Engineering Change Order for Combinational and Sequential Design Rectification

Jie-Hong R. Jiang
GIEE, EE Dept.
National Taiwan University
Taipei, Taiwan
jhjiang@ntu.edu.tw

Victor N. Kravets
IBM T. J. Watson Research Center
New York, NY
kravets@us.ibm.com

Nian-Ze Lee
GIEE
National Taiwan University
Taipei, Taiwan
nianzelee@gmail.com

***Abstract*—*Engineering change order* (ECO) becomes a crucial element in VLSI design flow to rectify function or fix non-functional requirements in late design stages. Even though commercial ECO solutions are available, ECO remains much room for improvement due to its high computational complexity and stringent physical restrictions. It is under active research and development. In this tutorial, we survey recent developments and list some challenges and future directions to make ECO tools more powerful and practical.**

***Index Terms*—engineering change order, design rectification, patch**

I. Introduction

Engineering change order (ECO) refers to a practice in the VLSI design flow to accommodate specification changes, to rectify functional errors, or to fix non-functional design requirements, such as timing and power, with minimal disturbance to the existing implementation, to save as much as possible the already-spent optimization efforts. Due to the high design complexity and tight design cycle driven by the ever-increasing user-experience demand and time-to-market pressure, it is vital to automate the ECO process. Although EDA tool vendors nowadays do provide commercial ECO solutions, there is still much room for improvement due to ECO's high computational complexity and stringent physical restrictions.

The primary purposes of this tutorial paper are to survey recent work on ECOs at different levels of abstractions, to provide theoretical foundations to some of the problems, and to suggest future directions for research. The literature studied in this paper does not mean to be complete. There are certainly many important related papers left out of our discussion due to the authors' limited knowledge and the page limitation. Nevertheless, we hope the selected coverage and discussion may trigger further efforts to make ECO tools more comprehensive and powerful.

The VLSI design and ECO flows are outlined and related in Figure 1 to facilitate the subsequent discussion about various ECO tasks. Note that the illustration may not meant to perfectly match all the details of industrial practice, but to highlight the essential steps throughout the overall ECO process, and to provide a unified framework to position and compare different endeavors in the ECO research landscape.

The VLSI design flow (the left half of the figure) starts with a specification file of the original design. The original specification is often implemented in some hardware description language by the circuit designer. The design undergoes a sequence of synthesis steps, including register transfer level (RTL) synthesis, logic synthesis, physical design. At the end of the design flow, a layout file of the original design is generated. In general, the physical design

This work was supported in part by the Ministry of Science and Technology of Taiwan. J.H.J. was supported in part by the Alexander von Humboldt Foundation during his visit to the University of Freiburg, Germany.

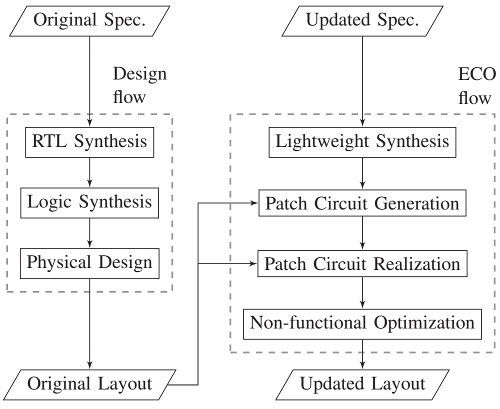

Fig. 1. VLSI design and ECO flows.

stage is the most time-consuming part of the entire toolchain, as it usually requires several iterative optimizations to meet various design objectives.

If an error is found or the specification gets changed at a late stage of the design flow, instead of re-invoking the complete EDA toolchain, the ECO flow (the right half of the figure) will be applied to update the original layout with minimal disturbance, so as to maximally reuse the previously invested optimization efforts. The ECO flow takes as input the original layout file and the updated specification file and first runs RTL and logic synthesis on the latter with lightweight optimization settings to generate a logic gate netlist of the updated design.

Depending on the types of specification changes, design errors, and the applied synthesis techniques in the design flow, the updated logic gate netlist may or may not have one-to-one register correspondence to the original netlist, which gives rise to two paradigms of combinational and sequential design rectifications. While combinational design rectification is more popular in practice due to its rather tractable complexity, sequential design rectification is of research interest as it allows more flexible specification changes and more powerful rectification [1].

Patch circuit generation is executed next, which aims at extracting the logical difference between the original and the updated designs by generating a subcircuit called a *patch circuit* for design rectification. The patch circuit will be realized by integration into the original layout in the step of *patch circuit realization*, where the placement and routing of the original layout are modified locally to implement the patch circuit. After the original layout has been functionally rectified, it will be optimized again for *non-functional design concerns* to resolve potential timing violations or IR drops caused by the realization of the patch circuit. At the end of the ECO flow, an updated layout file with minimal modifications to the original one will be produced for integrated circuit fabrication.

