Net-based Move in SA-based Placement for a Switch-Block-Free Reconfigurable Device

Masato Inagi† Masatoshi Nakamura Tetsuo Hironaka Takashi Ishiguro

Graduate School of Information Sciences
Hiroshima City university
3-4-1 Ozuka-higashi, Asaminami, Hiroshima 731-3194, Japan
† inagi@hiroshima-cu.ac.jp

R & D Center
Taiyo Yuden Co., Ltd.
6-16-20 Ueno, Taito, Tokyo 110-0005, Japan
tad-ishiguro@jty.yuden.co.jp

Abstract— In this paper, we propose an enhanced SA-based placement algorithm for a switch-block-free reconfigurable architecture, introducing a move function that shifts all the logic cells which belong to a randomly selected net to the same direction. Although neighbor solutions generated by the move function are similar to the current solution, iteration of straightforward move functions that move a single logic cell at a time rarely generates such a solution. The move function improves reachability between good solutions, and thus improves the quality of the final solution.

I. Introduction

FPGAs realize a target circuit by realizing logic cells by LUTs and connecting wires among the logic cells by switch blocks. However, a switch block requires a large area and many routing layers, and some papers (*e.g.*, [1]) have reported that the area ratio of routing resources (*e.g.*, switch block) in a recent FPGA reaches 80%. To solve the problem, a switch-block-free reconfigurable architecture, called *MPLD*, has been proposed [3]. An MPLD consists of an array of *multiple-output LUTs* (*MLUTs*), which realize both logic cells and wire connections. When placing a circuit on an MPLD using a standard method based on simulated annealing (SA), a logic cell is randomly selected and replaced, and then the placement is evaluated to decide whether to accept it or not. This method, however, rarely translates regionally optimized sub-circuits to adjust their positions. In this study, to shorten the distance between good placement solutions, we propose a *net-based move* in neighbor solution generation which shifts the logic cells connected by a target net, and evaluate its effectiveness by experiments.

II. Target Architecture

Unlike in FPGAs, both logic cells and nets are realized by the same element, called MLUT, in MPLDs. An MLUT has N input and N output terminals, and is realized by a $2^N \times N$-bit SRAM module. A pair of bits of address and data lines is called an *address-data* (*AD*) *pair*. As shown in Fig. 1, an MPLD basically consists of MLUTs diagonally connected by AD pairs, called *adjacent lines*. In addition, an MLUT can have AD pairs that connect to distant MLUTs, FFs, and I/O pads. These AD pairs are called *distant lines*.

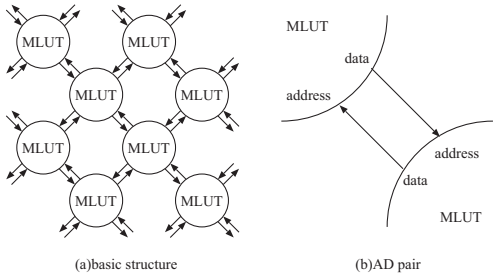

Fig. 1. Basic structure of MPLD

III. Placement Algorithm

Our placement algorithm is based on a SA-based algorithm [2], which generates a neighbor solution by randomly moving a single cell at a time. To reserve MLUTs for routing between MLUTs used for logic cells, its cost function considers detailed estimated wire congestion (Congestion) and distance between logic cells as obstacle (Nearness), in addition to total wire length (Length), which is also considered in placement for FPGAs (*e.g.*, [4]). That is,

$$\text{Cost} = p \times \text{Length} + q \times \text{Congestion} + r \times \text{Nearness}, \quad (1)$$

where p, q, and r are user-defined coefficients of Length, Congestion, and Nearness, respectively.

The main characteristic of MPLD placement is that MLUTs for routing need to be reserved around MLUTs used for logic cells.

IV. Net-based move

Here, we discuss net-based move.

The placement method proposed in [2] has two types of moves in its neighbor function: *exchange of cells* and *migration of cell*. (We call these moves *cell-based moves*.) All the solutions are mutually reachable by migration of cell, at least when the number of MLUTs is greater than the number of logic cells. This fact guarantees that the optimal solution is theoretically obtained by SA if the cooling schedule and initial/final temperatures are appropriate and adequate. In practical, however, it is likely that reachability between good solutions is reduced by the

