Limiting Interconnect Heating in Power-Driven Physical Synthesis

Xiuyan Zhang
Department of ECE
University of Illinois at Chicago
xzhang87@uic.edu

Shantanu Dutt
Department of ECE
University of Illinois at Chicago
dutt@uic.edu

Abstract

Current technology trend of VLSI chips includes sub-10 nm nodes and 3D ICs. Unfortunately, due to significantly increased Joule heating in these technologies, interconnect reliability has become a significant casualty. In this paper, we explore how interconnect power dissipation (of $CV^2/2$ per logic transition) and thus heating can be effectively constrained during a power-optimizing physical synthesis (PS) flow that applies three different PS transformations: cell sizing, Vth assignment and cell replication; the latter is particularly useful for limiting interconnect heating. Other constraints considered are timing, slew and cell fanout load. To address this multi-constraint power-optimization problem effectively, we consider the application of the aforementioned three transforms simultaneously (as opposed to sequentially in some order) as well as simultaneously across all cells of the circuit using a novel discrete optimization technique called discretized network flow (DNF). We applied our algorithm to ISPD-13 benchmark circuits: the ISPD-13 competition was for power optimization for cell-sizing and Vth assignment transforms under timing, slew and cell fanout load constraints; to these we added the interconnect heating constraint and the cell replication transform—a much harder transform to engineer in a simultaneous-consideration framework than the other two. Results show the significant efficacy of our techniques.

1. Introduction

Interconnect optimization is of make-or-break significance in VLSI designs in the sub-10nm regime. Since the density of transistors per unit area is increased dramatically, physical interconnects density is significantly high. Furthermore, interconnects are subjected to increased current density due to the thinning of interconnects, as well as increased power dissipation and thus heating due to high clock frequencies of current chips. All these factors contribute to high potential for electromigration in interconnects that can either cause breaks in them or cause shorts with adjacent interconnects that are densely packed in current technologies. Furthermore, With the slow-down of Moore's law of scaling transistors, the industry is adopting 3D IC technology to extend Moore's law by stacking chips vertically, and in this technology, Joule heating is the most serious reliability concern for interconnects [1].

The goal of this paper is to perform leakage-power-minimizing physical synthesis (PS) of a given circuit with the following considerations. (1) Explicitly constrain interconnect heating to a given upper bound. Fig. 1 illustrates the transformation of the heating constraint into a cell load capacitance constraint (this is also discussed in Sec. 2-B). (2) Consider three physical synthesis transforms, gate sizing, Vth assignment and cell replication to achieve the above. (3) For better optimization, consider all the three transforms *simultaneously* as well as apply them across all circuit components *simultaneously*—it is important to note that current state-of-the-art work such as [2-7] targeting the new ISPD-12/13 sizing problems, like earlier works, have applied PS transforms sequentially across the circuit in either a greedy or topologically-ordered approach. Simultaneous consideration of multiple transforms (or design points/variables) and multiple constraints simultaneously across the entire circuit can yield much better solutions than considering these design points sequentially and applying them sequentially across the circuit that most state-of-the-art techniques and commercial tools do. This has been clearly demonstrated in [10].

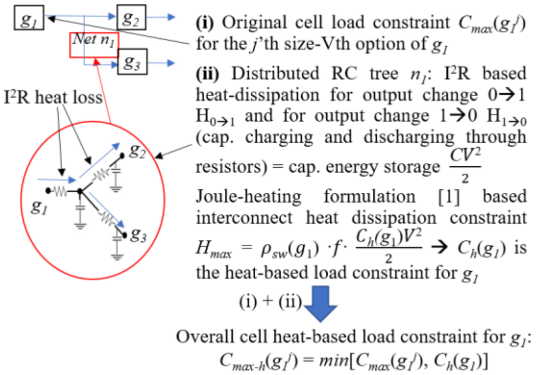

g1
g2
Net n1
g3
I2R heat loss
g2
g1
g3
(i) Original cell load constraint $C_{max}(g_1^j)$ for the j'th size-Vth option of g_1
(ii) Distributed RC tree n_1: I²R based heat-dissipation for output change 0→1 $H_{0\rightarrow1}$ and for output change 1→0 $H_{1\rightarrow0}$ (cap. charging and discharging through resistors) = cap. energy storage $\frac{CV^2}{2}$
Joule-heating formulation [1] based interconnect heat dissipation constraint $H_{max} = \rho_{sw}(g_1) \cdot f \cdot \frac{C_h(g_1)V^2}{2} \rightarrow C_h(g_1)$ is the heat-based load constraint for g_1
(i) + (ii)
Overall cell heat-based load constraint for g_1:
$C_{max\text{-}h}(g_1^j) = min[C_{max}(g_1^j), C_h(g_1)]$

Figure 1: I^2R-based cell heat-based load constraint $C_{max-h}(g_1^j)$ for cell g_1's *j*'th size-Vth option g_1^j.

Leakage power optimization for the either cell sizing or cell sizing combined with Vth assignment (subsequently called the *cell selection problem*), which is a subset of the more general PS problem, has been well studied over the years and various techniques have been developed such as linear programming (LP) [11, 12], convex programming [13, 14], Lagrangian relaxation (LR) [3-7, 15-17], dynamic programming (DP) [18-21], and network flow [3, 22]. Some of these techniques solve the cell selection problem in continuous domain which is runtime-efficient and performance-effective if a continuous range of cell sizes and Vth's are available. However, most silicon-verified cell libraries are by necessity discrete. Some techniques also use simpler delay formulations that are not accurate in current technology.

The ISPD 2012 problem [23] has addressed both issues above by providing a cell library with discrete sizing options and an LUT-based cell delay model. Besides the timing constraint, this contest problem also considers the maximum load capacitance and maximum slew constraints for each cell input which makes the sizing-Vth problem more complex than past techniques. Furthermore, the ISPD 2013 [24] contest problem introduced interconnect RC-tree structures with a more realistic delay model than Elmore.

There have been quite a few recent works that address the ISPD 2012 [23] and 2013 [24] sizing-Vth problems, that have each advanced the state-of-the-art considerably [2-7]. The work in [21] uses the go-with-the-winner metaheuristic to select non-dominated configurations from the space of sensitivity-based functions for each cell. In [3], initially a LR-based approach combined with the

SLIP '22, November 3, 2022, San Diego, CA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9536-6/22/11…$15.00
https://doi.org/10.1145/3557988.3569712

Lagrangian Dual Problem (LDP) is used to obtain the best performing design that explores the space of all possible cell types using parallelism. Subsequently, the power of a timing valid design is reduced using a min-cost flow technique, followed by a residual-slack utilizing method to further reduce power. In [5,6] also the authors also use LR-LDP using parallelism and a novel Lagrangian multiplier λ update policy to obtain fast solutions. The work in [7] represents the state-of-the-art for both ISPD 2012 and ISPD 2013 problems. Its main stage also uses an LR-LDP approach that operates at the local level of each cell by minimizing its local power and the lambda-delay function such that some local slack satisfaction is achieved. In addition to basic delay cost of a cell, unlike other work, it also considers the incremental delay change of alternative options from the current design point of each cell due to input slew change and its effect on cell delays in its fanout cone.

In this paper, we solve a superset of the ISPD-13 problem in two ways: **(1)** as mentioned earlier, in addition to the gate max-load, gate input slew, and delay constraints, we address the important interconnect heating constraint—this constraint is converted to output load constraint for each interconnect, and for each cell size-Vth option, we take the minimum (more strict) of the ISPD-13 max-load constraint and heating based load constraints as the new max-load constraint; see Fig. 1. We call this new problem ISPD13+heat (ISPD13-H) problem. **(2)** Besides the cell sizing-Vth combined transform of the ISPD-13 problem, we also consider the much more difficult transform of cell replication.

