during CRISP (Figure 4, line 20), we assign the target density for a grid tile to be the density the tile had at the beginning of the iteration of CRISP before cells were inflated. Overall, these modifications ensure that areas with no congestion, and thus no cell inflation, will remain undisturbed during spreading, making ILR suitable for incremental placement.

Legalization and detailed placement. The fidelity of layout modeling is essential to the success of CRISP. If routing or pin-density hot-spots identified by the router do not correspond to actual areas of routing congestion, CRISP could do harm to the placement rather than help. Since legalization often changes the routabililty of a design, it is particularly import that all placements CRISP evaluates using a router be legal. Thus any solution returned by CRISP will necessarily be legal and any observed improvements in routability will carry over into the final result. Unfortunately, legalization in the presence of many fixed obstacles often significantly perturbs locations, increasing wirelength, so we run detailed placement techniques after legalization to recover wirelength. To save runtime and preserve placement spreading, we limit detailed placement to one round of transforms.

3.4 The CRISP flow

In Figure 2, we outline the flow of CRISP. Pseudocode is also given in Figure 4. An example of CRISP reducing congestion on a highly-congested commercial design is shown in Figure 6.

CRISP: CONGESTION REDUCTION BY ITERATED SPREADING DURING PLACEMENT

```
▷ Input: Placement $P$, Global router $GR$, Cell width increment $\alpha$,
▷ Congestion target $congTarget$, Maximum iterations $maxIter$,
▷ Maximum area increase per iteration $maxAreaIter$
▷ Output: Congestion optimized placement $PBest$
create two arrays, $timesCongested$ and $cellCong$, having size
numMovableCells($P$), entries initialized to 0
if(isPlacementLegal($P$) == FALSE)
then legalizePlacement($P$), doDetailedPlacement($P$)
$congMap$ = callRouter($GR$,$P$), $pinMap$ = buildPinDensityMap($P$)
$PBest = P$, $CongBest$ = getCongMetric($congMap$,$pinMap$)
$currArea$ = getUsedCoreArea(), $totalArea$ = getTotalCoreArea()
$iters = losingStreak = 0$, $originalWidth$ = getCellWidths($P$)
do
    foreach cell $c$
        $cellCong[c]$ = max(lookUpCong(congMap,getLocation($P$,$c$)),
          lookUpCong(pinMap,getLocation($P$,$c$))
    $maxAreaThisIter = \min(0.95 \cdot totalArea,\ currArea + maxAreaIter \cdot totalArea)$
    foreach cell $c$ in order of decreasing congestion
        if($cellCong(c) < congTarget$) then break
        $timesCongested[c]{+}{+}$
        $newWidth$ = max(getWidth($P$,$c$) + 1,
          $\lceil(1 + \alpha \cdot timesCongested[c])\cdot$getNumPins($c$)$\rceil$)
        if($currArea$ + ($newWidth$ - getWidth($P$,$c$))$\cdot$getCoreRowHeight() $>$
          $maxAreaThisIter$) then continue
        $currArea$ += ($newWidth$ - getWidth($P$,$c$))$\cdot$getCoreRowHeight()
        setWidth($P$,$c$,$newWidth$)
    spreadPlacement($P$), legalizePlacement($P$), doDetailedPlacement($P$)
    $congMap$ = callRouter($GR$,$P$), $pinMap$ = buildPinDensityMap($P$)
    $CurrCong$ = getCongMetric($congMap$,$pinMap$)
    if($CurrCong < CongBest$)
    then $CongBest = CurrCong$, $PBest = P$, $losingStreak = 0$
    else $losingStreak{+}{+}$
    $iters{+}{+}$
while($iters < maxIters$ and $losingStreak < 2$)
return $PBest$
```

Figure 4: Determining the cells to inflate per CRISP iteration.