In this work, we refer to the patch circuit generation step as *functional ECO*. In the literature of ECO at the physical design stage, the patch circuit realization step is sometimes also referred to as functional ECO. To avoid confusion, we refer the term "functional ECO" only to the patch generation at the logic synthesis stage, rather than to the patch realization at the physical design stage.

The rest of this paper is organized as follows. After preliminaries are given in Section II, the computational complexity and algorithms for patch circuit generation will be discussed in Section III. Prior works on patch circuit realization and fixing non-functional design concerns in ECO will be reviewed in Section IV and Section V, respectively. Finally, in Section VI we conclude this paper by mentioning some directions for future ECO research.

II. Preliminaries

In ECO, we are concerned with two circuits, the implementation circuit C_F and the golden circuit C_G. In the sequel, we assume C_F and C_G implements Boolean functions $F(X)$ and $G(X)$, respectively. The ECO problem aims to rectify C_F to C'_F such that its new functions $F'(X)$ is equivalent to $G(X)$. The *rectification points* T in circuit C_F refer to the signals whose functions are to be rectified. We abuse the notation and let $F(X,T)$ denote the functions of C_F with its rectification signals being isolated from their driving fanins and being treated as pseudo primary inputs. The rectification corresponds to substituting the rectification signals with a new subcircuit called a *patch*, whose inputs, referred to as the *rectification inputs*, can be primary inputs or existing gate outputs of C_F. The *rectification cones* in C_F, referred to as the transitive fanin cones rooted in the rectification points toward their corresponding rectification inputs, are to be replaced by the patch. Note that the replacement should only affect the rectified signals. That is, the subcircuits in a rectification cone shared by non-rectification signals will not be removed due to the replacement.

A. (Dependency) Quantified Boolean Formula

A dependency quantified Boolean formula (DQBF) Φ can be expressed in the prenex form as

$$\forall X, \exists Y(D).\varphi(X,Y), \tag{1}$$

where $X = \{x_1,\ldots,x_n\}$ and $Y = \{y_1,\ldots,y_m\}$ are sets of Boolean variables, $D = \{D_1,\ldots,D_m\}$ is a set of *dependency sets* with $D_i \subseteq X$ being the *dependency set* of y_i, and φ is a quantifier-free Boolean formula over variables $X \cup Y$. We call $\forall X, \exists Y(D)$ and φ in Eq. (1) the *prefix* and *matrix*, respectively, of the DQBF Φ. In the prefix, the existential quantification is known as the Henkin quantifier [2], which allows the dependency set of an existential variable being explicitly specified.

A DQBF is true (satisfiable) if and only if there exists a set of Boolean functions $f_{y_1},\ldots,f_{y_m}$, called the *Skolem functions*, with f_{y_i} depending on variables D_i, such that substituting every appearance of y_i in φ with f_{y_i} for all $i = 1,\ldots,m$ makes the resultant φ a tautology. Deciding the truth or falsity of DQBFs is NEXPTIME-complete [3].

Quantified Boolean formulas (QBFs) are special cases of DQBFs, where any two dependency sets D_i of y_i and D_j of y_j satisfy either $D_i \subseteq D_j$ or $D_j \subset D_i$. That is, the dependency sets are totally (or linearly) ordered with respect to the inclusion relation, in contrast to the general case of partially ordered dependency sets. The linearly ordered quantification structure of QBFs allows a simplified prefix expression without explicitly specifying the dependency sets. Specifically, the quantifiers of a QBF are ordered in the prefix such that the dependency set of an existential variable y_i consists of the universal variables that appear on the left to y_i in the prefix. Deciding the truth or falsity of QBFs is PSPACE-complete [4], and can be solved more efficiently than general DQBFs.

III. Patch Circuit Generation

The functional ECO problem can be stated as follows.

Problem Statement 1: Given an implementation circuit C_F and a golden circuit C_G with their input and output correspondences known *a priori*, we are asked to rectify C_F to C'_F with minimum changes such that C'_F is functionally equivalent to C_G.

In functional ECO, the main computation task is to generate the patch function. The task can be further divided into two sub-tasks:

1) identity rectification points and their inputs, i.e., rectification cones, and
2) derive patch functions minimal with respect to some cost function.

We note that the minimality of changes to C_F is measured by physical layout changes, and at the logic gate level, it is often approximated by minimizing the patch gate count or other patch cost reflecting physical resources needed for actual implementation.

A. Combinational ECO

Problem Statement 2: The task of combinational ECO (CECO) refers to the functional ECO problem with C_F and C_G are given as combinational circuits.

Note that for sequential circuits C_F and C_G with one-to-one register correspondence, they can be turned into combinational circuits with one-to-one input and output correspondences, so their rectification problem can also be formulated as CECO.

Theorem 1: For the CECO problem, given a set of candidate rectification cones (including rectification points and their respective rectification inputs), testing whether C_F is rectifiable to C_G through the rectification cones is NEXPTIME-complete.

