ECOS: Stable Matching Based Metal-Only ECO Synthesis

Iris Hui-Ru Jiang, *Member, IEEE*, and Hua-Yu Chang

***Abstract*—To ease the time-to-market pressure and save the photomask cost, metal-only ECO realizes the last-minute design changes by revising the photomasks of metal layers only. This task is challenging because the pre-injected spare cells are limited in number and in cell types. Metal-only ECO has to implement these functional and/or timing changes using available spare cells. In this paper, we propose a stable matching based metal-only ECO synthesizer, named ECOS, that can implement the incremental design changes correctly without sacrificing timing and routability. The experiments are conducted on nine industrial testcases. These testcases reflect the real difficulties faced by designers and our results show that ECOS is promising for all of them.**

***Index Terms*—Metal-only ECO, resynthesis, spare cells, stable matching, technology remapping.**

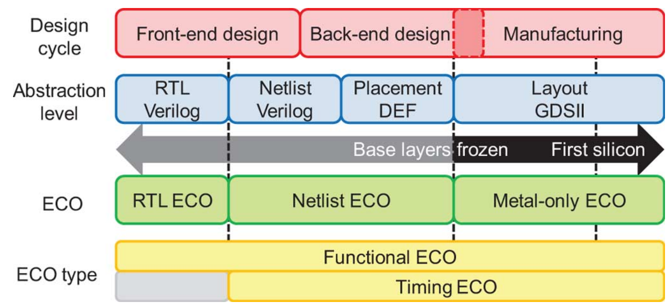


Fig. 1. ECO is a process that applies incremental design changes without backtracking to earlier design stages. Metal-only ECO means the ECO performed after the base layers are frozen. Routing can be done in parallel with the manufacturing of base layers.

I. Introduction

ENGINEERING change order (ECO) is a process that applies incremental design changes without backtracking to earlier design stages. As shown in Fig. 1, ECO is generally classified into two types. Functional ECO can be used to fix bugs or revise specification, while timing ECO targets to improve input slew, output loading and delay. Instead of rebuilding the design from scratch, ECO can not only shorten the design and fabrication time but also save the cost. The later the stage where ECO is performed, the fewer resources are available and the greater challenges can be met. Before the base layers (placement) are frozen, ECO can be performed on the gate-level netlist and/or the RTL code. At this stage, we can freely insert or move cells to complete ECO. However, after that, ECO can be realized by modifying only the photomasks of metal layers (routing). Hence, ECO performed after base layers are frozen, or even after the first silicon chips are produced, is referred to as metal-only ECO.

After the base layers are frozen, routing can be done in parallel with the manufacturing of base layers and thus the fabrication time can be shortened. After the first silicon chips are produced, we can save the photomask cost by reusing the base-layer part in the next tape-out. For nanotechnologies, the complete photomask set may cost over one million USD and the cost keeps increasing rapidly [1]. In addition, the photomasks for cells, including the base layers and low metal layers, dominate the photomask cost [2]. Hence, metal-only ECO is popular because of its cost effectiveness.

To facilitate metal-only ECO, a design is sprinkled with spare (redundant) cells at placement. Since spare cells are pre-injected, they are limited both in number and in cell types. In addition, to prevent floating signals, the inputs of spare cells are tied to either logic high or low. ECO is then performed by rewiring the inputs and outputs of spare cells.

Good metal-only ECO relies on the following four techniques (see Fig. 2).

1) Sufficient and evenly distributed spare cells: Spare cells should be uniformly spread over the whole design to accommodate sufficient resources for ECO at every possible location [3] and [4].
2) A good ECO router: The rewiring of inputs and outputs is done by routing. The routing can be done either by the normal mode or by the incremental mode. The incremental ECO router has to handle tremendous obstacles (existing routing patterns) and design rules and completes routing with the least change [5].
3) An efficient ECO list: The ECO list is a list of functional changes, i.e., the logic difference between the original design and the revised specification [6]. A short ECO list usually leads to a high ECO feasibility.
4) A powerful ECO synthesizer: To correctly and effectively fulfill the revisions on functionality and/or timing, the ECO synthesizer is required to utilize the physical information

Manuscript received July 09, 2010; revised October 24, 2010; accepted December 13, 2010. Date of publication February 04, 2011; date of current version February 17, 2012. An earlier version of this paper was presented at 46th ACM/IEEE Design Automation Conference, San Francisco, CA, July 2009 and was demonstrated at the IEEE International Symposium on Circuits and Systems, Paris, France, May 2010.

I. H.-R. Jiang is with the Department of Electronics Engineering and Institute of Electronics, National Chiao Tung University, Hsinchu 30010, Taiwan (e-mail: huiru.jiang@gmail.com).

H.-Y. Chang is with the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan (e-mail: huayu.chang@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2011.2104377

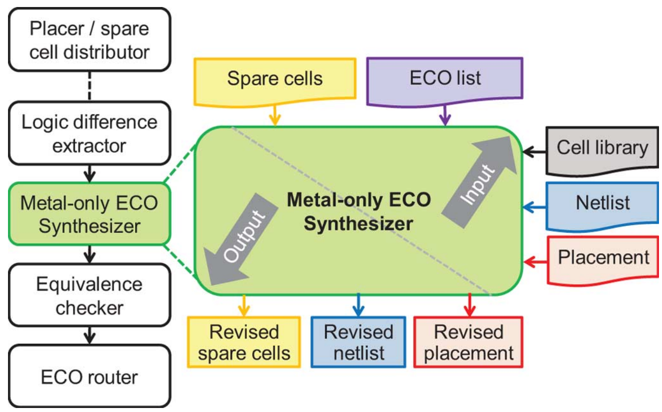


Fig. 2. Metal-only ECO flow. In this paper, we focus on the metal-only ECO synthesizer.

of spare cells, including locations and cell types, to guide technology remapping very well. In addition, due to limited resources, the shortage of spare cells may be severe, especially for late ECO runs. The selected spare cells may deviate from the ideal locations resulting in timing criticality and routing congestion.

For metal-only ECO synthesis, most prior endeavors focus either on timing ECO or on functional ECO, e.g., [7]–[11]. For timing ECO, Chen *et al.* proposed a technology-remapping technique based on dynamic programming [7]. Lu *et al.* revealed that most timing violations resulted from the excessive input slew and output loading [8]. Thus, by connecting spare cells onto the violated nets as buffers, they fix input-slew and output-loading violations first and then handle delay issues. Ho *et al.* iteratively restructured sub-circuits by a set of pre-computed circuit templates to improve timing [9]. On the other hand, for functional ECO, Kuo *et al.* replaced the original cells with spare ones whose inputs could be inserted with constant values (logic high/low) [10]. Constant insertion[1] increases the capability of spare cells. Modi and Marek-Sadowska incorporated [10] to a simulated annealing framework to enhance ECO feasibility [11].

In this paper, we propose a stable matching based metal-only ECO synthesizer to complete the functional changes using available spare cells without sacrificing timing and routability. As shown in Fig. 2, given the netlist and placement of a design, available spare cells and an ECO list (logic difference between the revised specifications and the original design), ECOS completes all functional changes described in the ECO list by rewiring spare cells with the minimum cost. Considering timing and routability, without loss of generality, the cost of the metal-only ECO synthesis problem is then modeled by the summation of the half-perimeter wirelength (HPWL) over all nets in the revised design. This metric is widely used to measure the total wirelength in physical design. The smaller HPWL generally implies the shorter interconnect delay and better routability [14]. Furthermore, the findings in [19]–[21] reveal that the congestion-weighted HPWL has a strong correlation with the routed wirelength. Based on the revised netlist generated by ECOS, an ECO router then completes the routing and generates the revised photomasks of metal layers.

[1]Constant insertion maximizes the functionality of a cell, e.g., a 2-input NAND cell of functionality $O = (I_1 \bullet I_2)'$ can be an INV (inverter) cell, if one input is tied to logic high.

ECOS, first of all, resynthesizes the given ECO list using affordable spare cell types with geometry proximity consideration. Secondly, each instance in the resynthesized list is replaced by an adequate spare cell, as well as the related nets are reconnected. The spare cells are selected based on the stable matching algorithm which solves the competition among several functional changes to one spare cell [15]. Moreover, the unobservable cells resulted from ECO can be freed up and constant insertion is applied to increase flexibility. Afterwards, formal equivalence checking can be performed to verify whether the revised design matches the revised functionality.

ECOS has the following features.

1) It completes functional ECO without sacrificing timing and routability.
2) It integrates physical information into resynthesis. We can quantify the impact on HPWL throughout the entire flow.
3) It solves the competition among functional changes by stable matching instead of the nondeterministic approach.
4) It can readily extend to congestion-driven ECO.
5) It handles non-tree type spare cells and ECO functions.
6) It considers constant value insertion for spare cells.
7) It recycles freed-up cells for the current or subsequent ECO runs.
8) It easily collaborates with existing synthesizers.

The experiments are conducted on nine industrial testcases. These designs reflect a variety of difficulties faced by designers. Compared with the automated traditional ECO synthesis flow and an ECO flow based on prior work [10], ECOS is promising. Moreover, utilizing the congestion map, ECOS can further deliver congestion-safe results.

The remainder of this paper is organized as follows. Section II formulates the ECO synthesis problem, describes the traditional ECO synthesis flow and introduces terminology. Section III details ECOS. Section IV discusses the extension on congestion-driven ECO. Section V shows the experimental results. Finally, Section VI gives the conclusion.

II. Problem Formulation and Traditional ECO Synthesis Flow

This section gives the problem formulation, describes the traditional ECO synthesis flow and introduces terminology.

A. Problem Formulation

As shown in Fig. 2, the metal-only ECO synthesis problem discussed in this paper is formulated as follows.

The Minimum-Cost ECO Synthesis Problem: Given the netlist and placement of a design, the cell library, a set of spare cells and an ECO list (a list of functional changes), complete the ECO list using the available spare cells, create the revised netlist with the minimum cost and generate the revised set of spare cells.