Congestion measurement. CRISP first determines the routing congestion of an initial placement by calling a fast and effective global router. To limit runtime, CRISP restricts the detouring that the router is allowed to perform. In the context of academic routing tools, CRISP limits the number of iterations of rip-up and reroute. When using an industrial router, CRISP can use more accurate constraints. Thus it limits the industrial router to 5% detouring (see Section 4). After the global router produces a solution, CRISP generates a congestion map and a pin-density map for the current placement. Using these maps, CRISP determines portions of the design that are problematic in terms of routing congestion or pin density. For pin density, the threshold for our experiments is chosen as 1 pin per minimum area of a standard cell (one standard row high and one site width wide). For routing congestion, we use different thresholds when using academic and industrial design tools. Since academic routers seek to reduce total routing overflow, we set the threshold to 95% congestion. For commercial designs, we wish to reduce the number of nets which have 90% or more congestion and so set a threshold of 85% congestion.

Cell inflation and spreading. Next CRISP assigns a congestion number to each standard cell based on the routing and pin density in the regions it occupies. CRISP limits the amount of inflation that may happen during a particular iteration by imposing a user defined maximum area increase. Each cell is examined for inflation in order of decreasing congestion until all cells have been examined or the maximum area is reached for that iteration. For all of our experiments, we limit inflation to 1% of the core area per iteration. In addition, we impose a limit of 95% core area usage to ensure that a legal placement is feasible. After as many congested cells as possible are inflated under the area limits, CRISP spreads the newly inflated cells to make the placement more legal. The goal of spreading is to produce a nearly-legal placement by perturbing the original solution as little as possible, without significantly degrading wirelength. CRISP's inflation and spreading steps are illustrated in Figure 3, and described in Sections 3.2 and 3.3. After the solution has been spread, CRISP calls a legalizer followed by detailed placement to recover wirelength. Figure 5 (center) shows a congestion map for an academic design and Figure 5 (right) shows which cells are inflated and to what degree during CRISP.

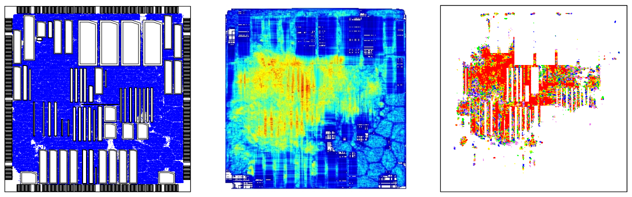

Figure 5: Placement of `adaptec1` with 60% target density (left), global routing congestion map (center), and a map of cells inflated during the first five iterations of CRISP (right). Cell colors at the right correspond to the relative amount of inflation with red cells being the greatest followed by orange, yellow, green, blue and violet.

Iterative congestion reduction. CRISP then calls the global router and compares routability metrics with previous placements. If the new placement improves routability metrics, it is saved. CRISP iterations continue in this way until a stopping criterion is met. Stopping criteria include (i) a maximum number of iterations, (ii) a maximum number of iterations in a row without congestion metric improvement, (iii) area restrictions preventing cell inflation, or (iv) complete elimination of congestion. CRISP returns the best-seen placement in terms of congestion metrics.

4. EXPERIMENTAL RESULTS

We compare CRISP with state-of-the-art congestion reduction techniques on both academic and commercial designs. For academic designs, we choose the ISPD placement and routing contest benchmarks. We place and route these designs with academic tools and compare CRISP with academic incremental congestion reduction techniques. On commercial designs, we demonstrate how CRISP reduces global congestion and detouring, improves detail routability, and is used to shrink die size.

4.1 ISPD contest benchmarks

Benchmark setup. To test the effectiveness of CRISP, we placed the ISPD placement and routing contest benchmarks [9] with the academic placer mPL6 [3]. mPL6 is an analytical placer which finished 2nd place overall at the ISPD 2006 placement contest. The 2006 placement contest featured density constraints for all benchmarks and mPL6 achieved the best total wirelength, while largely observing density constraints, but lost to the contest winner by runtime. For each benchmark, we produce two placements, one routable and one unroutable, by varying the density constraint passed to mPL6. The ISPD contest benchmarks range in size from 211,447 objects (543 fixed) for `adaptec1` to 2,507,953 objects (26,582 fixed) for `newblue7`. We exclude `newblue3` from our experiments because we found that it is trivially unroutable: it contains a standard cell (`o389042`) which connects to over 2200 nets using the same pin. No GCell in `newblue3` has capacity for so many nets, making global routing without overflow impossible.