When the candidate rectification cones are specified, the CECO problem reduces to the *topologically constrained logic synthesis problem* [5], [6] or the *partial equivalence checking problem* [7], which can be formulated by the DQBF. As the partial equivalence checking problem is shown to be NEXPTIME-complete [7], the above theorem follows. Specifically the DQBF is of the form

$$\forall X, Y, \exists T(D).(Y \equiv E(X)) \rightarrow (F(X,T) \equiv G(X)), \tag{2}$$

where X is the set of primary input variables, Y the set of internal variables that are rectification inputs in C_F, D consists of $(D_1,\ldots,D_k)$ with $D_i \subset X \cup Y$ being the dependency set of rectification point $t_i \in T$, E corresponds to the defining functions of Y in C_F, and the operator "$\equiv$" denotes elementwise equivalence between its two operands. Note that the formulation is so powerful that a rectification point t_i can functionally depend on another rectification point t_j as one can introduce a fresh new variable $y_j \in Y$, let $y_j \in D_i$, and assert $y_j \equiv t_j$ in the matrix of the DQBF. Figure 2 shows the ECO miter structure that corresponds to the formula of Eq. (2).

The DQBF is true if and only if the CECO problem is rectifiable with respect to the specified rectification cones. The Skolem function

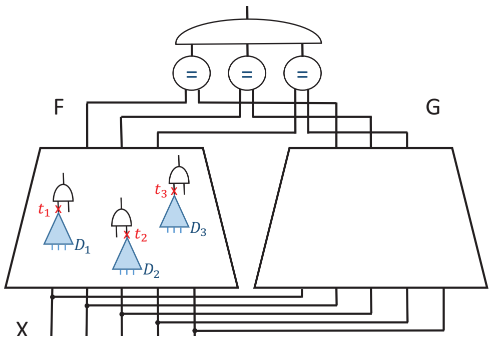

Fig. 2. ECO miter for patch generation.

model to the DQBF corresponds to the patch function. The DQBF based patch generation is purely functional, namely, without referring to the circuit structure of C_G.

However, the high computational complexity of DQBF may not be scalable and thus not suitable for industrial applications. The complexity originates from the Henkin quantifiers for the rectification inputs being explicitly specified. The explicit specification of quantifier dependencies induces a partially ordered quantification structure, which makes the decision procedure exceed the PSPACE complexity.

When only the rectification points are specified while their rectification inputs are relaxed so that there are no individual dependency sets but only a common dependency set, the complexity of the CECO problem can be reduced to PSPACE. Specifically, if the common dependency set is the set of primary inputs, then the CECO problem can be described by the 2QBF [8]

$$\forall X, \exists T. F(X,T) \equiv G(X), \tag{3}$$

where X is the set of primary inputs, and T is the set of rectification points. The QBF is true if and only if the rectification on T is possible and the corresponding Skolem functions of T form the solution patch. Figure 2 shows the construction of Eq. (3) for $D_1 = D_2 = D_3 = X$. On the other hand, when only the rectification points are specified while their rectification inputs are relaxed to a common subset of primary inputs and internal signals, the CECO problem can be described by the following 3QBF, an equivalent but different viewpoint from [9],

$$\forall Y, \exists T, \forall X'. (Y \equiv E(X)) \rightarrow (F(X,T) \equiv G(X)), \tag{4}$$

where Y is a selected subset of primary inputs and internal signals of C_F, $X' = X \setminus Y$, and E corresponds to the defining functions of Y in C_F. Again the QBF is true if and only if the rectification on T is possible and the corresponding Skolem functions of T form the solution patch. Figure 2 shows the construction of Eq. (4) for $D_1 = D_2 = D_3 = Y$. Note that the above reduction to QBF is possible because the rectification points T all share the same dependency set, which makes the quantification structure totally ordered. In contrast, if every rectification point has its own dependency set, then the quantification structure becomes partially ordered and a DQBF is needed to prevent exponential explosion of formula size.

Another way to cope with the DQBF complexity is to make the rectification cones have a fixed small number of inputs such that the truth tables of the Skolem functions can be explicitly encoded in an exist-forall 2QBF form [10]. Note that, unlike the forall-exist quantification order of Eq. (3), the exist-forall quantification order of the 2QBF is resulted from the first-order to second-order logic conversion, stating the existence of Skolem functions (represented in truth table variables) that work correctly under all input assignments.

To further reduce QBF to SAT solving, the single rectification point formulation using Craig interpolation [11] can be exploited and extended to multiple rectification points. In this case, the Skolem functions for the rectification points of Eq. (3) can be obtained iteratively [12] through a sequence of quantifier elimination by functional substitution [13]. Furthermore, the obtained Skolem functions can be re-expressed by internal signals [14] through the functional dependency computation [15].