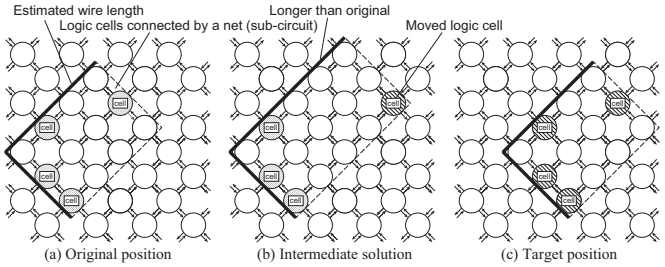

Fig. 2. Translation of sub-circuit based on cell-based move

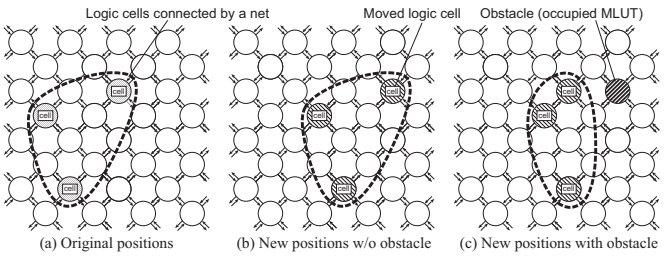

Fig. 3. Net-based move

limited moves. In particular, at a low temperature, which reduces the accept ratio of worse solutions, it is likely that a target circuit is partly optimally placed and is trapped at local optimal. Suppose that Fig. 2 (a) shows an optimally placed sub-circuit, and it should be replaced like Fig. 2 (c). Such a replacement only by cell-based moves needs to transit worse solutions like Fig. 2 (b). Moreover, all the cells need to be moved the same distance to the same direction. (Note that the distance between MLUTs is defined as the minimum number of adjacent lines among the paths on adjacent lines from one MLUT to the other.) However, the probability that such moves happen is small.

Therefore, we introduce a net-based move which shifts all the cells connected to a selected net to the same direction, to improve the mobility of optimized sub-circuits and thus the reachability to the optimal solution. In our net-based move, a net is randomly selected, and all the cells connected to the net are shifted the minimum distance to the same direction from top, right, bottom, left, top right, bottom right, bottom left, and top left. Note that the shifts to top, right, bottom, and left move logic cells two units because an MPLD is a diagonal grid of MLUTs. If the destination of a logic cell has been occupied by another cell as obstacle, the cell remains at the current MLUT (Fig. 3).

V. Experimental Results

We conducted experiments to evaluate the effectiveness of the net-based move. The placement and routing method proposed in [2] and the net-based move function were implemented in C language and executed on a PC with Core2Duo and 512MB memory. The target circuits were 16 circuits from ISCAS'89 benchmark circuit suite.

TABLE I
Placement and routing results of ISCAS'89

			cost			
circuit	**gates**	**area**	$\gamma = .0$	$\gamma = .1$	$\gamma = .2$	$\gamma = .5$
s27	10	15*30	4334	4172	4209	4237
s208	96	15*30	19810	19201	19349	19470
s298	119	15*30	37043	35937	35931	35925
s344	160	15*30	46255	44936	44849	45140
s349	161	15*30	45873	44889	44764	45201
s382	158	15*30	47088	46081	45743	46148
s386	159	15*30	65335	64265	63894	64144
s400	162	15*30	48304	47147	47018	47294
s420	196	15*30	48876	47781	47628	48238
s444	181	15*30	47772	46416	46230	46249
s510	211	33*36	131233	129184	128526	129830
s526	193	15*30	58795	57105	56942	57141
s526n	194	15*30	59575	58495	58139	58354
s641	379	33*36	106173	102241	102004	103108
s713	393	33*36	186900	182875	182237	182922
s820	289	33*36	179457	171583	172036	173072
AveR			1	0.973	0.972	0.978

In the experiments, each of p, q, and r was set to 0, 1, or 5. For each circuit, each pattern of p, q, and r was tried 10 times and the average cost of the final placement solutions was obtained. The target MPLD for each circuit was the smallest one that can contain the circuit among 15×30, 33×36, and 63×60 MPLDs. Note that an $a \times b$ MPLD has b vertical columns each of which has a MLUTs.

The experimental results are shown in Table I. The column labeled "area" lists the size of the target MPLD for each circuit. γ represents the ratio of net-based moves in all the neighbor solution generations (*i.e.*, $\gamma = .0$ means the conventional method [2]). The row labeled "AveR" shows the ratio against the results by the conventional method. As shown in the row, the net-based move improved the average cost of the final placement solution. In addition, the average cost of the final solutions for each circuit was robustly improved. The results showed the effectiveness of the net-based move.

VI. Conclusions

In this paper, we proposed an enhanced SA-based placement method with a net-based move for a switch-block-free reconfigurable architecture. Experimental results showed that the proposed method robustly improves the cost of the final solutions.

References

[1] A. Singh and M. Marek-Sadowska, "FPGA Interconnect Plannning," in *Proc. ACM Int. Workshop SLIP 2002*, 2002, pp.23–30.

[2] M. Nakamura, M. Inagi, K. Tanigawa, T. Hironaka, M. Sato, and T. Ishiguro, "EDA Environment for Evaluating a New Switch-Block-Free Reconfigurable Architecture," in *Proc. Int. Conf. ReConFig 2011*, Nov. 2011, pp.448–454.

[3] N. Hirakawa, M. Yoshihara, K. Tanigawa, T. Hironaka, and M. Sato, "A PLD Architecture for High Performance Computing," in *Proc. the 2008 Int. Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems*, 2008, pp.35–42.

[4] V. Bets and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in *Proc. Int. Conf. FPL 1997*, 1997, pp.213–222.