In solving the ISPD13-H problem, we observed that some circuits needed a significant amount of extra power compared to the ISPD13 solution to meet the heating based max-load constraint. This is necessitated by having to significantly upsize several driver cells to compensate for the delay increase caused frequently by downsizing their sinks to meet the heating-aware max-load constraint. In other situations, there was also a delay violation as depicted in Fig. 2(a). In this scenario, the delay increase due to sinks of gate g_1 needing to be downsized to meet its heat-based load constraint $C_h(g_1)$ (see Sec. 2-B) is compensated (to meet corresponding path delay constraints trough the sinks) by their own sinks needing to be upsized significantly so that their output delays reduce. This can result in either: (a) delay-constraint satisfaction for the concerned paths with significant power increase; or (b) delay violation(s) in these paths when the aforementioned delay reductions at the sink-of-sink outputs are not enough.

The above problem(s) can be alleviated by replication of cell $\boldsymbol{g_1}$, and the partitioning of its sinks across the 2 replicated cells $\boldsymbol{g_{1,1}}$ and $\boldsymbol{g_{1,2}}$, so that their nominal load intrinsically reduces. This allows easier satisfaction of the heating-based load constraint for the replicated cells without significant sink downsizing and corresponding upsizings of their fanins and/or fanouts (of course, the counter-balancing issue of the increased load on the drivers of the replicated cells will also need to be accounted for). This is illustrated in Fig. 2(b). We call this new problem ISPD13+heat+replicate (ISPD13-HR) problem. However, while the ISPD13 and ISPD13-H problems are parameterized selection problems of a given topologically static circuit design, with the parameters or variables being the cells' sizing-Vth options, the ISPD13-HR problem is fundamentally different in that it is a combination of determining the non-parameterized circuit design/topology and the sizing-Vth parameterized problem of the resulting cells.

Besides (1) and (2) above, the further contributions of this paper are: **(3)** Devising a discretized network flow (DNF) [10] based model and algorithm for solving the ISPD13 and ISPD13-H problems in which the sizing-Vth determination of all cells (cell selection) are *simultaneously* explored unlike in prior methods. **(4)** Devising a DNF-based model for solving the ISPD13-HR problem in which the possibility of cell replication for targeted cells, and the sizing-Vth determination of all cells are also *simultaneously* explored.

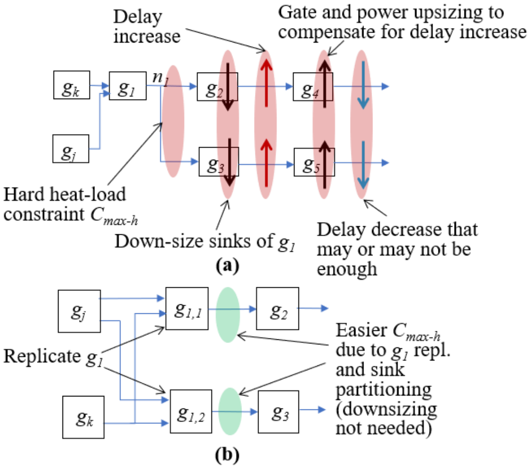

Delay increase
Gate and power upsizing to compensate for delay increase
g_k
g_1
n_1
g_2
g_4
g_j
g_3
g_5
Hard heat-load constraint C_{max-h}
Down-size sinks of g_1
Delay decrease that may or may not be enough
(a)
g_j
$g_{1,1}$
g_2
Replicate g_1
Easier C_{max-h} due to g_1 repl. and sink partitioning (downsizing not needed)
g_k
$g_{1,2}$
g_3
(b)

Figure 2: Two ways to satisfy the violated $\boldsymbol{C_{max-h}(g_1^j)}$, when $\boldsymbol{load_{op}(g_1)} > \boldsymbol{C_{max-h}(g_1^j) = C_h(g_1)}$: (a) Down-size sinks of $\boldsymbol{g_1}$, which causes negative power/delay effects. (b) Replicate $\boldsymbol{g_1}$ with no or little delay/power increase in the fanout cones of $\boldsymbol{g_{1,1}}$ and $\boldsymbol{g_{1,2}}$.

We also note that buffer insertion is an alternative to cell replication to achieve the aforementioned purpose. However, due to a cleaner sink cell bi-partitioning that is possible for cell replication, we pursue this approach in this paper. For example, if gate g_1 has 4 sinks, then its 2 replicated cells will have 2 sinks each. In buffer insertion to tackle a hard load constraint, if g_1 drives 2 sinks and the buffer (which will drive the other 2 sinks), its load may still be too high. Alternatively, if g_1 drives 1 sink and the buffer, the latter will need to drive 3 sinks, which may be too high a load for it. However, both approaches have their pros and cons, and in future work we will consider both transforms simultaneously using DNF so that a high QoS is obtained that applies either transform or none to each cell.

The rest of the paper is organized as follows. Section 2 gives problem definitions and general LR formulations. Section 3 and 4 provides the main ideas and concepts of the basic DNF method as applied to the ISPD13/ISPD13-H problems. Next, Sec. 5 discusses our modified DNF modeling for the ISPD13-HR problem. Finally, the experimental results and conclusions are given in Secs. 6 and 7, respectively.

2. Problem Definitions

The problems described above can be modeled as *discrete option selection problems* (OSPs), a wide subclass of DOPs, that have linear or non-linear optimization and constraint functions. The discretized network flow (DNF) algorithm we use is particularly suited to solving OSP problems. An OSP problem is defined as a collection of discrete *option sets* (*OSs*) $\{OS(x_i)\}$ for each *design/option* variable x_i of the problem. Also, given an optimization objective function $F(X)$ that is a function of the set X of design/option variables x_i corresponding each option set $OS(x_i)$, and multiple constraints $H_j(X) \leq b_j$ (we assume ≤ constraints here without loss of generality, but the DNF method can be extended to ≥ constraints), the OSP problem is to choose *exactly one* option from each $\{OS(x_i)\}$ (i.e., each x_i has a discrete value) in order to optimize (minimize or maximize) $F(X)$ while satisfying each constraint.

A) The ISPD13 Problem:

(1) Given a gate-level netlist/circuit C and a standard cell library L, each cell g_i is represented by an *option-choice* variable x_i over the domain $OS(g_i) = \{$size-Vth options/values for gate-type(g_i) in $L\}$.

(2) Optimization problem:

$$\textbf{Minimize } F(x_i) = \sum_{\forall x_i} lp(x_i) \qquad (1)$$

where $lp(x_i)$ is the leakage power corresponding to the option/value chosen for gate sizing-Vth variable x_i,

subject to: upper bound constraints on i) path timing D_c, ii) maximum load capacitance at each cell g_i's output for its j'th size-Vth option g_i^j (we will interchangeably use notations g_i^j and $o_{i,j}$ to mean the same thing, the latter being mainly used in DNF graph formulations, where they are more useful for a general description of DNF), and iii) maximum slew S_c for each cell input.