Our goal is to complete functional changes without sacrificing timing and routability; hence, without loss of generality, the metal-only ECO cost of a design is defined as the total half-perimeter wirelength (HPWL) of all nets and thus it is beneficial for timing and routability [14].

Fig. 3(a) shows an example design with two inputs, two outputs, four logic cells and six nets. The available spare cells include two AND and one INV (inverter) cells. The placement of logic and spare cells is also illustrated. For simplicity and easier visualization, the area of each cell in this example is 0 and all pins are located at the same point. The total cost can be computed as follows:

$$\sum_{i=1..6} \mathrm{HPWL}(n_i) = 6000+1000+3000+5000+5000+2000 = 22\,000. \quad (1)$$

B. Traditional ECO Synthesis Flow

Metal-only ECO is commonly performed by hand-editing the netlist. However, this ad hoc method is very time-consuming and resource intensive because the design related files have to be searched and edited many times during the whole ECO process [16].

Fig. 3(b) shows the ECO list of the design given in Fig. 3(a), functional change (FC) F_1 describes $n_3 = \mathrm{AND}(n_1, n_2)$, replacing cell U_3 with an AND cell. As shown in Fig. 3(c), cell U_3 becomes unused and cannot affect nets n_1, n_2 and n_3 any more after F_1 is applied, so U_3's inputs and output can be disconnected from nets n_1, n_2 and n_3 before F_1 is applied.

Checking the placement, we would like to replace U_3 with a spare AND cell close to cells U_1, U_2, U_4 and output O_2. We have two options, spare cells S_1 and S_2 and it can be seen that S_2 has the better proximity. Hence, the revised design could be as Fig. 3(d). The total cost of the revised design is as follows:

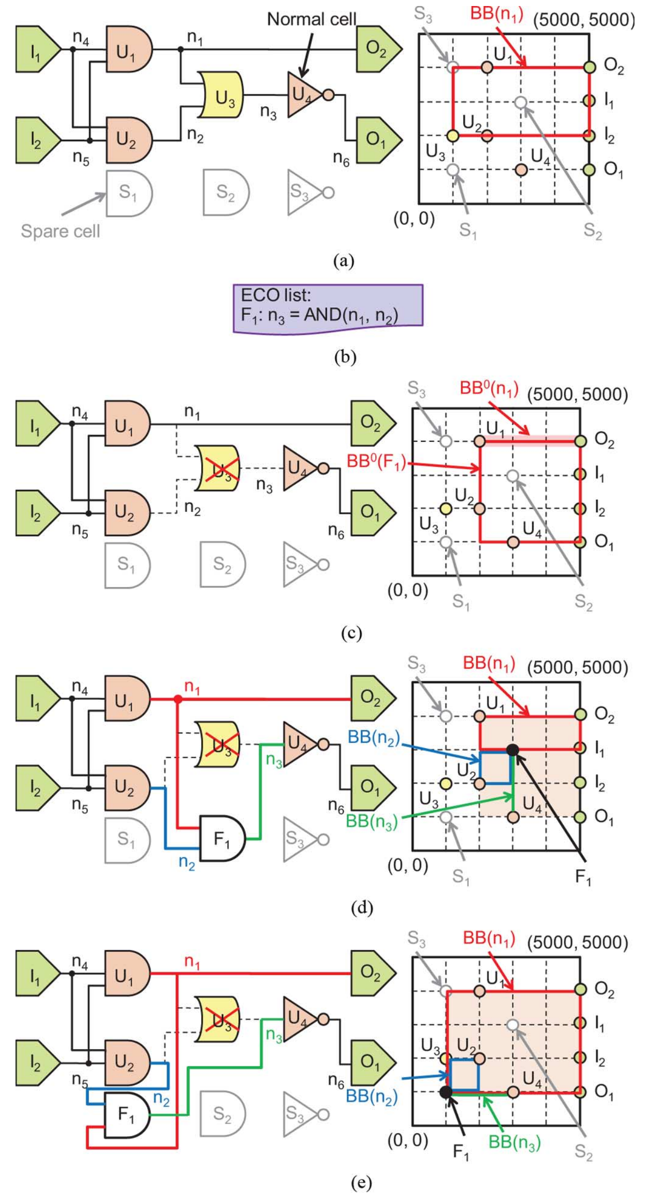
$$\sum_{i=1..6} \mathrm{HPWL}(n_i) = 4000+2000+2000+5000+5000+2000 = 20\,000. \quad (2)$$

The example demonstrated in Fig. 3 is relatively simple because spare S_2 directly matches F_1's functionality. However, when they are mismatched, we shall realize the ECO functionalities by available spare cell types. On the other hand, when the size of the ECO list is large, the hand-editing task would be time-consuming. Because the resource of spare cells is limited, when a specific spare cell is the best choice for several FCs, the ad hoc method, unfortunately, cannot handle the competition and even fails to complete the ECO list.

C. Terminology

In this subsection, we define the pre-ECO bounding box and HPWL of an FC or a net, as well as the lower bound of the ECO cost of assigning a spare cell to an FC. In Section III, we shall utilize them to facilitate our metal-only ECO synthesizer. Table I briefly summarizes the terms used throughout this paper.

First of all, unused connections can be removed before FCs are applied. For example, as shown in Fig. 3(c), nets n_1, n_2 and n_3 are not related to cell U_3 after F_1 is applied, so U_3 can be disconnected from these nets. The pre-ECO bounding box of an FC is the bounding box covering the FC's related nets after unused connections are removed and before the FC is applied.



Cell library of functional change F_1			
Spare cell name	Function	Area	Delay
S_1	AND	7,000	0
S_2	AND	6,000	0
S_3	INV	7,000	0

(f)

Fig. 3. (a) A design with two inputs (I_1 and I_2), two outputs (O_1 and O_2), four logic cells, (U_1, U_2, U_3 and U_4) and six nets ($n_1, n_2, \ldots, n_6$). The available spare cells include two AND and one INV cells. The highlighted rectangle is the bounding box of net n_1 and its HPWL is 6,000. The total cost is 22,000 (b) The ECO list describes cell U_3 should perform logic AND instead of OR (c) FC F_1's pre-ECO HPWL is 6,000, while n_1's pre-ECO HPWL is 3,000 (d) The revised netlist and placement after cell U_3 is replaced with spare S_2. The HPWL of net n_1 becomes 4,000 and the total cost is 20,000. Since cell U_3 becomes unused, it is freed up and can serve as a spare cell. (The input pins of cell U_3 are tied to logic high/low.) (d) The lower bound vs. the real ECO costs (HPWL) of assigning S_2 to F_1. The shaded area shows $\mathrm{BB}^1(F_1, S_2)$. If F_1 selects S_2, the lower bound of HPWL over nets n_1, n_2, n_3, $\mathrm{HPWL}^1(F_1, S_2)$, is 6,000, while the real cost $\mathrm{HPWL}(F_1)$ is $8,000 \; (= \mathrm{HPWL}(n_1) + \mathrm{HPWL}(n_2) + \mathrm{HPWL}(n_3))$ (e) If F_1 selects S_1, the lower bound of the ECO cost $\mathrm{HPWL}^1(F_1, S_1)$ is 7,000, while the real cost $\mathrm{HPWL}(F_1)$ is 11,000 (f) The corresponding spare cell library for F_1 used in Guided ABC, where the cell area is the lower bound ECO cost of assigning each spare cell to F_1, $\mathrm{HPWL}^1(F_1, S_k)$ for $k = 1, 2, 3$.

TABLE I
TERMINOLOGY

Term	Description
$N = \{n_i\}$	The set N of nets; a net n_i.
$F = \{F_j\}$	The set F of functional changes (FCs); an FC F_j.
$S = \{S_k\}$	The set S of spare cells; a spare cell S_k.
$\mathrm{BB}(n_i)$	The bounding box of net n_i.
$\mathrm{HPWL}(n_i)$	The HPWL of net n_i, HPWL based on $\mathrm{BB}(n_i)$.
$\sum_{ni \in N} \mathrm{HPWL}(n_i)$	The ECO cost, the total HPWL over all nets.
$\mathrm{HPWL}(F_j)$	The total HPWL of F_j's related nets.
$\mathrm{BB}^0(n_i)$	The pre-ECO bounding box of net n_i.
$\mathrm{HPWL}^0(n_i)$	The pre-ECO HPWL of net n_i, HPWL based on $\mathrm{BB}^0(n_i)$.
$\mathrm{BB}^0(F_j)$	The pre-ECO bounding box of FC F_j.
$\mathrm{HPWL}^0(F_j)$	The pre-ECO HPWL of FC F_j, HPWL based on $\mathrm{BB}^0(F_j)$.
$\mathrm{BB}^1(F_j, S_k)$	The bounding box of assigning spare S_k to FC F_j.
$\mathrm{HPWL}^1(F_j, S_k)$	The lower bound of HPWL of assigning spare S_k to FC F_j, HPWL based on $\mathrm{BB}^1(F_j, S_k)$.
$\mathrm{pref}(F_j, S_k)$	The preference value between F_i and S_k, $\Delta\mathrm{cost}(F_j, S_k) + \sum_{i:\, ni,j \in N} \mathrm{HPWL}(n_{i,j})$, $n_{i,j}$ connects F_i and F_j.
$\Delta\mathrm{cost}(F_j, S_k)$	The Manhattan distance between S_k and $\mathrm{BB}^0(F_j)$, $\mathrm{HPWL}^1(F_j, S_k)$ - $\mathrm{HPWL}^0(F_j)$.
$\mathrm{HPWL}(n_{i,j})$	HPWL of the internal net $n_{i,j}$ between F_i and F_j.
$(x^0(F_j), y^0(F_j))$	The initial coordinate of FC F_j's reference point.
$\mathrm{dist}(F_i, F_j)$	Manhattan distance between the F_i and F_j.
$\mathrm{dist}(F_i, S_k)$	Manhattan distance between the F_i and S_k.
b_j	Congestion bin.
$C(b_j)$	The congestion value of bin b_j.
$R(b_j)$	The routing supply of bin b_j.
$C(n_i, b_j)$	The congestion value of bin b_j contributed by net n_i.
$\mathrm{cong}(n_i)$	The consumed routing resources of net n_i.
$O(n_i, b_j)$	The overlap of net n_i and bin b_j, $O(n_i, b_j) = w(n_i, b_j)\cdot h(n_i, b_j)$.