Routability evaluation. To determine routability and guide CRISP, we use the winning router of the ISPD 2008 routing contest, NTHU-Route 2.0 [4]. CRISP limits NTHU-Route 2.0 to a single round of rip-up and reroute. This limits the runtime of the router

Benchmark & Target density	Placement			Global routing (NTHU-Route 2.0 [4])						Reduction of		
	Flow	Runtime (min)	HPWL (e6)	Estimated overflow	Runtime (min)	Final overflow	Vias (e6)	RWL (e6)	Detour ratio	Vias	RWL	Detours
ad1, 80%	mPL6	61	81.7	174626	1030	560	1.98	5.12	1.220	—	—	—
	mPL6 + Bonn	126	83.0	152802	52	16	1.88	4.77	1.103	5.1%	7.0%	11.7%
	mPL6 + CRISP	88	83.1	148486	65	36	1.83	4.72	1.100	7.4%	7.9%	12.0%
ad1, 70%	mPL6	68	84.7	124186	12	0	1.79	4.62	1.061	—	—	—
	mPL6 + Bonn	134	86.0	105482	5	0	1.78	4.62	1.045	0.2%	0.1%	1.6%
	mPL6 + CRISP	97	85.3	100784	5	0	1.72	4.54	1.045	1.8%	3.6%	1.6%
ad2, 70%	mPL6	80	97.3	97224	>1440	8086	2.00	5.42	1.104	—	—	—
	mPL6 + Bonn	159	100.0	70182	5	6	1.96	5.36	1.037	2.2%	3.0%	6.7%
	mPL6 + CRISP	108	96.1	66166	62	92	1.84	5.05	1.049	7.7%	6.7%	5.5%
ad2, 60%	mPL6	89	103.2	41322	1	0	1.92	5.29	1.025	—	—	—
	mPL6 + Bonn	179	107.7	28734	1	0	1.92	5.43	1.020	0.1%	-2.5%	0.5%
	mPL6 + CRISP	113	100.7	27174	1	0	1.80	5.08	1.021	6.3%	4.0%	0.4%
ad3, 80%	mPL6	379	216.9	269594	44	38	3.90	12.16	1.049	—	—	—
	mPL6 + Bonn	695	219.7	227630	11	0	3.88	12.20	1.042	0.5%	-0.3%	0.7%
	mPL6 + CRISP	442	218.7	182836	9	0	3.70	11.97	1.038	7.5%	3.6%	1.1%
ad3, 70%	mPL6	305	226.9	196600	13	0	3.82	12.37	1.037	—	—	—
	mPL6 + Bonn	614	228.7	167068	6	0	3.82	12.41	1.033	0.1%	-0.3%	0.4%
	mPL6 + CRISP	391	224.4	107830	4	0	3.54	11.92	1.027	8.8%	5.1%	1.0%
ad4, 90%	mPL6	221	191.2	97972	>1440	1266	3.62	10.98	1.078	—	—	—
	mPL6 + Bonn	460	192.1	84052	17	68	3.53	10.52	1.019	2.6%	4.2%	5.9%
	mPL6 + CRISP	267	192.3	49334	3	0	3.31	10.33	1.014	8.6%	5.9%	6.4%
ad4, 80%	mPL6	252	194.8	57474	4	0	3.44	10.50	1.013	—	—	—
	mPL6 + Bonn	533	195.3	47510	4	6	3.43	10.50	1.012	0.2%	0.0%	0.1%
	mPL6 + CRISP	359	195.9	9804	2	0	3.14	10.21	1.008	8.7%	2.8%	0.5%
ad5, 70%	mPL6	408	365.2	190026	56	4	5.67	13.91	1.036	—	—	—
	mPL6 + Bonn	785	372.0	150008	13	0	5.65	13.94	1.024	0.4%	-0.2%	1.2%
	mPL6 + CRISP	522	359.1	108950	6	0	5.19	13.17	1.020	8.5%	5.3%	1.6%
ad5, 60%	mPL6	414	391.2	85008	5	0	5.55	14.21	1.017	—	—	—
	mPL6 + Bonn	777	396.7	75314	3	0	5.56	14.34	1.016	-0.2%	-0.9%	0.1%
	mPL6 + CRISP	584	385.0	28892	2	0	5.08	13.57	1.013	8.5%	4.5%	0.4%