B) The ISPD13-H Problem:

The I^2R based heat dissipation per second H of an interconnect at cell g_i's output switching from 0 to 1 or 1 to 0 with a load capacitance of C, for a supply voltage of V, a clock frequency $f = \frac{1}{D_c}$, and a switching probability of $\rho_{sw}(g_i)$, is $H = \rho_{sw}(g_i) \cdot f \cdot \frac{CV^2}{2}$. Let H_{max} be the given heat dissipation upper-bound using Joule-heating formulations as in [1]. Since ρ_{sw} and V for a cell in a given circuit is a constant, the heating constraint: $H \leq H_{max}$ is easily converted to a heating-based max-load constraint (as also depicted in Fig. 1):

$$C_h(g_i) \leq \frac{2H_{max}}{\rho_{sw}(g_i) \cdot f \cdot CV^2} \qquad (2)$$

If the ISPD13 max-load constraint for option j of g_i is $C_{max}(g_i^j)$, then the new heating aware max-load constraint $C_{max-h}(g_i^j)$ is:

$$C_{max-h}(g_i^j) = \min[C_{max}(g_i^j), C_h(g_i)] \qquad (3)$$

The ISPD13-H problem is then the same as the ISPD13 one with $C_{max-h}(g_i^j)$ replacing $C_{max}(g_i^j)$ as the maximum load capacitance at cell g_i's output for its j'th size-Vth option g_i^j.

C) The ISPD13-HR Problem:

(1) Given the same inputs as the ISPD13 problem plus two additional option-choice variables $x_{i,1}$ and $x_{i,2}$ for the two possible replicated versions of gate g_i.

(2) Optimization problem:

$$\textbf{Minimize } F(x_i) = \sum_{\forall x_i} (1 - r_i) lp(x_i) + r_i \left(lp(x_{i,1}) + lp(x_{i,2}) \right) \quad (4)$$

where r_i is a 0/1 variable indicating choice of no-replication/replication, respectively, for gate g_i of the original circuit C, **subject to** the same upper bound constraints of ISPD13-H.

D) Details of the ISPD13-HR Constraint Functions:

The detailed OSP modeling of the ISPD13-HR problem is:

$$\textbf{Minimize } F(X) = \sum_{\forall gates\ g_i} (1 - r_i) lp(x_i) + r_i \left(lp(x_{i,1}) + lp(x_{i,2}) \right) \quad (5)$$

where $X = \cup_{all\ gates\ g_i} \{x_i, x_{i,1}, x_{i,2}, r_i\}$, $x_i, x_{i,1}, x_{i,2}$ belong to $OS(g_i) = \{$size-Vth options/values for gate-type(g_i) in $L\}$, and r_i belongs to $\{0,1\}$ (0/1 indicating choice of no-replication/replication for gate g_i of the original circuit C),

subject to:

(i) Input Slew Constraint: For each gate g_i and each sink g_j of g_i, the input slew at $inp(g_j, g_i)$, the input pin of g_j driven by g_i:

$$slew_{inp}(g_j, g_i) = \max_{drivers\ g_k of g_i} \{ f_{slew}(slew_{inp}(g_i, g_k), x_i,$$
$$load_{op}(g_i)) \} + slew(RC\text{-}tree(g_i, g_j)) \leq slew\ constraint\ S_c \quad (6)$$

where: (a) $f_{slew}(slew_{inp}(g_i, g_k), x_i,\ load_{op}(g_i))$ is an $(m+2)$-variable non-linear function that determines the slew between $inp(g_i, g_k)$ and the output of g_i (the slew of the so-called timing arc arc (g_i, g_k) from $inp(g_i, g_k)$ and the output of g_i); m is the number of sinks of g_i; (b) $slew_{inp}(g_i, g_k)$ is a recursive equation similar to that above for $slew_{inp}(g_j, g_i)$; (c) $load_{op}(g_i)$ $= \sum_{\forall sinks\ g_s\ of\ g_i} load_{inp}(x_s)$; $load_{inp}(x_i)$ is the input capacitance of gate g_i based on the value of its size-Vth variable x_i; and (d) $slew(RC\text{-}tree(g_i, g_j))$ is the additional slew from the output of g_i to the input of g_j along the corresponding RC path in the interconnect RC-tree connecting g_i's output to the input pins of its sinks; the RC-tree is a constant structure, independent of the design variables x_i's, but the loads $load_{inp}(x_s)$'s at its leaves are dependent on the x_s's of the sinks of g_i.

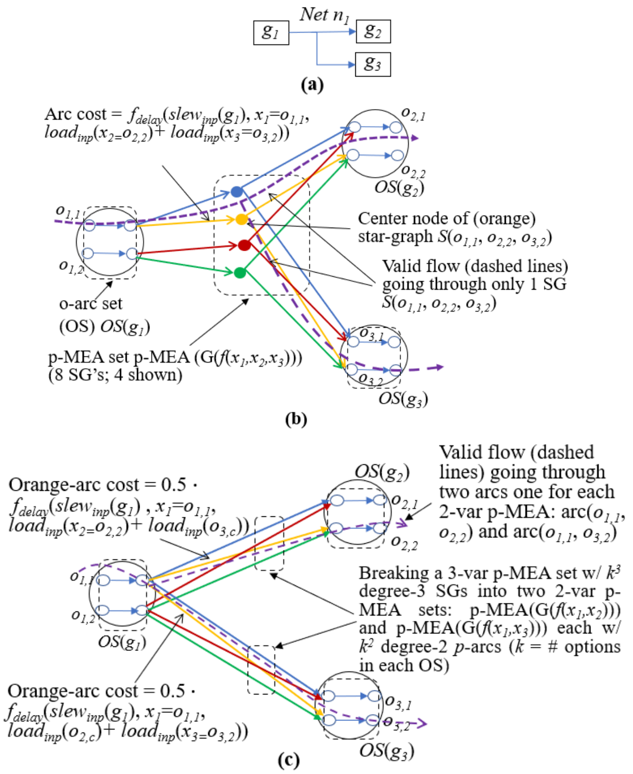

Net n_1
g_1
g_2
g_3
(a)
Arc cost = $f_{delay}(slew_{inp}(g_1), x_1{=}o_{1,1}, load_{inp}(x_{2}{=}o_{2,2}) + load_{inp}(x_3{=}o_{3,2}))$
$o_{2,1}$
$o_{2,2}$
$OS(g_2)$
$o_{1,1}$
$o_{1,2}$
Center node of (orange) star-graph $S(o_{1,1}, o_{2,2}, o_{3,2})$
Valid flow (dashed lines) going through only 1 SG $S(o_{1,1}, o_{2,2}, o_{3,2})$
o-arc set (OS) $OS(g_1)$
p-MEA set p-MEA (G($f(x_1,x_2,x_3)$)) (8 SG's; 4 shown)
$o_{3,1}$
$o_{3,2}$
$OS(g_3)$
(b)
Orange-arc cost = 0.5 · $f_{delay}(slew_{inp}(g_1), x_1{=}o_{1,1}, load_{inp}(x_{2}{=}o_{2,2}) + load_{inp}(o_{3,c}))$
$OS(g_2)$
$o_{2,1}$
$o_{2,2}$
Valid flow (dashed lines) going through two arcs one for each 2-var p-MEA: arc($o_{1,1}$, $o_{2,2}$) and arc($o_{1,1}$, $o_{3,2}$)
$o_{1,1}$
$o_{1,2}$
$OS(g_1)$
Breaking a 3-var p-MEA set w/ k^3 degree-3 SGs into two 2-var p-MEA sets: p-MEA(G($f(x_1,x_2)$)) and p-MEA(G($f(x_1,x_3)$)) each w/ k^2 degree-2 p-arcs (k = # options in each OS)
Orange-arc cost = 0.5 · $f_{delay}(slew_{inp}(g_1), x_1{=}o_{1,1}, load_{inp}(o_{2,c}) + load_{inp}(x_{3}{=}o_{3,2}))$
$o_{3,1}$
$o_{3,2}$
$OS(g_3)$
(c)

Figure 3: (a) A 3-cell subcircuit C'. (b) A directed DNF subgraph representation $G(f_{delay}(\ ))$ of a 3-variable function $f_{delay}(slew_{inp}(x_1), x_1, load_{inp}(x_2) + load_{inp}(x_3))$ as a set p-MEA($G(f_{delay}(x_1, x_2, x_3))$) of 2^3 (generally, k^3, k = # of options in the OS of each gate) star-graphs (SGs), each connecting one 3-variable option combination from the OSs of gates g_1, g_2, g_3. An MEA-valid flow through the p-MEA that goes through exactly 1 SG is shown. (c) Simplification of the 3-OS $G(f_{delay}(\ ))$ by partitioning the 3-OS p-MEA into two 2-OS p-MEAs, each with 2^2 arcs (generally m^2 arcs) representing the correspondingly partitioned functions $0.5 \cdot f_{delay}(slew_{inp}(x_1), x_1, load_{inp}(x_2) + load_{inp}(o_{3,c}))$ and $0.5 \cdot f_{delay}(slew_{inp}(x_1), x_1, load_{inp}(o_{2,c}) + load_{inp}(x_3))$, $o_{i,c}$ is the currently-chosen option for gate g_i, and thus a constant.