In addition to the above functional methods, it is often necessary to exploit circuit structures to enhance scalability. As the golden circuit serves as a functional representation of a new specification, we should bear in mind that it may or may not exhibit structural similarity to the implementation circuit. Nevertheless, when the golden circuit C_G is synthesized to the gate level by the same tool as the implementation circuit C_F, often it is reasonable to assume that the two circuits exhibit structural similarities to some extent provided that their functions do not deviate significantly. In the DeltaSyn [16] framework, the logic difference between C_F and C_G is computed by both functional and structural methods in two directions: From input to output, identify forward frontier equivalent signals between C_F and C_G as the candidate rectification inputs to the patch. From output to input, identify backward frontier isomorphic signals as the candidate rectification points. The forward direction can be done by structural hashing and functional sweeping technique [17], [18]. The backward direction was improved in [19] by Boolean matching, and in [20] by adding hints to annotate functionally changed signals. In [21], the rectification points and their rectification inputs are determined based on the rewiring strategy.

B. Sequential ECO

Problem Statement 3: The problem of sequential ECO (SECO) refers to the functional ECO problem with C_F and C_G are given as sequential circuits.

Theorem 2: For the SECO problem, given a set of candidate rectification cones each with its number of allowed flip-flops being specified, testing whether C_F is rectifiable to C_G through the rectification cones is NEXPTIME-compete.

Notice that each rectification cone $t_i \in T$ with dependency set D_i is allowed to be rectified by a sequential circuit with flip-flops. In this case, D_i will be augmented with the pseudo-primary inputs formed by the flip-flop outputs. Also the rectification points T will be augmented with the pseudo-primary outputs formed by the flip-flop inputs. In SECO, the rectification cones in Figure 2 may be substituted by sequential subcircuits with flip-flops.

Proposition 1: Given a sequential circuit with transition function Δ, initial states I, and bad states B, B is not reachable from I if and only if there exists an inductive state set Q such that $I \subseteq Q$, $Q \cap B = \emptyset$, and Q is closed under transition, i.e., $q' = \Delta(x,q) \in Q$ holds for any input x and $q \in Q$.

By Proposition 1, the equivalence between the rectified C'_F and C_G can be formulated by the following DQBF [22].

$$\forall X, Y, S_1, S_2, S'_1, S'_2, \exists T(D), Q(S_1 \cup S_2), Q'(S'_1 \cup S'_2). \tag{5}$$

$$(I(S_1, S_2) \rightarrow Q) \wedge \tag{6}$$

$$(Q \wedge (Y \equiv E(X, S_1)) \wedge R(X, S_1, S_2, S'_1, S'_2) \rightarrow Q') \wedge \tag{7}$$

$$(Q \rightarrow (F(X, S_1, T) \equiv G(X, S_2))) \wedge \tag{8}$$

$$(((S_1, S_2) \equiv (S'_1, S'_2)) \rightarrow (Q \equiv Q')) \tag{9}$$

where X, Y, S_1, S_2, S'_1, and S'_2 are the primary input variables, internal variables referred to by the rectification cones of C_F, current-state variables of C'_F, current-state variables of C_G, next-state variables of C'_F, and next-state variables of C_G; the set D consists of $(D_1, \ldots, D_k)$ with $D_i \subset X \cup Y \cup S_1$ being the dependency set of $t_i \in T$; Q and Q' are two new fresh variables quantified existentially with dependency sets $S_1 \cup S_2$ and $S'_1 \cup S'_2$, respectively, whose Skolem functions correspond to the characteristic functions representing a common unknown inductive state set of the product machine of C'_F and C_G; predicate I characterizes the initial states of the product machine; predicate $R = (S'_1 \equiv \Delta_1(X, S_1, T)) \wedge (S'_2 \equiv \Delta_2(X, S_2))$ for Δ_1 and Δ_2 being the transition functions of C'_F and C_G, respectively, characterizes the transition relation of the product machine. Eq. (6) asserts that all initial states are in Q; Eq. (7) asserts that Q is transition closed; Eq. (8) asserts that the outputs of the rectified circuit and the golden circuit are equivalent under all Q states; Eq. (9) asserts that Q and Q' characterize the same state set. As the rectification cone specified SECO problem can be formulated in DQBF, it is certainly in NEXPTIME. On the other hand, it subsumes the CECO counterpart, and thus is NEXPTIME-complete. This complexity result is somewhat surprising as the sequential generalization does not increase the theoretical complexity.

Apart from the above theoretical result, in practice SECO is much harder and rarely investigated. As general sequential equivalence checking is PSPACE-complete, much harder than the coNP-complete combinational equivalence checking, it is motivating to consider special sequential synthesis techniques for practical verification [23]. In [1], SECO for circuits under the retiming and resynthesis transformation is considered. Assuming that the implementation circuit is undergone retiming and possibly additional resynthesis transformation, the register correspondence between the implementation circuit and the golden circuit is rediscovered, which simplifies the SECO problem to two CECO sub-problems.

IV. Patch Circuit Realization

In this section, we discuss the realization of the patch circuit generated by the upstream functional ECO step. Depending on the moment when an ECO is required, patch realization is divided into two categories: *pre-mask* and *post-mask*.