Table 1: Using the Bonn flow [2] and CRISP to improve the routability of unroutable and routable mPL6 [3] placements of ISPD contest benchmarks [9]. Detouring is measured as the ratio of global routing segments to FLUTE [6] Steiner wirelength. Due to space limitations, we only report results for the `adaptec` benchmarks.

as well as the detouring in the routed solution.[1] Final routability of each placement is determined by routing the nets using NTHU-Route 2.0 with default parameters.

Comparing routability improvement techniques. We compare CRISP against a state-of-the-art congestion reduction technique based on BonnPlace [2] that we refer to as the “Bonn flow.” The Bonn flow temporarily inflates standard cells in congested regions, but differs from CRISP in that it inflates all cells where routing resources are more than 100% utilized. The techniques employed by BonnPlace are not incremental, so we make the Bonn flow incremental by first estimating congestion with NTHU-Route 2.0, inflating all cells in congested regions, and lastly replacing the inflated design with mPL6 using the initial target density.

Table 1 compares CRISP to the Bonn flow on unroutable and routable placements of the ISPD benchmarks. Due to space limits, we only reproduce detailed results for the `adaptec` benchmarks. For unroutable benchmarks, the Bonn flow improves via counts by 1.4%, routed wirelength by 1.6% and detouring by 3.3% on average. On the same designs, CRISP improves via counts by 8.7%, routed wirelength by 6.5% and detouring by 5.3% on average. CRISP improves the routability of each of the unroutable designs whereas the Bonn flow degrades routability on `newblue4`. Of the 13 unroutable benchmarks, CRISP produces the best solutions on 11 of the 13, with the Bonn flow producing less routing overflow on `adaptec1` and `adaptec2`. For the routable benchmarks, the Bonn flow maintains routability on all but `adaptec4`, reduces via counts and detouring by 0.1% and 0.5%, respectively, but *increases* routed wirelength 0.7%. CRISP produces the best results on all routable test-cases, decreasing via counts by 6.8%, routed wirelength by 4.0% and detouring by 0.8%. Note that CRISP is faster than the Bonn flow in all cases; CRISP takes 38% of the runtime of placing the design from scratch.

[1]The initial routing produced by NTHU-Route 2.0 uses Steiner trees, so at least one round of rip-up and reroute must be performed for detouring to occur.

Design	CRISP used?	Routing congestion		Timing slack (ns)	
		100% nets	90% nets	Worst	Total negative
design 1	No	18.1%	19.8%	-4.861	-177020
	Yes	**0.4%**	**6.8%**	**-3.292**	**-175931**
design 2	No	0.7%	4.7%	-0.961	-1249
	Yes	**0.2%**	**2.3%**	**-0.904**	**-1187**
design 3	No	4.3%	9.6%	**0.096**	0
	Yes	**0.1%**	**1.5%**	0.072	0
design 4	No	5.7%	11.7%	-0.097	**-17.8**
	Yes	**0.1%**	**3.2%**	**-0.062**	-18.6

Table 2: CRISP’s global routing and timing impact on commercial designs. For congestion we report the percentage of nets at least 90% and 100% congested (smaller is better).