Remark:
$C^0(b_j)$, $R^0(b_j)$, $C^0(n_i, b_j)$, $\mathrm{cong}^0(n_i)$, $O^0(n_i, b_j)$ are computed based on pre-ECO bounding boxes of all nets.

Similarly, the pre-ECO bounding box of a net is the bounding box of the net which unused connections are excluded.

Definition 1: Given an ECO list. Suppose an FC F_j is related to a set of nets, each of which contains multiple pins. These pins without F_j's pins define F_j's pre-ECO bounding box $\mathrm{BB}^0(F_j)$; F_j's pre-ECO HPWL $\mathrm{HPWL}^0(F_j)$ is the HPWL based on $\mathrm{BB}^0(F_j)$.

For example, as shown in Fig. 3(b) and (c), F_1 is related to nets n_1, n_2 and n_3 and thus $\mathrm{BB}^0(F_1)$ is formed by port O_2, cell U_1's output pin, cell U_2's output pin and cell U_4's input pin. (By definitioin 1, cell U_3 is excluded for $\mathrm{BB}^0(F_1)$ computation.) Hence, we have $\mathrm{HPWL}^0(F_1) = 6000$. In addition, $\mathrm{BB}^0(n_1)$ shrinks into a line segment ($\mathrm{HPWL}^0(n_1) = 3000$), while $\mathrm{BB}^0(n_2)$ and $\mathrm{BB}^0(n_3)$ become points ($\mathrm{HPWL}^0(n_2) = 0$, $\mathrm{HPWL}^0(n_3) = 0$).

The pre-ECO HPWL of an FC provides a loose lower bound of the total ECO cost induced by this FC. After assigning a spare cell to a specified FC, the lower bound can be computed more accurately.

Definition 2: Suppose an FC F_j is related to a set of nets, each of which contains multiple pins. Considering a spare cell S_k is assigned to F_j, $\mathrm{BB}^1(F_j, S_k)$ is the bounding box covering S_k and the pins on F_j's related nets excluding unused connections. The lower bound of the ECO cost $\mathrm{HPWL}^1(F_j, S_k)$ is the HPWL based on $\mathrm{BB}^1(F_j, S_k)$.

The lower bound of the ECO cost of assigning a spare cell to an FC reflects the potential wirelength contributed by this assignment. The real cost of this assignment reaches the lower bound when the bounding boxes of the FC's related nets are completely disjoint. Given an FC F_j and an available spare S_k, we have the following inequalities for three types of HPWL:

$$\mathrm{HPWL}^0(F_j) \le \mathrm{HPWL}^1(F_j, S_k) \le \mathrm{HPWL}(F_j). \quad (3)$$

As shown in Fig. 3(d), the lower bound ECO cost of assigning S_2 to F_1, $\mathrm{HPWL}^1(F_1, S_2)$, is 6000 $(= \mathrm{HPWL}^0(F_1))$, while the real cost $\mathrm{HPWL}(F_1)$ is 8000. Similarly, as shown in Fig. 3(e), we have $\mathrm{HPWL}^1(F_1, S_1) = 7000(\ge \mathrm{HPWL}^0(F_1))$ and $\mathrm{HPWL}(F_1) = 11\,000$. In addition, when either S_1 or S_2 is assigned to F_1, the corresponding bounding boxes of nets n_1, n_2 and n_3 overlap, so the real cost is higher than the lower bound.

If the cells to be revised become unobservable/unused after ECO, e.g., cell U_3 in Fig. 3(c), they can be freed up and behave as spare cells for current and/or subsequent ECO runs. Freeing up these unobservable/unused cells is done by tying their inputs to constant values (logic high or low) to avoid floating inputs and disconnecting their outputs to avoid multiple drives.

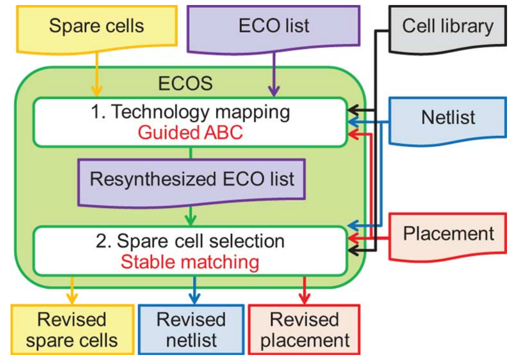


Fig. 4. Overview of ECOS.

III. STABLE MATCHING BASED METAL-ONLY ECO SYNTHESIS

In this section, we present a stable matching based metal-only synthesizer-ECOS. Fig. 4 shows ECOS contains two steps: technology mapping and spare cell selection. To optimize the ECO cost, different types of HPWL are modeled as the cost in the two steps.

The spare cells are limited in cell types and in number. The issue of cell types is solved at technology mapping: ECOS translates the given ECO list using available spare cell types with geometry proximity consideration.

The issue of number is handled at spare cell selection: since a specific spare cell may be duplicated multiple times in the resynthesized ECO list, ECOS resorts the competition among FCs for this spare cell to stable matching.

A. Technology Mapping: Guided ABC

An automatic method that can select spare cells of proper types and in good proximity is desirable. For the example given in Fig. 3, if S_2 is chosen, the total cost is 20 000; if S_1 is chosen instead, the total cost becomes worse, 23 000. Moreover, when the ECO functionalities mismatch spare cell types, the ECO list should be translated into available spare cell types.

The first step of ECOS performs technology mapping to resynthesize the given ECO list using the available spare cell types with physical information consideration. After this step, a resynthesized ECO list is produced. We build our synthesizer based on the well-established environment, ABC, developed by Berkeley logic synthesis and verification group [17]. Basically, ABC performs optimal-delay DAG-based technology mapping, i.e., it first optimizes delay and then recovers (reduces) area without hurting delay.

We guide ABC with spare cell types and proximity. In order to consider the geometry proximity of spare cells into resynthesis, the cell library is customized for each FC F_j. Each spare cell S_k is viewed as one unique library cell; its cell area is set to the lower bound of the ECO cost of assigning S_k to $F_j(\mathrm{HPWL}^1(F_j, S_k))$, while its cell delay is set to zero. Because the delay of each possible mapping is the same, i.e., 0, ABC is forced to perform area recovery (area reduction); area recovery minimizes HPWL^1 for each FC leading to a resynthesized list with good proximity. For example, for F_1 in Fig. 3, we have

$$\begin{aligned}\mathrm{area}(S_1) &= \mathrm{HPWL}^1(F_1, S_1) = 7000\\ \mathrm{area}(S_2) &= \mathrm{HPWL}^1(F_1, S_2) = 6000\\ \mathrm{area}(S_3) &= \mathrm{HPWL}^1(F_1, S_3) = 7000. \end{aligned} \quad (4)$$

The cell library for F_1 is created as Fig. 3(f).

Based on the customized spare cell library of each FC and optimal-delay DAG-based technology mapping, guided ABC can generate the best choice for each FC. For example, for F_1, spares S_1 and S_2 have the same delay (zero), so during area recovery, the cell of smaller area is selected, i.e., guided ABC maps F_1 as a spare S_2 cell. Hence, guided ABC can naturally choose the spare cell of a proper cell type and in good proximity.

The example given in Fig. 3 is simple because the ECO list has only one FC and some spare cells match the ECO functionality. However, the ECO list usually has multiple FCs and functionalities mismatch with available spare cells in most cases. Guided ABC translates one FC in the ECO list into available spare cell types at a time and an FC may be converted into several cells. Eventually, the resynthesized list is usually longer than the original one. Later, our results show that guided ABC generates the resynthesized list of a smaller size than a synthesizer without physical information consideration.

Moreover, DAG-based technology mapping cannot directly handle non-tree type spare cells and ECO functions. We resort this problem to ROBDDs [18]. If the spare cell types are a mixture of only multiplexors (MUX) and inverters (INV), the ECO list will be transformed to ROBDDs first and these ROBDDs are then simplified and converted to MUX/INV cells by ABC. Fig. 5 shows F_1 given in Fig. 3(b) is alternatively mapped as a multiplexor if only MUXs/INVs are available.

In addition, constant insertion can maximize the capability of each spare cell, e.g., a two-input NAND cell can be an INV cell by inserting a logic high to one input. Constant insertion is naturally integrated into technology mapping by including constant inserted counterparts of each spare cell into the cell library. It can be seen that the guidance made for ABC indeed can easily be built in other existing logic synthesizers.

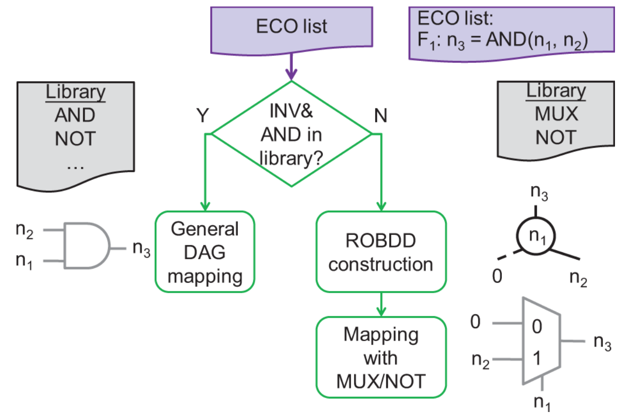


Fig. 5. Guided ABC for non-tree type spare cells. If spare cells contain only multiplexors and inverters, the ECO list is first transformed to ROBDDs and then mapped.