If an ECO is needed *before* the masks are fabricated, incremental physical design [24] can be used to realize the change in the original layout with minimal disturbance to preserve previous optimization efforts. We refer to this type of patch realization as pre-mask. On the other hand, if an ECO is needed *after* the masks are fabricated, to avoid the re-spin delay and cost, *spare cells*, which are unused standard cells uniformly distributed across the design, come to rescue and make the change realizable by only re-spinning the less expensive metal-layer masks [25], [26]. This type of realization is referred to as post-mask. We remark that, spare cells are also of great use in the non-functional optimization, which will be discussed in Section V.

A. Pre-mask Realization

For incremental placement, an ECO-system was devised in [27], which took an initial placement and applied minimal replacement of cells to realize the new netlist. It achieved efficient incremental placement and little increase in wire length. Recently, a timing-driven incremental placement technique [28] has been proposed to accelerate the timing optimization process to relieve the time-to-market pressure. Over a set of benchmarks consisting of 14-nm high-performance commercial microprocessors, the technique is able to improve the worst negative slacks significantly. In addition to placement, incremental remapping is also explored to improve timing along critical paths [29]. A constrained placement strategy that considers only the neighboring gates of a target gate is adopted to ease the runtime overhead incurred by integrating an accurate timing analysis based on fast detailed routing.

Authors in [30] pioneered an incremental routing (ECO routing) by proposing an algorithm for implicit connection graph maze routing. The algorithm is built on a tree-like data structure that has compact memory footprint and is able to answer maze expansion queries efficiently. In [31], a tile-based routing technique was proposed, which achieved scalability by using a routing graph reduction method. Timing refinement [32] and redundant-via-aware [33] incremental routing techniques have also been studied.

B. Post-mask Realization

Early works [34], [35] considered *constant insertion*, i.e., connecting inputs of a spare cell to either `VDD` or `GND` signals, for increasing the flexibility to realize a patch circuit. In [36], a flow is proposed to remap the patch circuit and assign the mapped gates to the already-placed spare cells. It emphasizes the notion of *resource-feasibility*, which accounts for the spare cells supply, types, and locations during the remapping process. Simulated annealing was applied as the underlying optimization technique to minimize the total wire length of the rectified design.

Deterministic techniques based on stable matching algorithms have been discussed in [37], [38] to guarantee consistent quality of results. To overcome the functionality and location constraints of pre-placed standard cells, metal-configurable gate-array spare cells were invented, and authors in [39] addressed the post-mask patch realization using these configurable spare cells. The significant technical challenges include the fragmentation of spare gate arrays to accommodate required functions and the congestion control for routability. Mixed-integer linear programming (MILP) is used to formulate and solve the spare array assignment and packing problem.

Authors in [40] observe that separating the concern of timing closure from patch realization sometimes fails to fix all timing violations, and show how to simultaneously perform both by an augmented bipartite graph formulation.

V. Non-Functional Design Concerns

In this section, we discuss how to use spare cells to twist a functionally rectified implementation to meet other non-functional design concerns, such as timing closure or power dissipation. Note that these non-functional concerns can also be taken into account during patch circuit realization, as shown in [40]. In the following, we discuss prior works that mainly focus on resolving timing violations or reducing IR drops.

A. Timing

In the literature, techniques to fix timing violations after patch circuit realization are often called *timing ECO*.

Spare-cell rewiring was first formulated in [41], [42] to optimize timing in post-mask ECO. Compared to existing rewiring-based incremental timing optimization, the spare-cell rewiring has intrinsically dynamic wiring costs: A rewiring operation would turn some spare cells into ordinary standard cells and vice versa, hence dynamically changing the wiring costs. Dynamic programming was used to search for optimal rewiring choices. Redundant wires, or dummy metals, have also been considered to minimize the mask cost in [43]. Besides negative slacks, path smoothness was taken as a new metric in [44] to reduce the number of timing ECO runs.

Technology remapping was used to resolve timing violations in [45]. Instead of taking the set of standard library cells, a limited set of spare cells is considered with dynamic changes in wiring costs, and the timing critical regions are iteratively remapped to resolve timing violations. This method also supported timing-aware patch circuit realization.

In addition to the combinational part, authors in [46] explored the sequential domain of a design and adopted clock scheduling to resolve timing violations. MILP was used to formulate both constrained spare-cell resources and changes in wiring costs. The technique is orthogonal to combinational methods and can be combined for better timing performance.

Besides timing violations, *input-slew* and *output-loading* violations were considered in [47] to ensure accurate delay estimation and guarantee the yield of the design.

B. Power

Although scattering spare cells onto the chip provides the flexibility to accommodate a late design change, it comes at the cost of increasing area and power leakage. On the other hand, the IR drop is becoming a severe issue as the supply voltage in advanced manufacturing technology nodes keeps decreasing. In [48], [49], the authors designed a reconfigurable spare cell acting both as an ordinary spare cell and a decoupling capacitor cell, which helps to reduce IR drops.