4.2 Commercial designs

Timing impact of CRISP. To judge how effective CRISP is at preserving the timing characteristics of commercial designs, we added CRISP to an industrial physical-synthesis flow. We applied CRISP to four designs which have high congestion after the initial placement stage of the flow. After CRISP, we applied medium effort timing transformations to optimize critical paths as well as the timing histogram. These transformations mainly consist of buffering and resizing techniques. After these timing optimizations, we measured timing with an industry timer and report the results in Table 2. Designs 1 and 2 in Table 2 are large commercial designs with approximately 1,000,000 objects (20,000 fixed blockages) and 700,000 objects (90,000 fixed blockages), respectively. Designs 3 and 4 are smaller designs both with approximately 100,000 objects (1,000

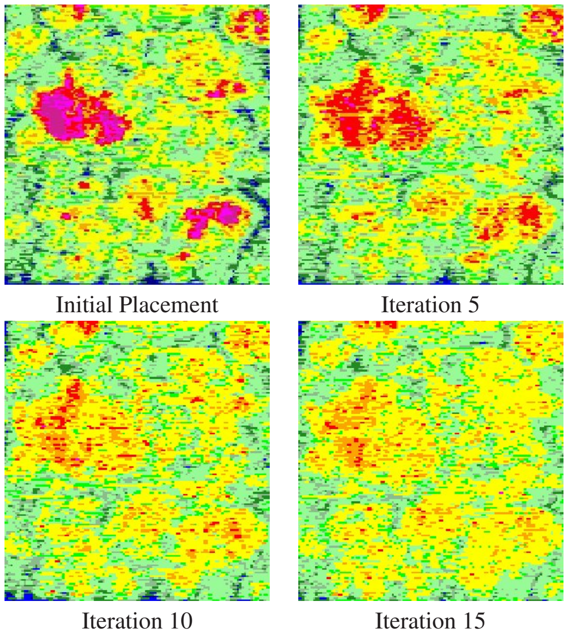

Figure 6: Incrementally relieving congestion problems on a heavily congested industrial design with low whitespace. Areas colored pink and purple have global routing resource usage over 100%. These areas are targeted by CRISP and eliminated.

fixed blockages). For each design, we report congestion and timing statistics after timing optimization. For all four designs, CRISP is effective in reducing the number of nets which pass through areas which are at least 100% or 90% congested. In terms of timing, CRISP has better worst slack in three of the four layouts and better total negative slack in the two larger designs, with only minor degradations for the smaller designs. Since CRISP does not consider timing in its flow, we attribute the gains in timing to the fact that the placements are more spread and thus to effectively apply cell resizing and buffering.

Detailed routing improvement. To judge the effectiveness of pin-density congestion removal by CRISP on detailed routing, we chose 40 high-performance designs and ran them through an industrial physical-synthesis flow. We added CRISP to the flow after clocks were inserted into the design such that CRISP targeted only pin density. On average, CRISP was able to reduce detailed routing runtime by 10.2%, detoured nets by 4.5%, DRC violations by 79.0% and shorts & opens by 62.5%. CRISP increased DRC violations, shorts or open count in five of the 40 designs, but by only one violation, short or open in these designs.

Core area reduction. Previous work has optimized routability of designs in order to reduce routing violations, routed wirelength and turn-around-time. While these are important metrics which we evaluate in our experiments, they do not necessarily communicate all the benefits that strong place-and-route tools can provide such as the ability to reduce manufacturing cost. To this end, we worked with expert designers to re-floorplan design 3 from Table 2 to use less die area and fewer routing resources. The result is design 4 (also shown in Table 2), which uses 5% less area than design 3. This increased the design utilization from 73% to 79%, which is high for a modern design. This also provides less area with which to perform spreading during CRISP. As Table 2 shows, without CRISP design 4 would have been extremely difficult to detail route since 5.7% of its nets were 100% or more congested after timing optimization. The congestion of design 4 during CRISP is shown in Figure 6. After applying CRISP to make design 4 routable, we used industry timing optimizations to close on timing and inserted clocks. After clock insertion, we used CRISP again to eliminate areas of high pin density which reduced shorts and opens from 370 to 41 and detailed routing runtime from 10.9 hours to 7.4 hours. The designers were able to fix all shorts and opens with minor alterations after CRISP, making design 4 routable without violation.