B. Spare Cell Selection: Stable Matching

Although guided ABC has already considered the physical information of spare cells, it cannot handle the competition among FCs well. The competition among FCs occurs when a spare cell is the best choice of several FCs, guided ABC duplicates it for these FCs. To solve this problem, we do not select spare cells directly in guided ABC, but defer the decision making to step 2. The second step of ECOS selects one spare cell for each FC in the resynthesized ECO list. With the global view of physical information, deferred decision making may lead to good results. We reduce spare cell selection to the stable marriage problem, which is suitable to solve the competition among many candidates in nature [15].

1) The Stable Marriage Problem: The stable marriage problem is formulated as follows.

The Stable Marriage Problem: Given a set of n men and m women, each man has ranked the women in order of preference and each woman has done likewise, marry them off in pairs such that no pair of man and woman would both prefer each other to their current partners. If there are no such pairs, all the marriages are stable.

Gale and Shapley showed that a stable marriage exists for any ranking when the preference lists are complete and have no ties. Gale–Shapley algorithm is listed in lines 1–9 in Fig. 6 [15]. This pairing method is male-optimal, i.e., every man is paired with his highest ranked feasible partner.

2) The Reduction: Since Gale–Shapley pairing is male-optimal, each FC in the resynthesized list is modeled as a man, while each spare cell is modeled as a woman. The preference reflects the added cost resulting from assigning a spare cell to an FC. The less added cost, the more preference. The added cost is the difference between the real HPWL of each FC and its HPWL^0. (Recall that at technology mapping, guided ABC considers HPWL^1 as the cost (cell area) for each FC.) Since HPWL^0 cannot be changed, we exclude HPWL^0 from preference calculation to avoid that FCs with small/large HPWL^0 bias the preference.

FCs in the resynthesized list may be dependent. For example, if one FC in the original ECO list is translated into more than one cell in the resynthesized ECO list, there exist some newly

```
StableMatching(M, W)
// M: the set F of FCs; W: the set S of spare cells
1. Initialize all m ∈ M and w ∈ W as free
2. while ∃ free man m who hasn't proposed to all women do
3.    w = the highest ranked women in m's preference list
4.    if w is free then
5.       (m, w) become engaged
6.    else // some pair (m', w) is currently engaged
7.       if w prefers m to m' then
8.          (m, w) become engaged
9.          m' becomes free
10.   Update preference
```

Fig. 6. Modified stable matching algorithm. FCs are men, while spare cells are women. During execution, engagement means a temporal assignment and at the end, all engaged pairs become married. To handle the interference among FCs, we update preference (see line 10) at the end of each iteration in Gale–Shapley algorithm [15].

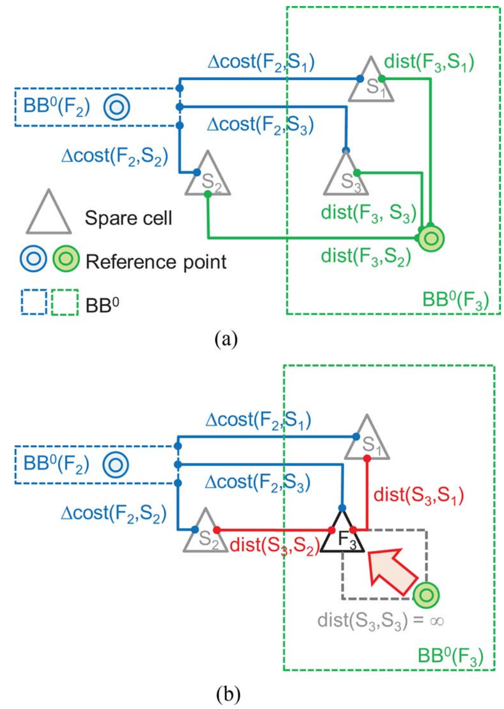


Fig. 7. Preference update. (a) The reference point of $F_2(F_3)$ is at the average x- and y-coordinates of the pins on its related nets (b) After the proposal of F_3 to spare cell S_3 is accepted, F_3's coordinate is set to S_3's and the related costs are updated.

created nets connecting the remapped cells. The added cost then contains two parts: The first part is the impact on the pre-ECO HPWLs of the existing nets, while the second part is the induced cost on the internal nets among FCs. Hence, the preference value $\text{pref}(F_j, S_k)$ between FC F_j and spare S_k is defined as follows:

$$\text{pref}(F_j, S_k) = \Delta\text{cost}(F_j, S_k) + \sum_{i:n_{i,j} \in N} \text{HPWL}(n_{i,j}) \quad (5)$$

where $\Delta\text{cost}(F_j, S_k) = \text{HPWL}^1(F_j, S_k) - \text{HPWL}^0(F_j)$, and $n_{i,j}$ is the internal net[2] connecting FCs F_i and F_j.

If each FC in the resynthesized list is independent, i.e., no internal connection among FCs, the preference order can be determined directly. For example, F_1 given in Fig. 3 has the respective preference values:

$$\begin{aligned} \text{pref}(F_1, S_1) &= \Delta\text{cost}(F_1, S_1) + 0 = 1000 \\ \text{pref}(F_1, S_2) &= \Delta\text{cost}(F_1, S_2) + 0 = 0 \\ \text{pref}(F_1, S_3) &= \Delta\text{cost}(F_1, S_3) + 0 = \infty. \end{aligned} \quad (6)$$

Thus, F_1 prefers S_2 and proposes to S_2. S_2 then accepts and the stable matching is found.

When FCs in the resynthesized list are dependent, the spare cell selection would affect each other. To break the interference between FCs, we try to estimate the induced cost on the internal nets among FCs. A reference point is introduced to each FC, representing the desirable location for its assigned spare cell. Each FC is initially located at its reference point, which is set to the center (the average x- and y-coordinates) over all pins on its related nets without considering unused connections. For example, the reference point of F_1 in Fig. 3 is the average x- and y-coordinates over port O_2, cell U_1's output pin, cell U_2's output pin and cell U_4's input pin.

$$\begin{aligned} x^0(F_1) &= \frac{(2000 + 2000 + 3000 + 5000)}{4} = 3000 \\ y^0(F_1) &= \frac{(4000 + 2000 + 1000 + 4000)}{4} = 2750. \end{aligned} \quad (7)$$

With setting the reference points, we can compute the induced cost on internal nets and then rank the preference between FCs and spare cells. For example, Fig. 7(a) depicts the reference points and the pre-ECO bounding boxes of F_2 and F_3. Assume there exists an internal net between F_2 and F_3. According to (5), the preference of F_2 proposing to S_1 is as follows:

$$\begin{aligned} \text{pref}(F_2, S_1) &= \Delta\text{cost}(F_2, S_1) + \text{HPWL}(n_{2,3}) \\ &= \Delta\text{cost}(F_2, S_1) + \text{dist}(F_3, S_1). \end{aligned} \quad (8)$$

$\Delta\text{cost}(F_2, S_1) = \text{HPWL}^1(F_2, S_1) - \text{HPWL}^0(F_2)$, meaning the distance between S_1 and F_2's pre-ECO bounding box. Considering F_2 proposes to S_1, the induced cost of the net between F_2 and F_3, $\text{HPWL}(n_{2,3})$, is the Manhattan distance between F_3's reference point and S_1, denoted as $\text{dist}(F_3, S_1)$. Hence, we have

$$\begin{aligned} \text{pref}(F_2, S_1) &= \Delta\text{cost}(F_2, S_1) + \text{dist}(F_3, S_1) \\ \text{pref}(F_2, S_2) &= \Delta\text{cost}(F_2, S_2) + \text{dist}(F_3, S_2) \\ \text{pref}(F_2, S_3) &= \Delta\text{cost}(F_2, S_3) + \text{dist}(F_3, S_3). \end{aligned} \quad (9)$$

As shown in Fig. 7(b), since there is an internal net between F_2 and F_3, after F_3 is engaged to S_3, $\text{HPWL}(n_{2,3})$ is updated and F_2's preference list is updated accordingly:

$$\begin{aligned} \text{pref}(F_2, S_1) &= \Delta\text{cost}(F_2, S_1) + \text{dist}(S_3, S_1) \\ \text{pref}(F_2, S_2) &= \Delta\text{cost}(F_2, S_2) + \text{dist}(S_3, S_2) \\ \text{pref}(F_2, S_3) &= \Delta\text{cost}(F_2, S_3) + \text{dist}(S_3, S_3) = \infty. \end{aligned} \quad (10)$$

Please note that the distance between F_3 and S_3, $\text{dist}(S_3, S_3)$, is set to a large value rather than 0 to prevent F_2 from proposing

[2] An internal net may have multiple pins, i.e., connecting more than two FCs. In this case, its HPWL is computed based on its bounding box.

to S_3. Doing so can guarantee that the method is stable and can always generate a solution.

Spare cell selection follows Gale-Shapely algorithm and additionally updates preference lists at the end of each iteration (see line 10 in Fig. 6).

As the execution of stable matching progresses, once a spare cell S_k is temporarily assigned to an FC F_j (see lines 5 and 8 in Fig. 6), F_j's coordinate is updated to S_k's location. This assignment affects F_j's related internal nets and the influenced preference values should be updated accordingly. An engaged FC's preference list keeps the same to maintain stability until it turns to be free. When it turns to be free, its location is changed back to its reference point (x^0, y^0) and its related preference values are updated.