The authors in [50] proposed an approach to repair IR-drop failures by modifying the power delivery network. They adopted multiple-objective optimization to minimize both IR drops and routing congestion, and relied on the greedy Pareto optimal method to search for the Pareto optimal front.

VI. Challenges and Future Directions

Although there have been extensive efforts aiming to address various ECO problems, the challenges remain. We mention some future directions for research.

1) *Tight integration between logic and physical ECO*: To date, ECOs at logic and physical levels are often performed separately with minimal coupling. A similar situation, known as the design closure problem, is encountered in the design flow. On the one hand, logic level ECO provides more freedom for rectification; on the other hand, physical level ECO provides better resource estimation. Design rectification has the potential of mainly benefiting from tight integration of ECOs at different abstraction levels.

2) *Learning from benchmarks:* Design errors or specification changes in industrial designs may exhibit specific common hidden patterns. Examining industrial benchmarks and exploiting their peculiarities may improve computation efficiency and rectification quality. A preliminary study is performed in [9] predominantly through manual efforts. The tasks of ECO benchmark analysis and rectification model building may be made more automatic via machine learning algorithms. Building meaningful ECO rectification models, similar to fault modeling and diagnosis [51], may potentially reduce the search space and patch size.

3) *Efficient rectification cone identification:* Searching rectification points and their inputs is one of the most crucial steps in ECO. Simulation can be an effective method to enhance the search efficiency [21], where no optimization to the simulation is attempted. Devising intelligent simulation strategies may achieve further improvements.

4) *Sequential rectification:* Although SECO has the same complexity as CECO when the rectification cones and their numbers of flip-flops are given, SECO in practice is much harder than CECO and remains largely unexplored. How to exploit sequential flexibility for design rectification is a challenging open problem.

5) *Design for Rectifiability* Similar to design for testability, one can make a design more rectifiable by adding not only spare cells but also programmable or reconfigurable components. How to add minimal reconfigurable components while achieving maximal rectifiability is an interesting subject for investigation. Moreover, design for rectifiability has its potential to lift ECO at a higher abstraction level [52].

6) *ECOs for emerging system design principles and technologies:* In the post Moore's Law era, new methodologies for system design, such as approximate design, probabilistic design, stochastic computing, and neuromorphic computation, are emerging. In addition, new technologies beyond conventional CMOS logic are under active research. Their design rectification may, in turn, require new computation models and engines. E.g., ECO for stochastic systems or probabilistic design may require a new logic formalism [53] beyond DQBF.

7) *Software workarounds:* When hardware rectification to a design is not possible, software workarounds may be considered, provided that the design runs software applications. This direction is pursued in [54]. Further advancements are needed for practical industrial applications.

Acknowledgment

J.H.J. thanks Christoph Scholl for hosting his visit to the University of Freiburg, Germany, September, 2019 – February, 2020. This work was done during the visit.

References

[1] N.-Z. Lee, V. N. Kravets, and J.-H. R. Jiang, "Sequential engineering change order under retiming and resynthesis," in *Proceedings of International Conference on Computer Aided Design*, pp. 109–116, 2017.

[2] L. Henkin, "Some remarks on infinitely long formulas," in *Infinitistic Methods, Proceedings of Symposium on Foundations of Mathematics*, pp. 167–183, 1961.

[3] G. Peterson, J. Reif, and S. Azhar, "Lower bounds for multiplayer noncooperative games of incomplete information," *Computers & Mathematics with Applications*, vol. 41, no. 7, pp. 957 – 992, 2001.

[4] L. J. Stockmeyer and A. R. Meyer, "Word problems requiring exponential time," in *Proceedings of Symposium on Theory of Computing*, pp. 1–9, 1973.

[5] S. Sinha, A. Mishchenko, and R. K. Brayton, "Topologically constrained logic synthesis," in *Proceedings of International Conference on Computer Aided Design*, pp. 679–686, 2002.

[6] V. Balabanov, H.-J. K. Chiang, and J.-H. R. Jiang, "Henkin quantifiers and Boolean formulae: A certification perspective of DQBF," *Theoretical Computer Science*, vol. 523, pp. 86–100, 2014.

[7] K. Gitina, S. Reimer, M. Sauer, R. Wimmer, C. Scholl, and B. Becker, "Equivalence checking of partial designs using dependency quantified Boolean formulae," in *Proceedings of International Conference on Computer Design*, pp. 396–403, 2013.

[8] K.-F. Tang, P.-K. Huang, C.-N. Chou, and C.-Y. R. Huang, "Multi-patch generation for multi-error logic rectification by interpolation with cofactor reduction," in *Proceedings of Design, Automation & Test in Europe Conference & Exhibition*, pp. 1567–1572, 2012.

[9] V. N. Kravets, J.-H. R. Jiang, and H. Riener, "Learning to automate the design updates from observed engineering changes in the chip development cycle," in *Proceedings of Design, Automation & Test in Europe Conference & Exhibition*, 2020.