5. CONCLUSIONS

In this paper we have presented CRISP, an incremental technique for Congestion Reduction by Iterated Spreading during Placement. CRISP combines highly accurate congestion modeling with carefully chosen incremental placement transformations. CRISP leverages recent advances in global routing algorithms to model congestion and enhancing previous congestion-driven placement techniques to make them incremental. We have empirically validated CRISP on a number of modern placement instances using (i) academic tools and (ii) integrated industrial design-tool flows. CRISP consistently improves routability on common benchmarks, reducing via counts by 8.7%, global routed wirelength by 6.5% and detouring by 5.3%. We have also verified CRISP's effectiveness on industrial designs and demonstrated CRISP's ability to preserve timing and improve detailed routability by eliminating pin-density hot-spots. Finally, we have shown that with the aid of strong place-and-route tools, designers can shrink die sizes, which leads to savings in manufacturing cost; we believe that our work is the first to demonstrate this link.

6. REFERENCES

[1] C. J. Alpert, D. P. Mehta and S. S. Sapatnekar, eds., *Handbook of Algorithms for VLSI Physical Design Automation*, CRC Press, 2008.

[2] U. Brenner and A. Rohe, "An Effective Congestion Driven Placement Framework," *ISPD* 2002, pp. 6-11.

[3] T. F. Chan, J. Cong, J. Shinnerl, K. Sze and M. Xie, "mPL6: Enhanced Multilevel Mixed-size Placement with Congestion Control," *Modern Circuit Placement*, eds. G.-J. Nam and J. Cong, Springer, pp. 247-288, 2007.

[4] Y.-J. Chang, Y.-T. Lee and T.-C. Wang, "NTHU-Route 2.0: A fast and stable global router," *ICCAD* 2008, pp. 338-343.

[5] C. C. N. Chu and M. Pan, "IPR: An Integrated Placement and Routing Algorithm," *DAC* 2007, pp. 59-62.

[6] C. C. N. Chu and Y.-C. Wong, "FLUTE: Fast Lookup Table Based Rectilinear Steiner Minimal Tree Algorithm for VLSI Design," *IEEE TCAD* 27(1), pp. 70-83, 2008.

[7] Z.-W. Jiang, B.-Y. Su and Y.-W. Chang, "Routability-driven analytical placement by net overlapping removal for large-scale mixed-size designs," *DAC* 2008, pp. 167-172.

[8] C. Li, M. Xie, C.-K. Koh, J. Cong and P. H. Madden, "Routability-driven Placement and White Space Allocation," *IEEE TCAD* 26(5), pp. 858-871, 2007.

[9] G.-J. Nam, C. C. N. Sze and M. Can Yildiz, "The ISPD global routing benchmark suite," *ISPD* 2008, pp. 156-159.

[10] N. Selvakkumaran, P. N. Parakh and G. Karypis, "Perimeter-degree: A Priori Metric for Directly Measuring and Homogenizing Interconnection Complexity in Multilevel Placement," *SLIP* 2003, pp. 53-59.

[11] P. Spindler and F. M. Johannes, "Fast and accurate routing demand estimation for efficient routability-driven placement," *DATE* 2007, pp. 1226-1231

[12] K. Tsota, C.-K. Koh and V. Balakrishnan, "Guiding Global Placement with Wire Density," *ICCAD* 2008, pp. 212-217.

[13] N. Viswanathan, M. Pan and C. C. N. Chu "FastPlace 3.0: A Fast Multilevel Quadratic Placement Algorithm with Placement Congestion Control," *ASP-DAC* 2007, pp. 135-140.

[14] X. Yang, R. Kastner and M. Sarrafzadeh, "Congestion Estimation During Top-down Placement," *IEEE TCAD* 21(1), pp. 72-80, 2002.