Fig. 8 details the execution of spare cell selection by stable matching (see Fig. 6). Given a resynthesized ECO list with four FCs-F_1(INV), F_2(INV), F_3(AO22) and F_4(INV), where F_3 is connected to F_1 and F_2 individually. Suppose there are four spare cells, S_1(INV), S_2(INV), S_3(INV) and S_4(AO22). Fig. 8(a) depicts the status at the very beginning: Every FC is free (see line 1 in Fig. 6) and each FC is located at its reference point. The initial preference lists are also given, where preference values are sorted in ascending order and '-' represents an impossible match due to cell type conflict. The preference values are specified inside the parentheses, e.g.,

$$\text{pref}(F_1, S_2) = \Delta\text{cost}(F_1, S_2) + \text{dist}(F_3, S_2) = 6 + 13. \quad (11)$$

In this case, F_2 and F_4 have competition for S_3. Assume F_1, F_2, F_3 and F_4 sequentially propose in this execution. As shown in Fig. 8(b), F_1 proposes to its highest ranked spare cell S_2 and S_2 accepts F_1's proposal (see lines 3–5). The engaged pair is indicated by the shaded items. Since F_1 and F_3 are connected by net $n_{1,3}$, F_3's preference values are updated accordingly (highlighted by bold numbers inside parentheses). Then, as shown in Fig. 8(c), F_2 and F_3 sequentially propose to S_3 and S_4 and the influenced preference values are updated. Fig. 8(d) shows once F_4 proposes to S_3, S_3 prefers F_4 to F_2, so S_3 is engaged to F_4 and F_2 becomes free (see lines 6–9 in Fig. 6). Then, F_2's preference list is updated to facilitate its next proposal. Finally, as shown in Fig. 8(e), F_2 proposes to S_1 and S_1 accepts. After stable matching, each FC is matched to a spare cell and the competition among FCs are resolved.

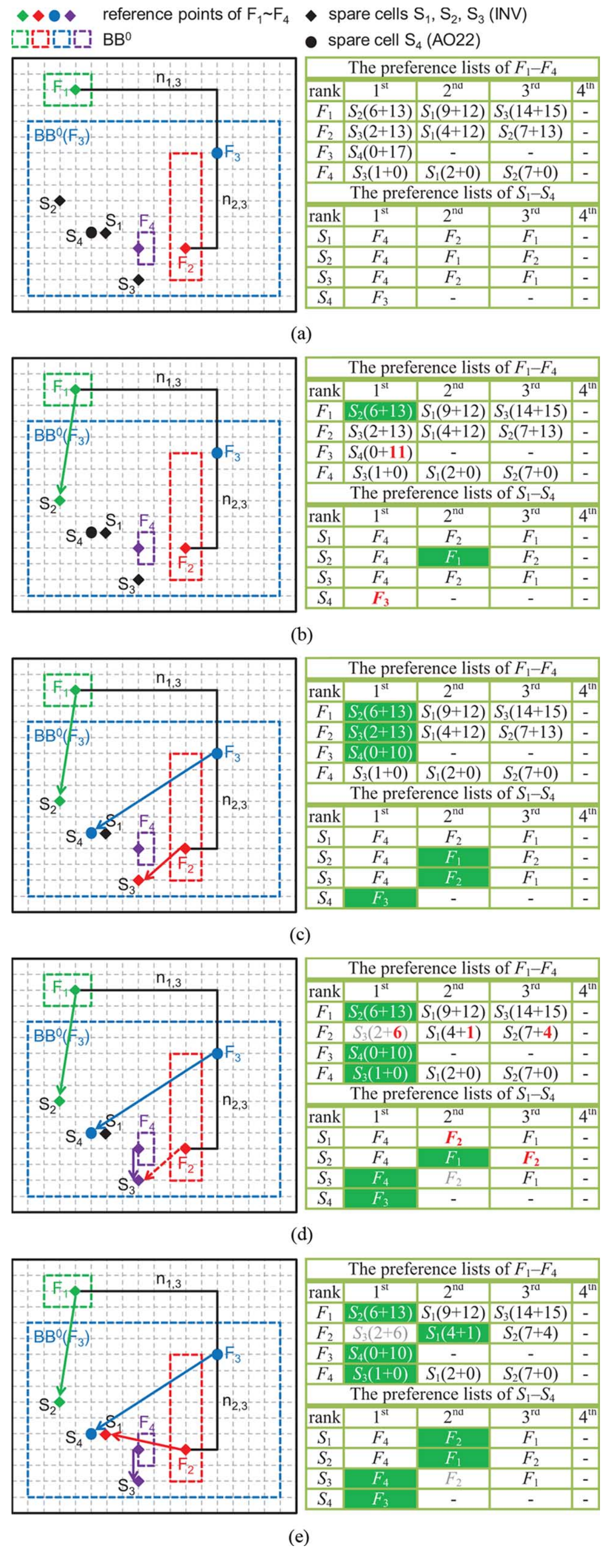


Fig. 8. Stable matching with preference update (a) Given an instance with 4 FCs, 4 spare cells and 2 internal nets. Initially, every FC and spare cell is free. The initial preference lists are also given, where the preference values are specified inside parentheses (b) F_1 proposes to its highest ranked spare S_2 and gets accepted. The influenced preference values are updated accordingly (c) F_2 and F_3 sequentially propose to S_3 and S_4 (d) F_4 propose to S_3; S_3 accepts F_4 and dumps F_2. F_2 turns back to be free and its preference list is updated (e) F_2 proposes to S_1 and S_1 accepts. This is a stable matching.

C. The Complexity of Spare Cell Selection

Preference values, constant insertion options, freed up cells can be obtained during step 1. Step 2 could focus on ranking and matching. Based on the resynthesized ECO list, ranking n FCs and m spare cells can be done in $O(nm \log m + mn \log n)$ time. The free FCs are maintained by a queue. The worst case of the stable matching algorithm is $O(nm \log n)$, i.e., each possible proposal is examined and the preference ranking is updated. Hence, the overall time complexity of step 2 is $O(nm(\log m + \log n)) = O(nm \log m)$ since n is less than or equal to m. Only the spare cells of functionally compatible types can be matched for an FC, the expected number of possible proposals is far less than nm. Moreover, the preference list of an engaged FC should be updated only when it turns back to be free. Hence, the practical complexity is quite lower than the worst case.

IV. Extension to Congestion-Driven Metal-Only ECO

To facilitate the ECO router, the congestion information should be combined into the ECO cost as well. The congestion information is usually stored in a congestion map and it could be estimated based on the global routing topology or the bounding box of each net [19]–[21]. The global routing topology (Steiner trees or L-/Z-shaped routes) may give a good estimation when it coincides with the actual route [19]. On the other hand, the bounding box method is efficient since no routing structure is required. Lu *et al.* use the via density with respect to the bounding box to estimate the congestion since an incremental ECO router is used [8]. The findings in [19], [20] reveal that the congestion-weighted HPWL has a strong correlation to the routed wirelength.

Hence, for congestion-driven ECO, guided ABC remains the same, but we do the following treatments to stable matching.

1) The congestion map is built based on the congestion model used in [20].
2) The congestion map guides the preference values and ECOS incrementally updates the congestion map.

A. Congestion Estimation

We adopt the congestion estimation method proposed by [20]. First of all, as shown in Fig. 9, the placement region is evenly divided into non-overlapping bins of height H_b and of width W_b. Each congestion bin b_j is associated with a congestion value $C(b_j)$ and a routing supply $R(b_j)$; $C(b_j)$ means the amount of congestion caused by all nets within bin b_j, while $R(b_j)$ means the total routing resources supplied by bin b_j. Basically, a congestion-safe bin b_j means its congestion value does not exceed its routing supply:

$$C(b_j) \leq R(b_j). \tag{12}$$

Fig. 9 shows net n_i's bounding box $\mathrm{BB}(n_i)$ is $H(n_i)$ high and $W(n_i)$ wide. Let C_h (respectively C_v) be the maximum number of nets in all metal layers that can pass a congestion bin horizontally (respectively vertically). The total consumed routing resources $\mathrm{cong}(n_i)$ of net n_i with an L-/Z-shaped route can be computed as follows:

$$\mathrm{cong}(n_i) = \frac{H_b}{C_h} \cdot W(n_i) + \frac{W_b}{C_v} \cdot H(n_i). \tag{13}$$

Based on [19], [20], $\mathrm{cong}(n_i)$ is uniformly distributed over net n_i's bounding box. Hence, for the congestion bin b_j located at the top right of net n_i's bounding box in Fig. 9, we have the congestion value $C(n_i, b_j)$ contributed by net n_i on bin b_j as follows:

$$C(n_i, b_j) = \mathrm{cong}(n_i) \cdot \frac{h(n_i, b_j) w(n_i, b_j)}{H(n_i) W(n_i)} \tag{14}$$

where $h(n_i, b_j)$ and $w(n_i, b_j)$ are the respective vertical and horizontal overlap between net n_i and bin b_j. The congestion value $C(b_j)$ of bin b_j is the total congestion values contributed by all nets on b_j and we have

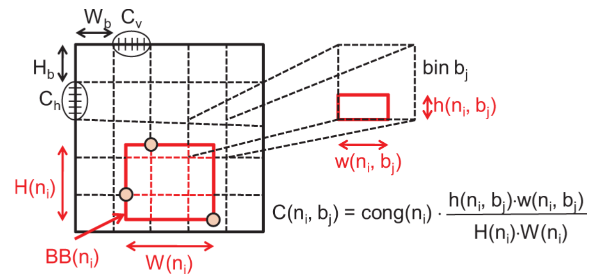


Fig. 9. Congestion estimation based on [20]. The congestion value of bin b_j contributed by net n_i, $C(n_i, b_j)$, is determined by the consumed routing resource of net n_i and the overlap of net n_i's bounding box and bin b_j.

$$C(b_j) = \sum_{i:BB(n_i)\cap b_j \neq \emptyset} C(n_i, b_j). \tag{15}$$

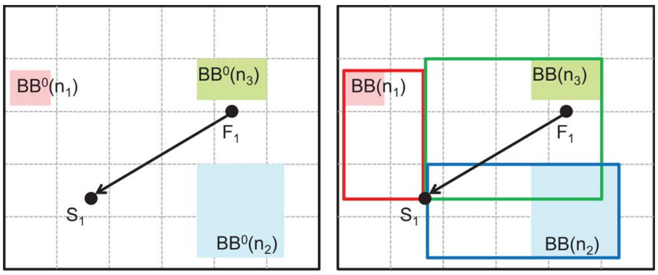
On the other hand, considering block porosity, macros may occupy or block some routing resources and introduce a base congestion value to each related congestion bin [20]. Hence, the routing supply of bin b_j is

$$R(b_j) = U_b(W_b H_b - B(b_j)) \tag{16}$$

where U_b is the target utilization and $B(b_j)$ is the routing resources consumed by macros in bin b_j.