[10] S. Jo, T. Matsumoto, and M. Fujita, "SAT-based automatic rectification and debugging of combinational circuits with LUT insertions," in *Proceedings of Asian Test Symposium*, pp. 19–24, 2012.

[11] B.-H. Wu, C.-J. Yang, C.-Y. R. Huang, and J.-H. R. Jiang, "A robust functional ECO engine by SAT proof minimization and interpolation techniques," in *Proceedings of International Conference on Computer Aided Design*, pp. 729–734, 2010.

[12] A. Q. Dao, N.-Z. Lee, L.-C. Chen, M. P.-H. Lin, J.-H. R. Jiang, A. Mishchenko, and R. K. Brayton, "Efficient computation of ECO patch functions," in *Proceedings of Design Automation Conference*, pp. 51:1–51:6, 2018.

[13] J.-H. R. Jiang, "Quantifier elimination via functional composition," in *Proceedings of International Conference on Computer Aided Verification*, pp. 383–397, 2009.

[14] H.-T. Zhang and J.-H. R. Jiang, "Cost-aware patch generation for multi-target function rectification of engineering change orders," in *Proceedings of Design Automation Conference*, pp. 96:1–96:6, 2018.

[15] J.-H. R. Jiang, C.-C. Lee, A. Mishchenko, and C.-Y. R. Huang, "To SAT or not to SAT: Scalable exploration of functional dependency," *IEEE Transactions on Computers*, vol. 59, no. 4, pp. 457–467, 2010.

[16] S. Krishnaswamy, H. Ren, N. Modi, and R. Puri, "DeltaSyn: An efficient logic difference optimizer for ECO synthesis," in *Proceedings of International Conference on Computer Aided Design*, pp. 789–796, 2009.

[17] Q. Zhu, N. Kitchen, A. Kuehlmann, and A. Sangiovanni-Vincentelli, "SAT sweeping with local observability don't-cares," in *Proceedings of Design Automation Conference*, pp. 229–234, 2006.

[18] A. Mishchenko, S. Chatterjee, J.-H. R. Jiang, and R. K. Brayton, "FRAIGs: A unifying representation for logic synthesis and verification," 2005. ERL Technical Report.

[19] S.-L. Huang, W.-H. Lin, P.-K. Huang, and C.-Y. R. Huang, "Match and replace: A functional ECO engine for multierror circuit rectification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 3, pp. 467–478, 2013.

[20] H. Ren, R. Puri, L. Reddy, S. Krishnaswamy, C. Washburn, J. Earl, and J. Keinert, "Intuitive ECO synthesis for high performance circuits," in *Proceedings of Design, Automation & Test in Europe Conference & Exhibition*, pp. 1002–1007, 2013.

[21] V. N. Kravets, N.-Z. Lee, and J.-H. R. Jiang, "Comprehensive search for ECO rectification using symbolic sampling," in *Proceedings of Design Automation Conference*, pp. 71:1–71:6, 2019.

[22] C. Scholl and R. Wimmer, "Dependency quantified Boolean formulas: An overview of solution methods and applications - extended abstract," in *Proceedings of International Conference on Theory and Applications of Satisfiability Testing*, pp. 3–16, 2018.

[23] J.-H. R. Jiang and W.-L. Hung, "Inductive equivalence checking under retiming and resynthesis," in *Proceedings of International Conference on Computer Aided Design*, pp. 326–333, 2007.

[24] J. Cong and M. Sarrafzadeh, "Incremental physical design," in *Proceedings of International Symposium on Physical Design*, pp. 84–92, 2000.

[25] K.-H. Chang, I. L. Markov, and V. Bertacco, "Reap what you sow: Spare cells for post-silicon metal fix," in *Proceedings of International Symposium on Physical Design*, pp. 103–110, 2008.

[26] C.-Y. Tan and I. H.-R. Jiang, "Recent research development in metal-only ECO," in *Proceedings of International Midwest Symposium on Circuits and Systems*, pp. 1–4, 2011.

[27] J. A. Roy and I. L. Markov, "ECO-system: Embracing the change in placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 12, pp. 2173–2185, 2007.

[28] J. Jung, G.-J. Nam, L. N. Reddy, I. H.-R. Jiang, and Y. Shin, "OWARU: Free space-aware timing-driven incremental placement with critical path smoothing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 9, pp. 1825–1838, 2017.

[29] A. Kumar, T.-H. Wu, and A. Davoodi, "SynECO: Incremental technology mapping with constrained placement and fast detail routing for predictable timing improvement," in *Proceedings of International Conference on Computer Design*, pp. 551–556, 2008.

[30] J. Cong, J. Fang, and K.-Y. Khoo, "An implicit connection graph maze routing algorithm for ECO routing," in *Proceedings of International Conference on Computer Aided Design*, pp. 163–167, 1999.

[31] Y.-L. Li, J.-Y. Li, and W.-B. Chen, "An efficient tile-based ECO router using routing graph reduction and enhanced global routing flow," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 345–358, 2007.