(ii) Heating-based Output Load Constraint: For each gate g_i:

$$load_{op}(x_i) \leq \min\ (C_{max}(x_i), C_h(g_i)) \qquad (7)$$

where $C_{max}(x_i)$ is a variable load constraint depending on the size-Vth option x_i, i.e., if $x_i =$ the option g_i^j of g_i, then $C_{max}(x_i) = C_{max-h}(g_i^j)$ (generally larger is the size of g_i, larger or more flexible is its output load constraint), and as explained earlier in Sec 2-B, $C_h(g_i)$ is the heating-constraint derived load constraint on g_i.

(iii) Path Delay Constraint: For each output gate (a gate that drives a primary output (PO) of the circuit or a FF input) g_i driving output pin $op(g_i)$, the formulation and the delay constraint on the worst-case arrival time $t_{arr}(op(g_i), g_i)$ of the signal from g_i at $op(g_i)$ are (for simplicity of exposition here, we assume that each gate drives only one output pin; the formulations are easily extended to driving multiple output pins):

$$\begin{aligned} t_{arr}(op(g_i), g_i) = & \max_{all\ drivers\ g_k of g_i} \{ t_{arr}(g_i, g_k) \\ & + f_{delay}(slew_{inp}(g_i, g_k), x_i, load_{op}(g_i)) \} \\ & + delay(RC - tree(g_i, op(g_i))) \\ & \leq delay\ constraint\ D_c \end{aligned} \qquad (8)$$

where: (a) $f_{delay}(slew_{inp}(g_i, g_k), x_i, load_{op}(g_i))$ is an $(m+2)$-variable non-linear function that provides the delay between $inp(g_i, g_k)$ and the output of g_i (the delay of arc (g_i, g_k)); (b) $t_{arr}(g_i, g_k)$ is the (worst-case) arrival time of the signal at $inp(g_i, g_k)$, and has a similar recursive formulation as given in Eqn.8

for the POs; and (c) $delay(RC-tree(g_i, op(g_i)))$ is the additional delay from the output of g_i to $op(g_i)$, and is an Elmore-style delay model using an effective downstream capacitance model [7, 24].

E) Lagrangian Relaxation Based Modeling:

Lagrangian relaxation (LR) is a well-known approach to solving a constrained optimization problem (such as ISPD13-HR) in which the new objective function includes both the optimization and constraint functions as follows (for simplicity, we assume here that all constraints are of the ≤ or upper-bound type, as is the case for our problems):

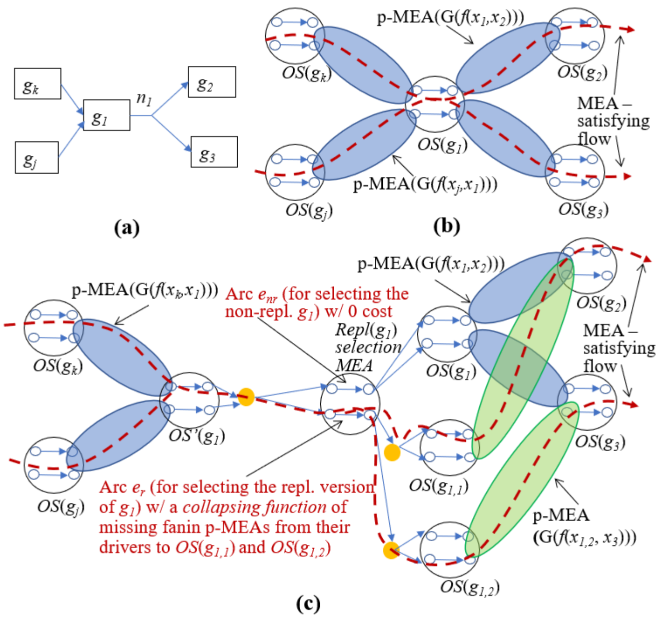
$$\boldsymbol{Minimize}\ F'(X) = F(X) + \sum_j w_j \times (H_j(X) - b_j) \quad (9)$$

where $F(X)$ is the original objective function, the $H_j(X)$'s are the constraint functions of the original problem of the type $H_j(X) \leq b_j$, and the w_j's are Lagrangian multipliers (LM's). As can be seen, in the new objective function $F'(X)$, violations of constraints are penalized by higher function cost. The w_j's can be determined adaptively based on how "critical" (or difficult to satisfy) the H_j's based on the *constraint slack* $b_j - H_j(X)$ after each iteration of the Lagrangian solver (in our case, the DNF algorithm).

3. Representing a Function by a DNF Subgraph

Discretized network flow (DNF) [10] is an effective approach to addressing discrete optimization problems (DOPs) such as OSPs using ideas from min-cost network flow (NF) [25] augmented by some *discretization requirements* that the flow needs to satisfy in order to obtain legal solutions to DOPs. This approach is motivated by the observations: (a) NF, while being a continuous linear optimization technique (it solves a subclass of linear programming problems), has a discrete flavor to it in that the structure on which it operates is a directed graph, and (b) being a continuous solver, it obtains solutions very time efficiently (time complexity is a low-degree pseudo-polynomial). However, clearly, NF cannot produce legal solutions to DOPs, and a set of discretization requirements need to be imposed on NF in order to solve a large class of DOPs (linear or non-linear, and convex or non-convex). We have identified two main discretization requirements which need to be satisfied in NF in order to solve DOPs: 1) a *fixed-charge* or *step function* cost $c(e)$ on some arcs e, which means that if there is any non-zero flow f through e, then a constant cost of $c(e)$ is incurred (similar to the fixed-charge network flow or FCNF problem [26]), and 2) a *mutually exclusive arc* (*MEA*) requirement on various disjoint arc sets S, which is that a non-zero flow only pass through exactly one arc in each such S. For solving an OSP problem, the MEA constraint is used, for example, to ensure that only one option is chosen from each option set of the problem. The resulting NF computation with these discretizations is called *discretized network flow* (*DNF*) [10].