For example, suppose Fig. 3(d) shows a partial placement without macros. Assume $H_b = W_b = 4{,}000$, $C_h = C_v = 10$, $U_b = 0.5$. Let bin b_1 cover the area from (0,0) to (4000, 4000). Based on (13)–(16), we have

$$\begin{aligned}
\mathrm{cong}(n_1) &= 400 \times 3000 + 400 \times 1000 = 1\,600\,000\\
\mathrm{cong}(n_2) &= 400 \times 1000 + 400 \times 1000 = 800\,000\\
\mathrm{cong}(n_3) &= 0 + 400 \times 2\,000 = 800\,000\\
\mathrm{cong}(n_4) &= 400 \times 3000 + 400 \times 2000 = 2\,000\,000\\
\mathrm{cong}(n_5) &= 400 \times 3000 + 400 \times 2000 = 2\,000\,000\\
\mathrm{cong}(n_6) &= 400 \times 2000 + 0 = 800\,000\\
C(n_1, b_1) &= 1\,600\,000 \times \left(\frac{2}{3}\right) = 1\,066\,667\\
C(n_2, b_1) &= 800\,000\\
C(n_3, b_1) &= 800\,000\\
C(n_4, b_1) &= 2\,000\,000 \times \left(\frac{2}{3}\right) = 1\,333\,333\\
C(n_5, b_1) &= 2\,000\,000 \times \left(\frac{2}{3}\right) = 1\,333\,333\\
C(n_6, b_1) &= 800\,000 \times \left(\frac{1}{2}\right) = 400\,000\\
C(b_1) &= 5\,733\,333\\
R(b_1) &= 0.5 \times 4000 \times 4000 = 8\,000\,000.
\end{aligned} \tag{17}$$



(a) (b)

Fig. 10. The preference value for congestion-driven ECO (a) Given an FC F_1 with three related nets n_1, n_2 and n_3. Their pre-ECO bounding boxes are shown as shaded boxes. Consider F_1 proposes to spare cell S_1 (b) If F_1 proposes to S_1, the bounding boxes and the consumed routing resources of n_1, n_2 and n_3 will be enlarged. The congestion values on these bins are changed. The updated congestion values are used to compute the preference between F_1 and S_1.

TABLE II
TESTCASE STATISTICS

Statistics					
Case	#Pins	#Cells	#Nets	#Spare	#FC
testcase1	483	28,591	28,705	350	7
testcase2	483	28,591	28,705	2,300	49
testcase3	483	28,591	28,705	2,300	94
testcase4	33	198	181	40	3
testcase5	30	938	850	100	4
testcase6	490	13,674	15,071	280	13
testcase7	3,531	484,537	225,275	5,310	442
testcase8	437	579,452	296,257	4,771	538
testcase9	274	295,958	142,588	2,970	281

B. Congestion-Driven Stable Matching

Since ECO affects partial nets, based on the congestion estimation described above, the unchanged portions of nets as well as macros give an initial congestion map. Congestion-driven ECO targets to complete FCs with minimal wirelength and without violating the routing supply of any bin.

We use the congestion map instead of congestion-weighted HPWL to guide stable matching. The reason is that the minimal congestion-weighted HPWL cannot guarantee congestion safeness. Considering the pre-ECO bounding boxes of all nets, i.e., (13)–(15) are computed based on pre-ECO bounding boxes, we superposition their congestion values onto each related congestion bin and obtain an initial congestion map.

During spare cell selection (stable matching), when an FC in the resynthesized ECO list is engaged by a spare cell, the congestion values contributed by its related nets will be updated. As shown in Fig. 10(a), assume FC F_1 is related to nets n_1, n_2 and n_3. (The pre-ECO bounding boxes of these nets are indicated by the shaded boxes.) As shown in Fig. 10(b), considering F_1 is implemented by spare cell S_1, the bounding boxes of F_1's related nets may be enlarged and the congestion values of the related bins are changed accordingly. Our objective is to select a spare cell S_k for each FC F_j so that the related bins are congestion-safe and the resulting wirelength $\mathrm{HPWL}(F_j)$ is small.

Hence, for congestion-driven ECO, the preference between F_j and S_k contains two parts: the accumulated congestion values over the related bins and the congestion penalties of the related bins when S_k accepts F_j:

$$\mathrm{pref}(F_j, S_k) = \sum_{i:n_i \in F_j} \sum_{l:BB(n_i)\cap b_l \neq \emptyset} \left(C(b_l) + \lambda \left([C(b_l) - R(b_l)]^+ \right)^2 \right). \quad (18)$$

(The coefficient λ is user-specified.) The second term represents the quadratic congestion overflow penalty. $[x]^+ = \max(0, x)$, i.e., $[x]^+$ equals x when x is nonnegative; otherwise, $[x]^+$

TABLE III
ECO SYNTHESIS FLOW COMPARISON

	Features		ECO synthesis flow	
Method	FR	CI	Step 1: Technology mapping	Step 2: Spare cell selection
Human	-	-	Manual	Manual
Ad hoc	Y	Y	Blind ABC	Greedy selection
ECOS	Y	Y	Guided ABC	Stable matching
Modified [10]	Y	Y	Window-based guided ABC	Greedy selection

Y/N: Yes/No.
FR: Freed-up cell recycling. CI: Constant value insertion.
ABC: Synthesis & verification environment [17].
Modified [10]: Step 1: Guided ABC using spare cells within a search window. If guided ABC fails, the window progressively enlarges, and guided ABC repeats. Step 2: Greedy selection.

TABLE IV
GUIDED ABC VERSUS BLIND ABC

Case	#FR	#FC in the ECO list	#FC in the resynthesized ECO list	
			Blind ABC	Guided ABC
testcase1	7	7	15	10
testcase2	51	49	316	142
testcase3	121	94	503	203
testcase4	3	3	3	3
testcase5	4	4	4	4
testcase6	11	13	31	28
testcase7	470	442	1,767	1,571
testcase8	567	538	2,281	1,989
testcase9	292	281	1,184	1,072

#FR: The number of freed up cells.
#FC: The number of functional changes in the original and resynthesized ECO list.

TABLE V
CPU TIME COMPARISON

CPU Time (sec)							
Method	Ad hoc			ECOS			Modified [10]
Step	Blind ABC A	Greedy selection B	Total A+B	Guided ABC C	Stable matching D	Total C+D	Total E
testcase1	0.05	<0.01	0.06	0.52	<0.01	0.53	0.10
testcase2	0.27	0.07	0.34	7.69	0.06	7.75	12.73
testcase3	0.35	0.10	0.45	0.51	0.07	0.58	27.11
testcase4	0.16	<0.01	0.16	0.18	<0.01	0.18	0.03
testcase5	0.18	<0.01	0.18	0.20	<0.01	0.20	0.04
testcase6	<0.01	0.06	0.06	0.19	<0.01	0.19	1.37
testcase7	2.16	1.10	3.26	85.31	8.24	93.55	132.64
testcase8	2.18	1.22	3.41	87.33	7.19	94.51	157.94
testcase9	0.83	0.40	1.24	33.19	1.90	35.08	54.99
Ratio	-	-	0.37	-	-	1.00	7.27

TABLE VI
TOTAL COST (HPWL) COMPARISON

	Pre-ECO	Ad hoc		ECOS		ECOS with in-place reuse		Modified [10]	
Method	Total cost X	Total cost F	delta F-X	Total cost G	delta G-X	Total cost H	delta H-X	Total cost I	delta I-X
testcase1	4,049,536,290	4,052,083,390	2,547,100	4,051,021,350	1,485,060	4,050,178,050	641,760	4,051,069,890	1,533,600
testcase2	4,142,631,960	4,428,774,180	286,142,220	4,331,136,060	188,504,100	4,326,405,080	183,773,120	4,345,357,300	202,725,340
testcase3	4,142,631,960	4,638,942,680	496,310,720	4,555,641,140	413,009,180	4,541,501,480	398,869,520	4,592,655,120	450,023,160
testcase4	2,310,270	2,319,790	9,520	2,319,790	9,520	2,319,790	9,520	2,319,790	9,520
testcase5	12,945,900	13,012,260	66,360	13,012,260	66,360	13,012,260	66,360	13,012,260	66,360
testcase6	676,725,610	686,642,090	9,916,480	680,745,570	4,019,960	680,745,570	4,019,960	684,257,330	7,531,720
testcase7	15,476,755,310	17,065,002,670	1,588,247,360	16,896,832,010	1,420,076,700	16,587,814,490	1,111,059,180	17,531,904,350	2,055,149,040
testcase8	13,046,033,700	15,428,177,240	2,382,143,540	15,235,816,960	2,189,783,260	15,235,816,960	2,189,783,260	15,621,507,440	2,575,473,740
testcase9	7,244,415,440	8,046,589,680	802,174,240	7,869,825,400	625,409,960	7,869,825,400	625,409,960	8,054,111,600	809,696,160
Sum	-	-	5,567,557,540	-	4,842,364,100	-	4,513,632,640	-	6,102,208,640
Ratio	-	-	1.15		1.00	-	0.93	-	1.26

ECOS with in-place reuse: ECOS with in-place reusing freed-up cells.
delta: The cost difference between the pre-ECO total HPWL and the resulting HPWL, Y-X, Y = F, H, I.

equals 0. When the congestion value of some bin exceeds its routing supply, the quadratic congestion overflow is included into the preference value. Moreover, the first term reflects the sum of the congestion values of bins over F_j's related nets. Based on (13)–(15), the first term is highly correlated to the sum of $\mathrm{cong}(n_i)$ over F_j's related nets thus depending on their HPWLs. Defining preference as (18) not only maintains congestion safeness but also implicitly minimizes HPWL.