[32] M. A. Kazda, Z. Li, G.-J. Nam, and Y. Zhou, "Timing refinement re-routing," 2014. US Patent 8,635,577.

[33] H.-A. Chien and T.-C. Wang, "Redundant-via-aware ECO routing," in *Proceedings of Asia and South Pacific Design Automation Conference*, pp. 418–423, 2014.

[34] Y.-M. Kuo, Y.-T. Chang, S.-C. Chang, and M. Marek-Sadowska, "Engineering change using spare cells with constant insertion," in *Proceedings of International Conference on Computer Aided Design*, pp. 544–547, 2007.

[35] Y.-M. Kuo, Y.-T. Chang, S.-C. Chang, and M. Marek-Sadowska, "Spare cells with constant insertion for engineering change," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 3, pp. 456–460, 2009.

[36] N. A. Modi and M. Marek-Sadowska, "ECO-map: Technology remapping for post-mask ECO using simulated annealing," in *Proceedings of International Conference on Computer Design*, pp. 652–657, 2008.

[37] I. H.-R. Jiang, H.-Y. Chang, L.-G. Chang, and H.-B. Hung, "Matching-based minimum-cost spare cell selection for design changes," in *Proceedings of Design Automation Conference*, pp. 408–411, 2009.

[38] I. H.-R. Jiang and H.-Y. Chang, "ECOS: Stable matching based metal-only ECO synthesis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 3, pp. 485–497, 2011.

[39] H.-Y. Chang, I. H.-R. Jiang, and Y.-W. Chang, "ECO optimization using metal-configurable gate-array spare cells," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 11, pp. 1722–1733, 2013.

[40] H.-Y. Chang, I. H.-R. Jiang, and Y.-W. Chang, "Simultaneous functional and timing ECO," in *Proceedings of Design Automation Conference*, pp. 140–145, 2011.

[41] Y.-P. Chen, J.-W. Fang, and Y.-W. Chang, "ECO timing optimization using spare cells," in *Proceedings of International Conference on Computer Aided Design*, pp. 530–535, 2007.

[42] K.-H. Ho, Y.-P. Chen, J.-W. Fang, and Y.-W. Chang, "ECO timing optimization using spare cells and technology remapping," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 5, pp. 697–710, 2010.

[43] S.-Y. Fang, T.-F. Chien, and Y.-W. Chang, "Redundant-wires-aware ECO timing and mask cost optimization," in *Proceedings of International Conference on Computer Aided Design*, pp. 381–386, 2010.

[44] H.-Y. Chang, I. H.-R. Jiang, and Y.-W. Chang, "Timing ECO optimization via Bézier curve smoothing and fixability identification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 12, pp. 1857–1866, 2012.

[45] K.-H. Ho, J.-H. R. Jiang, and Y.-W. Chang, "TRECO: Dynamic technology remapping for timing engineering change orders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 11, pp. 1723–1733, 2012.

[46] K.-H. Ho, X.-W. Shih, and J.-H. R. Jiang, "Clock rescheduling for timing engineering change orders," in *Proceedings of Asia and South Pacific Design Automation Conference*, pp. 517–522, 2012.

[47] C.-P. Lu, C.-T. Chao, C.-H. Lo, and C.-W. Chang, "A metal-only-ECO solver for input-slew and output-loading violations," in *Proceedings of International Symposium on Physical Design*, pp. 191–198, 2009.

[48] H.-T. Chen, C.-C. Chang, and T. Hwang, "Reconfigurable ECO cells for timing closure and IR drop minimization," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 18, no. 12, pp. 1686–1695, 2009.

[49] H.-T. Chen, C.-C. Chang, and T. Hwang, "New spare cell design for IR drop minimization in engineering change order," in *Proceedings of Design Automation Conference*, pp. 402–407, 2009.

[50] T.-W. Tseng, C.-T. Lin, C.-H. Lee, Y.-F. Chou, and D.-M. Kwai, "A power delivery network (PDN) engineering change order ECO approach for repairing IR-drop failures after the routing stage," in *Proceedings of International Symposium on VLSI Design, Automation and Test*, pp. 1–4, 2014.

[51] A. Smith, A. Veneris, M. F. Ali, and A. Viglas, "Fault diagnosis and logic debugging using Boolean satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 10, pp. 1606–1621, 2005.

[52] Q. Wang, A. M. Gharehbaghi, T. Matsumoto, and M. Fujita, "High-level engineering change through programmable datapath and SMT solvers," in *Proceedings of International Symposium on Circuits and Systems*, pp. 1–5, 2019.

[53] N.-Z. Lee and J.-H. R. Jiang, "Dependency stochastic Boolean satisfiability: A logical formalism for NEXPTIME decision problems with uncertainty," *arXiv preprint arXiv:1911.04112*, 2019.

[54] T.-P. Liu, S.-R. Lin, and J.-H. R. Jiang, "Software workarounds for hardware errors: Instruction patch synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 12, pp. 1992–2003, 2013.