We demonstrate the use of DNF for OSPs (and particularly ISPD13-HR), along with its iterative framework, with an illustration for the $(m+2)$-variable $f_{delay}(slew_{inp}(g_i, g_k), x_i, load_{op}(g_i))$ in Fig. 3. This, like all other functions in the ISPD13 problem, is library based, where the values for some discrete combinations of the input values are provided in a table, and for intermediate input values (not represented in the table), the outputs are obtained via interpolation. Fig. 3(b) gives a directed graph representation $G(f_{delay}(\,))$ of this function for $m = 2$ for the sub-circuit of Fig. 3(a). For simplicity of exposition, we assume here that there are two discrete values (size-Vth options) for the 3 gates shown. In this representation, the graph (and particularly, its arc costs) is instantiated for the value of the input slew (determined via STA) for the current iteration of the DNF solver; hence $f_{delay}(\,)$ becomes in general an $(m+1)$-variable function dependent only on the x_i variables of the driver and sink gates. For each combination q of the size-Vth option of each gate (e.g., $(o_{1,1}, o_{2,1}, o_{3,1})$ in Fig. 3(b)), there is a star-graph (SG), denoted by $S(q)$, representing it: the SG has a center node which has arcs from/to an option for each driver/sink; the cost of the combination (value of $f_{delay}(\,)$) is represented as the cost of the arc from the driver option to the center node (the actual value is not exactly this but based on it—it has a normalized version of the raw delay value so that different metric costs can be added in a meaningful way, and additionally has an LM coefficient). The orange SG in the figure represents the above combination.



g_k
g_j
g_1
n_1
g_2
g_3
(a)
p-MEA(G(f(x_1,x_2)))
$OS(g_k)$
$OS(g_2)$
MEA – satisfying flow
$OS(g_1)$
p-MEA(G(f(x_j,x_1)))
$OS(g_j)$
$OS(g_3)$
(b)
p-MEA(G(f(x_k,x_1)))
Arc e_{nr} (for selecting the non-repl. g_1) w/ 0 cost
p-MEA(G(f(x_1,x_2)))
$Repl(g_1)$ selection MEA
$OS(g_k)$
$OS(g_1)$
$OS(g_2)$
MEA – satisfying flow
$OS'(g_1)$
$OS(g_3)$
$OS(g_j)$
Arc e_r (for selecting the repl. version of g_1) w/ a *collapsing function* of missing fanin p-MEAs from their drivers to $OS(g_{1,1})$ and $OS(g_{1,2})$
$OS(g_{1,1})$
p-MEA (G(f($x_{1,2}$, x_3)))
$OS(g_{1,2})$
(c)

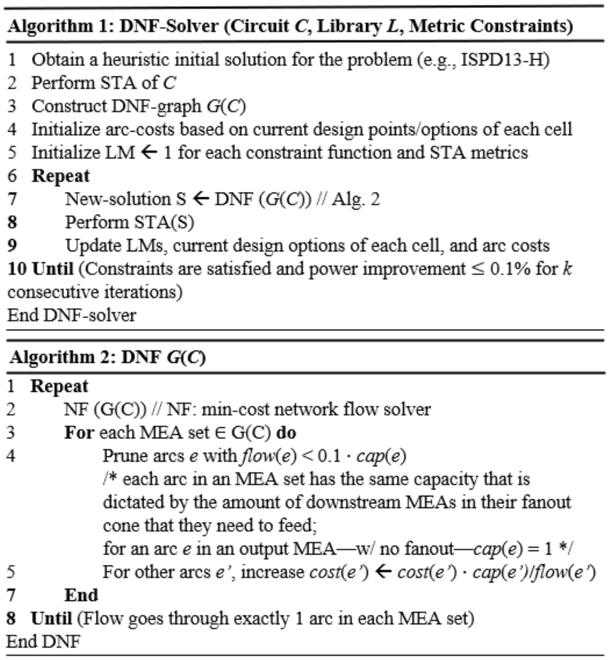
Figure 4: (a) A 5-cell subcircuit. (b) The DNF graph for this subcircuit, and a valid solution flow (red dashed lines) through it. (c) The more complex DNF graph incorporating cell replication as an option for g_1 for the replication-based subcircuit version, shown in Fig. 2(b), of the subcircuit in (a), and a valid solution flow (red dashed lines).

In a min-cost flow based solver, the SG of $G(f(\,))$ for a function f (that is a sub-function in the Lagrangian function $F'(X)$ of Eqn. 9) that the flow goes through, selects the corresponding combination of values for the input variables of $f(\,)$. Clearly for a valid flow-selected solution: (a) The flow should go through exactly 1 SG for each $G(f(\,))$ for a valid selection. (b) Also, since each $OS(g_i)$ will be part of different $G(f(\,))$'s (e.g., for $f_{delay}(\,)$ and $f_{slew}(\,)$ for g_i's own output as well as those of each of its drivers), all valid flows for each such $G(f(\,))$ should also select exactly one option of $OS(g_i)$. As mentioned earlier, these flow requirements are called MEAs for their respective arc sets (e.g., each SG set for each $G(f(\,))$ and the arc set in each $OS(g_i)$; see Fig. 3(b)).

For a function $f(\,)$ with t x_i variables, and assuming a uniform number k of options in each $OS(g_i)$, there are k^t option combinations, and thus k^t SGs in $G(f(\,))$; this set of SGs is denoted by *p-MEA*$(G(f(\,)))$. Since t is at most $(m+1)$, where m is the # of sinks of g_i, the average m is around 3, and the number of pruned/relevant options is about 8, $k^t = 4^8$ is a medium-sized constant that is not computationally intractable.

However, for better runtime efficiency, we partition each $(m+1)$-variable $f(x_i, x_{i1}, \ldots, x_{im})$ (g_i being the driving cell and $g_{i1}, \ldots, g_{im}$, its sinks) into m 2-variable functions $f(x_i, x_{i1}, \{o_{j,c} |\ for\ sink\ g_j\ \neq g_{i1}\}), \ldots, f(x_i, x_{im}, \{o_{j,c} |\ for\ sink\ g_j\ \neq g_{im}\})$, where each $o_{j,c}$ is a constant and is the currently flow-selected option for of g_j (in the previous DNF iteration—in the first iteration, the $o_{j,c}$'s are determined by an initial greedy heuristic). For conciseness, we abbreviate each $f(x_i, x_{ir}, \{o_{j,c} |\ for\ sink\ g_j\ \neq g_{ir}\})$, by $f(x_i, x_{ir})$. Thus we have m $G(f(x_i, x_{ir}))$'s each with a p-MEA set of k^2 simple arcs, as shown in Fig. 3(c); the cost of, say, arc $(o_{i,1}, o_{i1,3})$, is $(1/m) \cdot f(o_{i,1}, o_{i1,3}, \{o_{j,c} |\ j = i2, i3, \ldots\})$. This is a significant decrease from k^{m+1} SGs, each with $(m+1)$ arcs, to only mk^2 arcs. Interestingly, for a convex function, the DNF solver, in multiple iterations, can reach the same (good) solution on the partitioned sub-graphs in $\cup\{G(f(x_i, x_{ir}))\}$ with cost = the minimum cost selection obtained by the solver in the un-partitioned $G(f(x_i, x_{i1}, \dots, x_{im}))$. However, for a non-convex function, this approximation may lead to a local minima being chosen by DNF in $\cup\{G(f(x_i, x_{ir}))\}$ with cost > the minimum cost selection obtained in $G(f(x_i, x_{i1}, \dots, x_{im}))$.

As shown in Fig.3(b), each $OS(g_i)$ is represented in the DNF graph as a set of k arcs, each corresponding to one size-Vth option of g_i. Similar to each p-MEA set, each $OS(g_i)$ also has an MEA requirement, since exactly one option needs to be chosen for each g_i. Further, each arc in $OS(g_i)$ has a normalized leakage power cost corresponding to g_i's option that it represents.



```
Algorithm 1: DNF-Solver (Circuit C, Library L, Metric Constraints)
Obtain a heuristic initial solution for the problem (e.g., ISPD13-H)
Perform STA of C
Construct DNF-graph G(C)
Initialize arc-costs based on current design points/options of each cell
Initialize LM ← 1 for each constraint function and STA metrics
Repeat
    New-solution S ← DNF (G(C)) // Alg. 2
    Perform STA(S)
    Update LMs, current design options of each cell, and arc costs
Until (Constraints are satisfied and power improvement ≤ 0.1% for k
consecutive iterations)
End DNF-solver
```

```
Algorithm 2: DNF G(C)
Repeat
    NF (G(C)) // NF: min-cost network flow solver
    For each MEA set ∈ G(C) do
        Prune arcs e with flow(e) < 0.1 · cap(e)
        /* each arc in an MEA set has the same capacity that is
        dictated by the amount of downstream MEAs in their fanout
        cone that they need to feed;
        for an arc e in an output MEA—w/ no fanout—cap(e) = 1 */
        For other arcs e', increase cost(e') ← cost(e') · cap(e')/flow(e')
    End
Until (Flow goes through exactly 1 arc in each MEA set)
End DNF
```

Figure 5: The DNF Based Solver.