V. EXPERIMENTAL RESULTS

A. Experimental Settings

We implemented our algorithm in C++ language and executed the program on a PC with an Intel Core2 CPU T9400 of 2.53 GHz frequency and 4 GB memory under Windows 7 OS.

Totally nine industrial testcases are used. The spare cells for combinational logic for testcases 1–3 contain basic and complex logic cells (such as full adders); testcase4 and testcase5 use only multiplexors and inverters; testcases 6–9 use only basic logic cells, e.g., inverters, buffers, 2-input NANDs and 2-input NORs. The statistics is listed in Table II, including the number of pins (#Pins), the number of cells (#Cells), the number of nets (#Nets), the number of spare cells (#Spare) and the number of FCs (#FC). For testcases 2 and 3, spare cells are mainly located at corners; for testcases 7–9, although spare cells are evenly distributed, the number of spare cells is much fewer than other cases (around 1%). These testcases reflect the real difficulties faced by designers. In our experiments, the netlist and placement of the original design are described in DEF format [22], while the ECO list is specified in VERILOG format (using cell types specified in the cell library).

B. Comparison Between ECO Synthesis Flows

Table III outlines several ECO synthesis flows: human, ad hoc, ECOS and modified [10]. As described in Section II, the human method is manual (hand-editing), quite commonly adopted in design houses. We developed the ad hoc method to automate the traditional ECO synthesis flow for comparison. The ad hoc method adopts blind ABC to automate the human method. First of all, blind ABC translates each FC in the ECO list using all available spare cell types. At this step, blind ABC sets the area and delay of each cell to 0. Then, the spare cells are greedily selected based on the lower bound ECO cost, HPWL[1], defined in Section II-C. Moreover, we implemented modified [10] by incorporating the concept of prior work [10] into our framework. At step 1, we extract spare cells within the current search window to resynthesize the ECO list. If guided ABC fails to resynthesize the ECO list or the required number of some spare cell type is not affordable, the search window is enlarged and resynthesis is repeated. At step 2, spare cells are greedily selected according to HPWL.

C. Wirelength-Driven Functional ECO Results

Table IV lists the number of freed-up cells (#FR) and the sizes of the given and resynthesized ECO lists (#FC). #FR could be greater than #FC of the given ECO list when the freed up cell has multiple outputs (see testcases 2 and 3). Since spare cells are limited in cell types, the size of the resynthesized ECO list is greater than the given one. It can be seen that guided ABC always generates much fewer FCs than blind ABC. The smaller resynthesized ECO list may lead to the smaller ECO cost due to fewer internal nets, i.e., the bounding boxes of the related nets of an FC would have a smaller overlap.

Table V lists CPU times. Compared with the time-consuming human method, the automatic methods can complete ECO efficiently. 'Ratio' represents ECOS' speedup with respect to ad hoc and modified [10]. For difficult ECO cases, e.g., testcases 2, 3, 7–9, guided ABC consumes reasonable time to generate a short resynthesized ECO list. Moreover, the induced CPU times of stable matching are reasonably small; this phenomenon shows the scalability of our algorithm. For these cases, modified [10] may repeat guided ABC many times thus incurring long CPU times.

Table VI compares the total cost before and after ECO. The pre-ECO HPWL (column X) is the lower bound of total HPWL. 'ECOS with in-place reuse' means ECOS with reusing freed-up cells in the current ECO run. 'delta' represents the difference on HPWL compared with the lower bound HPWL. On average, ECOS outperforms ad hoc and modified [10] by 15% and 26%, respectively. In-place reuse brings 7% more reduction on HPWL, especially effective when the spare cells are gathered at one corner, like testcase2 and testcase3.

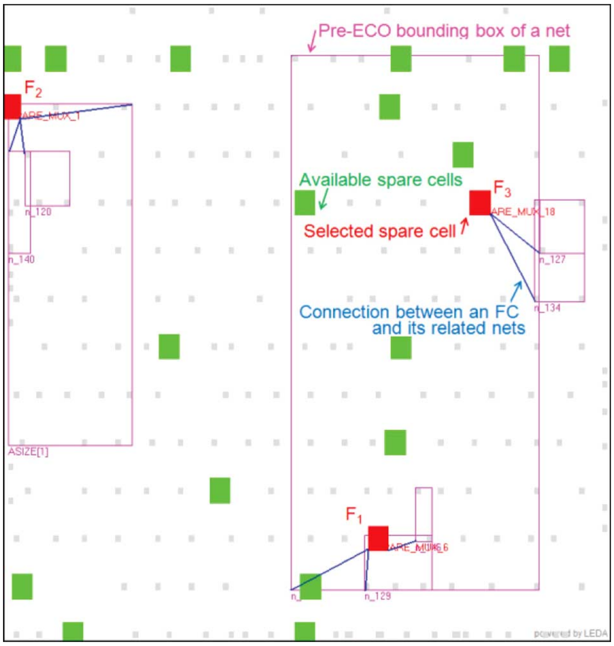


Fig. 11. Testcase4: Ad hoc, ECOS and modified [10] are all optimal.

In general, for early ECO runs, the ad hoc method delivers quite good solutions. For example, testcase4 has three FCs; for each FC, Fig. 11 highlights the pre-ECO bounding boxes of its related nets, available spare cells, the selected spare cell and the connection between the selected spare cell and its related nets. The ad hoc method, ECOS and modified [10] all generate the optimal solution. It can be seen that there is no competition and the nearest spare cell is assigned to each FC.

However, practical projects in design houses usually repeat ECO tasks many times. For late ECO runs, the spare cells are very limited, thus the available spare cells would be far away from each FC's reference point and the competition would be severe. In this situation, ECOS can help designers to complete ECO efficiently and effectively. For example, testcase3 shows an extreme case. Instead of uniform distribution, the spare cells are gathered to one corner. Unfortunately, the related nets of most FCs spread over large area. Hence, the resulting cost is much higher than the original value. Fig. 12 depicts the status of one FC of testcase3. This FC is converted to three FCs in the resynthesized ECO list. Because no available spare cells are located within its pre-ECO bounding box, without in-place reuse, its two internal nets incur long HPWL.

On the other hand, for testcases 7–9, although spare cells are evenly distributed, the number of spare cells is few and thus the competition would be severe. Moreover, because the spare cell types are basic, an FC of complicated functionality will be translated into multiple basic cells. Hence, the size of the resynthesized ECO list is large, which is 4 times the size of the original. Fig. 13 shows an FC in testcase9. This FC is converted to two FCs in the resynthesized ECO list. The selected spare cells, the related nets and the newly created net are also shown. It can be seen that ECOS generates good solutions because the competition and internal nets among FCs are modeled and well solved.

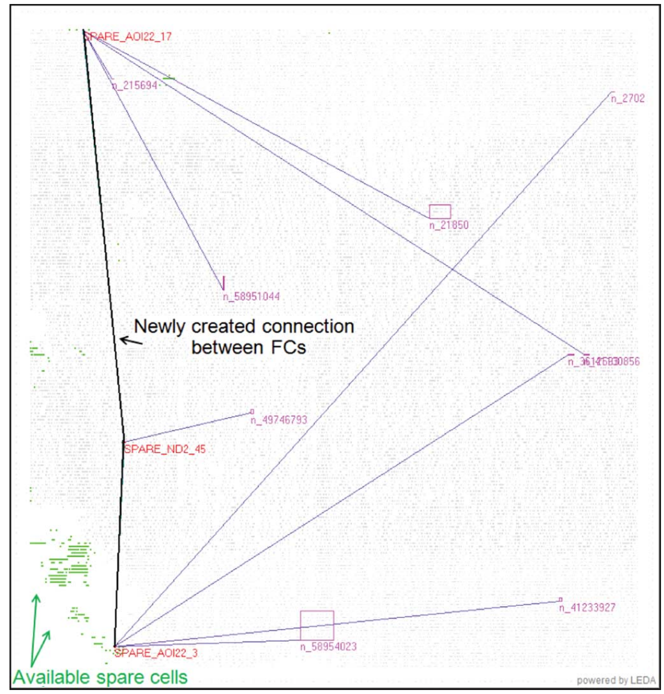


Fig. 12. Testcase3: The spare cells are gathered around the left-bottom corner, but FCs spread out.

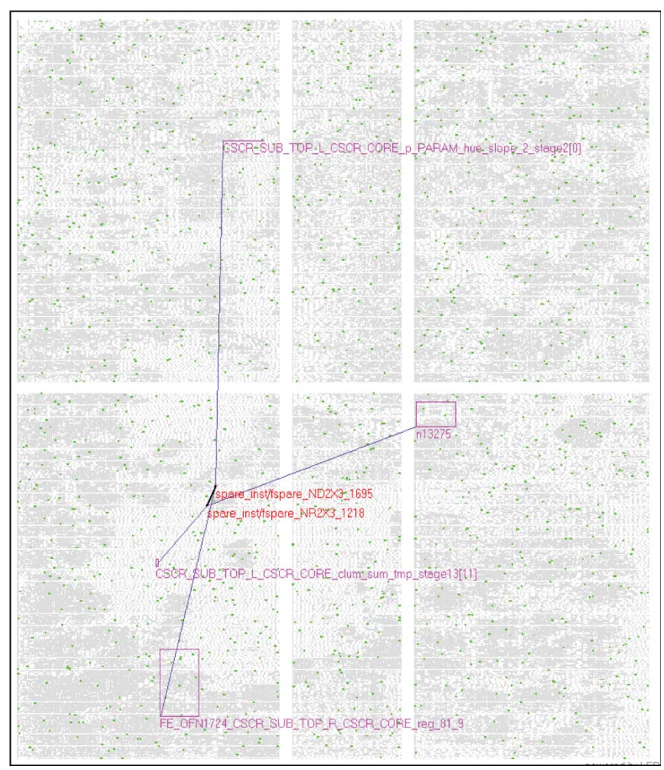


Fig. 13. Testcase9: The spare cells are evenly distributed; ECOS generate good solutions.

D. Congestion-Driven ECO Results

Based on the models of congestion estimation and preference described in Section IV, Table VII compares the congestion-driven ECO with wirelength-driven ECO. The initial congestion

TABLE VII
CONGESTION COMPARISON.

	Total cost (HPWL)		Congestion overflow	
Method	HPWL -driven	Congestion -driven	HPWL -driven	Congestion -driven
testcase1	4,051,021,350	4,051,765,230	27.88	27.03
testcase2	4,331,136,060	4,346,206,500	256.72	222.15
testcase3	4,555,641,140	4,576,576,200	858.22	485.24
testcase4	2,319,790	2,331,550	0.36	0.34
testcase5	13,012,260	13,017,860	1.64	1.64
testcase6	680,745,570	686,426,490	42.96	12.81
testcase7	16,896,832,010	16,896,832,010	529.71	529.71
testcase8	15,235,816,960	16,056,565,840	35,636.59	5,596.01
testcase9	7,869,825,400	8,182,118,080	6,348.58	547.55
Ratio	1.00	1.01	1.00	0.65

HPWL-driven: The preference values are defined as Section III.B.
Congestion-driven: The preference values are defined as Section IV.B, where $\lambda = 1.0$.

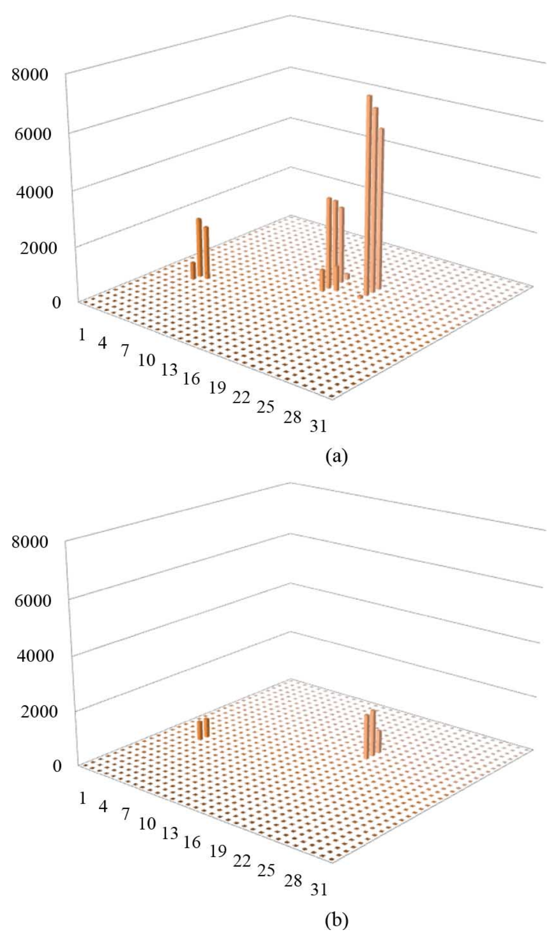


Fig. 14. The congestion map of testcase8. The height of each bar indicates its congestion overflow; 0 represents a congestion-safe bin (a) HPWL-driven ECOS (b) Congestion-driven ECOS.

map is constructed based on the pre-ECO bounding boxes of all nets. The routing supply is set to the maximum congestion value among the congestion bins in the initial map. This setting drives ECOS to select spare cells located in uncongested bins. In addition, the value of λ is set to 1.0 in our experiments.

Congestion overflow represents the total congestion values exceeding the routing supply, $\Sigma_j[C(b_j) - R(b_j)]^+$. It can be seen that congestion-driven ECOS on average reduces the congestion overflow by 35% with 1% HPWL overhead. When the pre-ECO bounding box of each FC spans over a large portion of the design, the available spare cells are likely located within the pre-ECO bounding box, thus no matter which spare cell is chosen, the resulting congestion values of the related bins are quite close. This phenomenon causes no improvement on testcase5 and testcase7. Fig. 14 shows the congestion maps generated by wirelength-driven and congestion-driven ECOS for testcase8. The design is divided into 32×32 grids and the spikes indicate bins with congestion overflow.

VI. CONCLUSION

In this paper, we have proposed a metal-only ECO synthesizer, named ECOS. ECOS integrates physical information into resynthesis and handles the competition among FCs by stable matching. We also extended ECOS to consider congestion. Future work includes the extension to consider mixed-type spare cells and to unify timing and functional ECO.

ACKNOWLEDGMENT

The authors would like to thank Dr. E. Y.-W. Tsai, Faraday Technology Corp., Hsinchu, Taiwan, for his valuable suggestions and providing the benchmarks.

REFERENCES

[1] International Technology Roadmap for Semiconductors (ITRS) 2007 [Online]. Available: http://www.itrs.net/
[2] A. Balasinski, "Optimization of sub-100-nm designs for mask cost reduction," *J. Microlithogr., Microfab., Microsyst.*, vol. 3, no. 2, pp. 322–331, Apr. 2004.
[3] K.-H. Chang, I. L. Markov, and V. Bertacco, "Reap what you sow: Spare cells for post-silicon metal fix," in *Proc. ACM Int. Symp .Phys. Design (ISPD)*, 2008, pp. 103–110.
[4] Z.-W. Jiang, M.-K. Hsu, Y.-W. Chang, and K.-Y. Chao, "Spare-cell-aware multilevel analytical placement," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2009, pp. 430–435.
[5] 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 Trans. Comput.-Aided Design*, vol. 26, no. 2, pp. 345–358, Feb. 2007.
[6] S. Krishnaswamy, H. Ren, N. Modi, and R. Puri, "DeltaSyn: An efficient logic difference optimizer for ECO synthesis," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 2009, pp. 789–796.
[7] Y.-P. Chen, J.-W. Fang, and Y.-W. Chang, "ECO timing optimization using spare cells and technology remapping," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 2007, pp. 530–535.
[8] C.-P. Lu, M. C.-T. Chao, C.-H. Lo, and C.-W. Chang, "A metal-only-ECO solver for input slew and output loading violations," in *Proc. ACM Int. Symp. Phys. Design (ISPD)*, 2009, pp. 191–198.
[9] K.-H. Ho, J.-H. R. Jiang, and Y.-W. Chang, "TRECO: Dynamic technology remapping for timing engineering change orders," in *Proc. ACM/IEEE Asia South Pacific Design Autom. Conf. (ASP-DAC)*, 2010, pp. 331–336.
[10] Y.-M. Kuo, Y.-T. Chang, S.-C. Chang, and M. Marek-Sadowska, "Spare cells with constant insertion for engineering change," *IEEE Trans. Computer-Aided Design*, vol. 28, no. 3, pp. 456–460, Mar. 2009.
[11] N. Modi and M. Marek-Sadowska, "ECO-map: Technology remapping for post-mask ECO using simulated annealing," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, 2008, pp. 652–657.
[12] 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 *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2009, pp. 408–411.

[13] I. H.-R. Jiang and H.-Y. Chang, "ECOS: A metal-only ECO synthesizer," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2010.
[14] , L. Wang, Y. Chang, and K. Cheng, Eds., *Electronic Design Automation: Synthesis, Verification and Testing*. : Elsevier/Morgan Kaufmann, 2009.
[15] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 69, pp. 9–14, 1962.
[16] S. Golson, "The human ECO compiler," in *Proc. Synopsys Users Group (SNUG)*, 2004, pp. 1–57.
[17] ABC: A System for Sequential Synthesis and Verification [Online]. Available: http://www.eecs.berkeley.edu/~alanmi/abc/
[18] R. E. Bryant, "Symbolic Boolean manipulation with ordered binary-decision diagrams," *ACM Comput. Surv.*, vol. 24, no. 3, pp. 293–318, Sep. 1992.
[19] P. Spindler and F. M. Johannes, "Fast and accurate routing demand estimation for efficient routability-driven placement," in *Proc. Design, Autom. Test in Europe Conf. Expo. (DATE)*, 2007, pp. 1226–1231.
[20] Z.-W. Jiang, B.-Y. Su, and Y.-W. Chang, "Routability-driven analytical placement by net overlapping removal for large-scale mixed-size designs," in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2008, pp. 167–172.
[21] Y. Zhang and C. Chu, "CROP: Fast and effective congestion refinement of placement," in *Proc. IEEE Int. Conf. Comput.-Aided Design (ICCAD)*, 2009, pp. 344–350.
[22] LEF/DEF Exchange Format: Reference Documentation Plus Parser [Online]. Available: http://www.si2.org/openeda.si2.org/projects/lefdef

Iris Hui-Ru Jiang (M'07) received the B.S. and Ph.D. degrees in electronics engineering from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1995 and 2002, respectively.

She has been with VIA Technologies, Inc. from 2002 to 2005. She is currently an Assistant Professor with the Department of Electronics Engineering and the Institute of Electronics, NCTU. Her current research interests include VLSI physical design and interaction between logic synthesis and physical design.

Dr. Jiang is a member of the Association for Computing Machinery and Phi Tau Phi.

Hua-Yu Chang received the B.S. degree from National Chengchi University, Taipei, Taiwan, in 1998, and the M.S. degree from National Chiao Tung University, Hsinchu, Taiwan, in 2001, both in computer science. He is currently working toward the Ph.D. degree in electronic design automation at the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei, Taiwan.

He has ten years' work experience in networking and computer graphics. He is currently a Technical Section Manager with VIA Technologies Inc. His current research interests focus on physical design as well as logic synthesis.