4. The DNF Graph and Solver

The entire DNF graph $\boldsymbol{G(C)}$ for the circuit $\boldsymbol{C}$ is constructed by connecting the various $\boldsymbol{G(f(\,))}$'s according to the structure of the circuit, and merging the common $\boldsymbol{OS(g_i)}$'s of the $\boldsymbol{G(f(\,))}$'s that contain $\boldsymbol{OS(g_i)}$. This is shown in Fig. 4(b) for the circuit in Fig. 4(a).

The DNF solver works over multiple iterations to get better solutions in terms of constraint-satisfaction and power optimization, via minimization of $\boldsymbol{F'(X)}$ (Eqn. 9). These are the DNF iterations alluded to earlier. Furthermore, after each DNF iteration, if either a constraint is under-satisfied (violated) or over-satisfied, then the LM coefficient of that constraint function (composed of its component atomic $\boldsymbol{f(\,)}$ functions described above and shown in Fig. 3(b-c), e.g., the $\boldsymbol{f_{delay}(\,)}$'s along a violating critical path) are either increased or decreased, respectively. Thus the arc-costs in each p-MEA($\boldsymbol{G(f(\,))}$) is correspondingly adjusted. Along with LM update, the current-design options $\boldsymbol{o_{i,c}}$'s are updated, and STA is performed to determine the slew at each gate input. Both these changes cause the arc costs of the partitioned PMEAs to potentially change every iteration in a way that drives the solver towards a more accurate and better solution (constraint-satisfying, if currently violated, and power-minimizing, if constraints are over-satisfied).

Finally, each iteration of DNF, in which an MEA-satisfying near min-cost flow is determined, is itself obtained by multiple iterations of classical min-cost flow (NF). An iteration of NF will not necessarily respect MEA requirements (NF being a continuous solver, its flows can go through multiple arcs in each MEA set). The MEA requirement is then met by pruning out arcs in each MEA set that have 0 or low flow amount through them, thereby converging through multiple NF iterations to full flow going through exactly one arc in each MEA set (and thereby a near min-cost flow). This provides us a valid discrete/OSP solution in the current DNF iteration. The DNF pseudo-code is given in Fig. 5.

5. Incorporating the Cell Replication Option

As explained in Sec. 1 and depicted in Fig. 2, cell replication has the potential to reduce power in solving the ISPD13-H problem. In some cases, without cell replication it is not even possible to satisfy delay constraints due to a consecutive set of sink-size reductions (that can increase their output delays) to satisfy the strict heating-based load constraint of a driver gate. However, cell replication is not a straightforward option to incorporate in the DNF graph since, unlike cell size-Vth options, it is an option that changes the basic structure of the circuit; see Fig. 2(b). As shown in Fig, 4(c), the way we incorporate this option in the DNF graph is for every driver-sink-set combination $(\boldsymbol{g_i}, \boldsymbol{S(g_i)} = \{\boldsymbol{g_{i1}}, \dots, \boldsymbol{g_{im}}\})$ where the driver $\boldsymbol{g_i}$ is targeted for possible replication (when its heat-based load constraint cannot be satisfied easily), we construct the alternate subcircuit $[\boldsymbol{g_{i,1}}, \boldsymbol{S(g_{i,1})}] \cup [\boldsymbol{g_{i,2}}, \boldsymbol{S(g_{i,2})}]$, where $\boldsymbol{g_{i,1}}$ and $\boldsymbol{g_{i,2}}$ are the two replicated versions of $\boldsymbol{g_i}$, with sink sets $\boldsymbol{S(g_{i,1})}$ and $\boldsymbol{S(g_{i,2})}$, resp., and $\boldsymbol{S(g_{i,1})} \cup \boldsymbol{S(g_{i,2})} = \boldsymbol{S(g_i)}$ and $\boldsymbol{S(g_{i,2})}$. The criticalities of paths in terms of their negative slacks and total current input loads are balanced across $\boldsymbol{S(g_{i,1})}$ and $\boldsymbol{S(g_{i,2})}$ (sum of the square of differences in these 2 metrics is minimized). We then connect this replication-based subcircuit's DNF subgraph and the original subcircuit's DNF subgraph via a replication-option 2-arc MEA set $\boldsymbol{Repl(g_i)} = (\boldsymbol{e_{nr}}, \boldsymbol{e_r})$ at their input points: $\boldsymbol{e_{nr}}$, is connected to the original subcircuit, and e_r to its replicated version; see Fig. 4(c).

Further, the fanout p-MEAs from the drivers of $\boldsymbol{g_i}$ to this replication-based DNF subgraph are constructed assuming they drive $\boldsymbol{g_i}$ (instead of its replications) using a copy $\boldsymbol{OS'(g_i)}$ of $\boldsymbol{OS(g_i)}$, as shown in Fig. 4(c). This also means that if the flow goes via $\boldsymbol{e_{nr}}$ to $\boldsymbol{OS(g_i)}$, as opposed to via $\boldsymbol{e_r}$ to $\boldsymbol{OS(g_{i,1})}$ and $\boldsymbol{OS(g_{i,2})}$, then our DNF solver ascertains that the flow selects the same option in $\boldsymbol{OS'(g_i)}$ of $\boldsymbol{OS(g_i)}$ by mimicking the MEA arc prunings and cost-updates (see Fig. 5) of $\boldsymbol{OS'(g_i)}$ in $\boldsymbol{OS(g_i)}$). In order to account for the "missing" p-MEAs from these drivers to $\boldsymbol{OS(g_{i,1})}$ and $\boldsymbol{OS(g_{i,2})}$, there is a *collapsing* cost function $\boldsymbol{CF(OS(g_{i,1}) \cup OS(g_{i,2}))}$ that is the cost of arc $\boldsymbol{e_{nr}}$ through which flow reaches the replicated subcircuit option for $\boldsymbol{g_i}$ (see Fig. 4(c)). A relatively simple version of this function is:

$$\boldsymbol{CF(OS(g_{i,1}) \cup OS(g_{i,2}))} = \underset{\boldsymbol{across\ all\ drivers\ g_j of\, g_i}}{\boldsymbol{a\ weighted\ average\ K}} [\textit{min-cost}\ (\textit{p-MEA}(\boldsymbol{G(f(x_j, x_{i,1}))})) + \textit{min-cost}\ (\textit{p-MEA}(\boldsymbol{G(f(x_j, x_{i,2}))}))] - \underset{\boldsymbol{across\ all\ drivers\ g_j of\, g_i}}{\boldsymbol{K}} [\textit{min-cost}\ (\textit{p-MEA}(\boldsymbol{G(f(x_j, x_i))}))] \tag{10}$$

Thus the $\boldsymbol{CF(\,)}$ cost in arc $\boldsymbol{e_{nr}}$ is the difference between an appropriately averaged cost across all the missing p-MEAs to the OS's of the 2 replicated cells $\boldsymbol{g_{i,1}}$ and $\boldsymbol{g_{i,2}}$, and a similar average for all the present p-MEAs to $\boldsymbol{OS(g_i)}$ (a difference is needed for the flow cost to the replicated subcircuit, since that flow incurs the cost of the p-MEAs to $\boldsymbol{g_i}$, as shown in Fig. 4(c)).

To add another level of complexity to the DNF modeling of cell replication, if any of the drivers $\boldsymbol{g_j}$ of $\boldsymbol{g_i}$ also has a replicated option, then replicated cells of $\boldsymbol{g_i}$ are partitioned across the replicated cells of $\boldsymbol{g_j}$. And the corresponding CF function cost for $\boldsymbol{e_{nr}}$ is augmented to obtain the minimum of the two $\boldsymbol{CF(\,)}$'s for the non-replicated $\boldsymbol{g_j}$ and its replicated version.

6. Experimental Results

The proposed technique was implemented in C++ and the experiments were performed on workstations with Intel(R) Core-i9 10900K @ 3.7GHz CPU with 64 GB memory.

In order to establish the quality of our DNF solver, we first compare in Table 1, our leakage power (LP) results for the original ISPD13 problem that has no heating constraint, to those of the state-of-the-art technique of [7] that we abbreviate as "South Brazil" (SB+) for 8 mainly mid-size to large circuits. There are two versions of results from SB+: one is *SB-LR*, which is a Lagrangian relaxation (LR)-based gate-sizer with an internal STA timer. The other one is *SB+*, which uses the design obtained by SB-LR, with a post-processing stage for further power improvement which interacts with and uses timing information directly from the Synopsys PrimeTime (PT) STA [30]. Our STA (*DNF*-timer) uses the exact same delay and slew models as used by the *SB-LR* STA and recommended by [24]: (1) Elmore-delay for the distributed RC-tree interconnect [28] with (2) the method of [29] to compute the effective capacitance (C_{eff}) for the driver node of a net, and (3) the slew-propagation formulations (Algo. 5 in [7]). Therefore, our DNF results are precisely comparable to SB-LR. Compared to both SB-LR and SB+, it can be seen that DNF has close (mainly) to better power results.

We also note that for circuit *DES-F* (*des-perf-fast*), DNF has a significantly different power performance (42-50% improvement) w.r.t SB-LR/SB+ compared to those for other circuits. The discrepancies of PT with various academic STAs, in spite of the latter following the general delay/slew models of PT, (primarily since PT has not made all parameters of their models clear) are well known [6, 7]. However, to verify that there is no prima facie incorrectness of DNF's DES-F results, we determined its and the related circuit DES-S's (for which DNF's power result is close to SB-LR/SB+) timing and power results using PT. We found that PT provides the same leakage power for the two circuits as our analyzer, and that PT's % timing discrepancies for them w.r.t our internal STA are almost identical. Thus we conclude that our DES-F results don't seem to have any mistake. Nevertheless, in the last row of Table 1, we give the total power results for all circuits except DES-F, and this also shows that our results are very close to those of SB-LR/SB+. We believe this establishes the state-of-the-art quality of the DNF solver (comparable to the current state-of-the-art SB-LR/SB+ solvers).

Next, we discuss the two DNF techniques' results for the heating-based problem which have all the given metric constraints (for delay, slew, and load) of the ISPD13 gate-sizing problem, plus, as discussed in Sec. 1, one additional constraint–the heating-based output-load constraint for all cells. Table 2 shows two versions of the DNF technique for this problem: (1) no cell replication used in DNF (problem ISPD13-H), and (2) cell-replication based DNF (problem ISPD13-HR). As described in Sec. 2-B, we can obtain a heating-based load constraint $C_h(g_i)$ for each cell g_i in a given circuit. However, to streamline the experimentation process, and without any loss of solving generality for DNF, we use a simple and uniform heating-based load constraint C_{L-H} for a circuit C as:

$$C_{L-H}(C) = \alpha \cdot C_{L-max}(C) \qquad (11)$$

where $C_{L-max}(C)$ is the maximum output total-sink-cell load across all cells of C for its final design provided by DNF for the original ISPD13 problem, and α = 0.3 in our experiments—this value of α hits the sweet-spot of being realistic (not making the heat-based load cap constraint intensely hard) but also hard enough to stress the solutions obtained without cell replication.

All the results shown in Table 2 have 0 slew and load violation. First of all, using $C_{L-H}(C)$, we find that about 4.54% of cells violate this load constraint in the final DNF-based designs for the original ISPD13 problem. On the other hand, the DNF solutions explicitly taking the heating-based load constraint into account (problems ISPD13-H and ISPD13-HR) satisfy this constraint for all cells. albeit at a significant cost for ISPD13-H: its solutions violate the delay constraint by an average of 3%. On the other hand, ISPD13-HR solutions have 100% delay satisfaction, underscoring the usefulness of cell replication and its effective deployment by DNF in the face of hard load constraints. We can also see that ISPD13-HR solutions have better power performance than ISPD13-H solutions by more than 4%, again underscoring the efficacy of both cell-replication and DNF. The reason for these results is the conceptual scenarios of circuit performance with and without cell replication described in Secs. 1 and 5. Also, not surprisingly, leakage power consumption increases in ISPD13-H/HR solutions compared to ISPD13 (where there are no heat constraints) by 24.6%/19.6%. This is due to a combination of cell replication and the required cell upsizings when not replicating (see below).Table 2 also shows that while we start with an average of 4.5% of cells violating the heating-based load constraint without any specific consideration of this constraint in the original ISPD13 problem, in the ISPD13-HR solutions only an average of 1.5% of cells are replicated to satisfy this constraint (other cells satisfy it by a combination of sink downsizing, if needed, and their own upsizing and possible upsizings in the fanout cones of its sinks to compensate for the increased delay at sink outputs).

7. Conclusions

We introduced cell replication in the physical synthesis process to better tackle hard heating-based load constraints. To this end, we incorporated the cell replication option in the DNF graph in a complex and innovative structure. Our results establish:

- The state-of-the-art quality of DNF for the original ISPD13 problem, and thus confidence in its QoS for the two heating-constraint based problems.
- The usefulness of incorporating the cell-replication transform in the physical synthesis process in order to satisfy hard I^2R heat dissipation upper bound constraints on interconnects (translated to corresponding upper bound load constraint on the driver cell), and the efficacy of DNF in judiciously deploying this transform: DNF solutions to the heating-based problem using cell replication ISPD13-HR satisfies all hard heating-based load constraints while also satisfying delay and slew constraints with only 1.5% of cells undergoing replication. Further, the solutions for the heating-based problem that does not use cell replication ISPD13-H, while satisfying all load constraints, have an average of 3% delay violation, and also have 4% higher power than ISPD13-HR solutions.
- The general ability of DNF graph modeling for incorporating almost any design options (including those that change the basic structure of the problem/circuit, as does cell replication) in a streamlined manner, and the ability of the DNF solver to efficiently process any DNF graph modeling any problem.

Acknowledgements

This work was funded by NSF grant CCF 2035610.

Table 1: Leakage Power (LP) results for DNF, SB-LR and SB+ [7]. All results have 0 slew and load violations with DNF-timer. A positive (negative) % difference indicates power increase (decrease).

Benchmark	LP (uW)			% diff. DNF vs. others	
	DNF	SB-LR	SB+	SB-LR	SB+
USB-F	1540.5	1545	1554	-0.29%	-0.87%
USB-S	1152	1073.5	1074	7.31%	7.26%
PCI-F	88243	88071	85438	0.20%	3.28%
PCI-S	59327.5	57173	56963	3.77%	4.15%
FFT-F	203059	203927	194307	-0.43%	4.50%
FFT-S	90125.5	87270.5	86600	3.27%	4.07%
DES-F	371289	749955	648824	-50.49%	-42.78%
DES-S	334635	338912	330425	-1.26%	1.27%
TOTAL	**1149372**	**1527927**	**1405183**	**-24.78%**	**-18.20%**
TOTAL w/o DES-F	**778082.5**	**777972**	**756359.5**	**0.01%**	**2.87%**

Table 2: Metrics comparison for heat-based DNF technique between non-replication-cell version (ISPD13-H) and replication-cell version (ISPD13-HR). All results have 0 slew and load violations with DNF-timer, and α = 0.3.

Benchmark	# of Comb cells	delay constr. (ps)	cell heat-load constr. $C_{L\text{-}H}$ (fF)	Max o/p-cell-load in orig. DNF (fF)	% of Cells has heat-load Viol. in orig. DNF	LP (uW)			Max delay Viol. (ps)		Runtime (Hr)		# (%) of Rep cells
						ISPD13-H	ISPD13-HR	% diff. -HR vs. -H	ISPD13 -H	ISPD13 -HR	ISPD13 -H	ISPD13 -HR	ISPD13-HR
USB-F	510	300	34	112	2.15%	1616.5	1575	-2.6%	0.00	0	0.006	0.008	12 (2.4%)
USB-S	510	450	26	87	2.74%	1199	1192	-0.6%	0.00	0	0.002	0.002	5 (1.0%)
PCI-F	28K	750	47	156	4.33%	108971	106441	-2.3%	66.78	0	2.7	2.7	203 (0.7%)
PCI-S	28K	1000	22	72	5.37%	76845	74136	-3.5%	30.51	0	2.7	2.7	141 (0.5%)
FFT-F	31K	1400	72	239	2.44%	246419	239096	-3.0%	55.34	0	3.9	3.9	291 (0.9%)
FFT-S	31K	1800	51	169	3.28%	105387	104327	-1.0%	18.93	0	3.8	3.8	167 (0.5%)
DES-F	104K	1140	92	308	4.85%	483075	458332	-5.1%	50.22	0	7.7	7.5	2396 (2.3%)
DES-S	104K	1300	56	185	5.08%	409508	389714	-4.8%	21.76	0	7.5	8	1621 (1.6%)
TOTAL	**327K**	**8140**	**398**	**1328**	**4.54%**	**1433021**	**1374813**	**-4.1%**	**243.54**	**0**	**28.3**	**28.6**	**4836 (1.5%)**

References

[1] Li, M. (2016). Joule Heating Induced Interconnect Failure in 3D IC Technology,. PhD Thesis, 2016, UCLA. https://escholarship.org/uc/item/1pf907hr

[2] J. Hu et al. Sensitivity-guided metaheuristics for accurate discrete gate sizing. In IEEE/ACM International Conference on Computer-Aided Design, pages 233–239, 2012.

[3] L. Li et al. An efficient algorithm for library-based cell-type selection in high-performance low-power designs. In IEEE/ACM International Conference on Computer-Aided Design, pages 226–232. IEEE, 2012.

[4] V. S. Livramento et al. A hybrid technique for discrete gate sizing based on lagrangian relaxation. ACM Transactions on Design Automation of Electronic Systems, 19(4):40, 2014.

[5] A. Sharma et al. Fast Lagrangian relaxation based gate sizing usingmulti-threading. In IEEE/ACM International Conference on Computer-Aided Design, pages 426–433. IEEE, 2015.

[6] A. Sharma, D. Chinnery, S. Dhamdhere and C. Chu, "Rapid gate sizing with fewer iterations of Lagrangian Relaxation," *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 337-343.

[7] G. Flach et. al. Effective Method for Simultaneous Gate Sizing and V-th Assignment Using Lagrangian Relaxation. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 33(4):546-557, 2014.

[8] F. Gao, and J. P. Hayes, "Total power reduction in CMOS circuits via gate sizing and multiple threshold voltages", in Proc. Des. Autom. Conf., pp. 31-36, 2005.

[9] A. Srivastava, D. Sylvester, and D. Blaauw, "Power minimization using simultaneous gate sizing, dual-Vdd and dual-Vth assignment", in Proc. Des. Autom. Conf., pp. 783-787, 2004.

[10] H. Ren and S. Dutt, "Effective Power Optimization via Simultaneous Vdd, Vth Assignments, Gate-Sizing and Placement Under Timing and Voltage-Island Constraints", *IEEE Trans. CAD*, 30(5): 746-759, May 2011.

[11] D. Chinnery and K. Keutzer, "Linear programming for sizing, vth and vdd assignment," in Proc. International Symposium on Low Power Electronics and Design, 2005, pp. 149–154.

[12] S. Shah, et. al., "Discrete vt assignment and gate sizing using a self-snapping continuous formulation," ICCAD, 2005, pp. 705–712.

[13] H. Chou, Y.-H. Wang, and C. C.-P. Chen, "Fast and effective gate-sizing with multiple-vt assignment using generalized lagrangian relaxation," ASP-DAC, 2005.

[14] S. Roy, et. al., "Numerically convex forms and their application in gate sizing," IEEE Transactions on Computer Aided Design, 2007.

[15] H. Tennakoon and C. Sechen, "Efficient and accurate gate sizing with piecewise convex delay models," DAC, 2005.

[16] H Tennakoon and C Sechen, "Gate sizing using Lagrangian relaxation combined with a fast gradient-based pre-processing step," In Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design, 2002.

[17] J. Wang, D. Das, and H. Zhou, "Gate sizing by lagrangian relaxation revisited," ICCAD, 2007.

[18] S. Hu, M. Ketkar, and J. Hu, "Gate sizing for cell-library-based designs," IEEE Transactions on Computer Aided Design, 2009.

[19] Y Liu and J Hu, "GPU-based parallelization for fast circuit optimization," TODAES, 2011.

[20] Y. Liu and J. Hu. A new algorithm for simultaneous gate sizing and threshold voltage assignment. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(2):223–234, 2010.

[21] M. Ozdal, S. Burns, and J. Hu, "Gate sizing and device technology selection algorithms for high-performance industrial designs," in Proc. Intl. Conf. on Computer-Aided Design, 2011, pp. 724–731.

[22] J. Wang, D. Das, and H. Zhou, "Gate sizing by lagrangian relaxation revisited," ICCAD, 2007.

[23] M. M. Ozdal, et. al., "The ISPD-2012 discrete cell sizing contest and benchmark suite," in Proc. Intl. Symposium on Physical Design, 2012.

[24] M. M. Ozdal et al., "An improved benchmark suite for the ISPD-2013 discrete cell sizing contest," In Proceedings of the 2013 ACM international symposium on International symposium on physical design, pages 168–170. ACM, 2013.

[25] R. K. K. Ahuja, et al., *Network Flows: Theory, Algorithms, and Applications*, Pearson Education, 1993.

[26] A. Nahapetyan, and P. Pardalos, "Adaptive Dynamic Cost Updating Procedure for Solving Fixed Charge Network Flow Problems", *Computational Optimization and Applications journal*, 2008.

[27] R. Puri, D. S. Kung, and A. D. Drumm, "Fast and accurate wire delay estimation for physical synthesis of large ASICs," *GLSVLSI*, 2002.

[28] R. Gupta, et. al., "The elmore delay as bound for rc trees with generalized input signals," in *Proc. ACM/IEEE DAC*, 1995, pp. 364–369.

[29] J. Qian, S. Pullela, and L. Pillage, "Modeling the "effective capacitance" for the RC interconnect of CMOS gates," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 13, no. 12, pp. 1526–1535, Nov. 2006.

[30] https://www.synopsys.com/implementation-and-signoff/signoff/primetime.html