FPGA-based Heterogeneous Solver for Three-Dimensional Routing

Kento Hasegawa, Ryota Ishikawa, Makoto Nishizawa, Kazushi Kawamura,
Masashi Tawada, and Nozomu Togawa
Dept. of Computer Science and Communications Engineering, Waseda University
e-mail: {kento.hasegawa, ryota.ishikawa, makoto.nishizawa, kazushi.kawamura, tawada, togawa}
@togawa.cs.waseda.ac.jp

Abstract— A heuristic algorithm is one of the approaches to solve an NP-hard problem. In order to enhance the capability of the system, heterogeneous computing is often adapted. In this paper, we propose an FPGA-based heterogeneous solver for three-dimensional routing. The proposed system is implemented into multiple FPGA boards and a single-board computer. The experimental results demonstrate that the proposed system outperforms a single FPGA system.

I. Introduction

Due to the advance in IC manufacturing, recent highly integrated system-on-chips (SoCs) have been manufactured adapting three-dimensional placement and routing techniques. Since an enormous number of function blocks have to be placed, effectively connecting all of them is a difficult problem.

As exemplified above, a three-dimensional routing problem is difficult to solve because of its huge solution space. To tackle this problem, heuristic algorithms are often adapted, which can effectively solve NP-hard problems. However, the solution obtained from a heuristic algorithm is not guaranteed to be optimal. To stably obtain a more optimal solution, simultaneously using several types of algorithms and running them several times are both important.

In order to accelerate the computation, field-programmable gate arrays (FPGAs) are often utilized. Due to the effort of FPGA vendors, we can easily implement a program into an FPGA with useful synthesis tools utilizing hardware description language as well as high-level synthesis. In addition, a newly developed SoC has both of a processor and an FPGA part which are interconnected.

From the discussion above, we propose a three-dimensional routing solver utilizing multiple FPGA boards. Since several types of algorithms are implemented into multiple FPGA boards, we can realize heterogeneous parallel computing. Our goal is to obtain more optimal solutions with less time.

II. Three-dimensional Routing

Here we introduce a three-dimensional routing problem. In order to simplify the problem, we consider a three-dimensional puzzle [1]. In a three-dimensional puzzle, a field with $X \times Y \times Z$ blocks is given. In this field, pairs of numbers from 1 to N are exclusively placed in each block. Each number shows the terminal of a line, and N is the number of lines. Each line can pass through non-occupied blocks. Our goal of this puzzle is to connect the same numbers by lines without crossing.

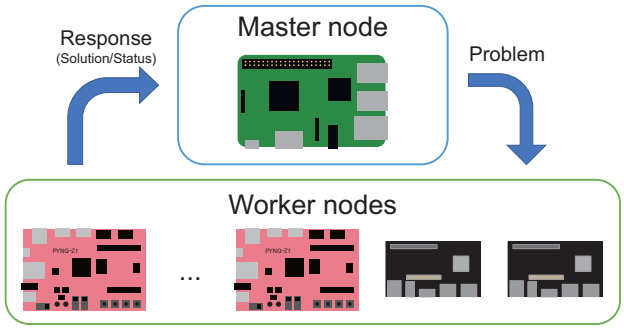

Fig. 1. Overview of the proposed system.

This three-dimensional puzzle is difficult to solve. First, since we have to search paths for N lines in a three-dimensional space, the solution space is quite large. Moreover, because the solution paths must not cross each other, we have to explore a optimal solution.

A heuristic algorithm is one of the approaches to tackle this problem, but that still take so long until obtaining an acceptable solution. In this paper, we propose an FPGA-based heterogeneous solver where multiple algorithms are implemented into different FPGA boards.

III. FPGA-based Heterogeneous Solver for Three-Dimensional Routing

In order to effectively solve the puzzle, we propose an FPGA-based heterogeneous solver for three-dimensional routing puzzle.

A. System Architecture

The proposed system is composed of two types of nodes: a master node and multiple worker nodes. The master node and the worker nodes are connected with a TCP/IP network. The overview of the system is illustrated in Fig. 1.

The master nodes runs a server on a single-board computer. The server provides GUIs of the system, and manages the whole system. When a puzzle problem is input to the server from GUI, the server distributes it to the worker nodes. When a response is sent from a solver, it is stored on the server. After a few seconds from distributing problems, the server gathers the stored response and obtains the most effective solution.

978-1-7281-4123-7/20/$31.00

The worker node is an FPGA board which has a communication program and a solver program. In the worker nodes, multiple solvers are implemented into different FPGA boards. The FPGA boards used in our proposed system have both a processor part and an FPGA part in an SoC. Fig. 2 depicts the

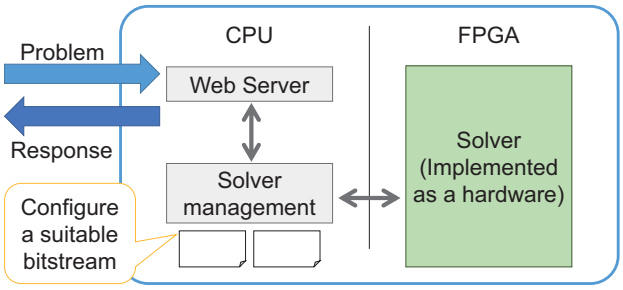

Fig. 2. Overview of a worker node.

behavior of a solver. The solver first receives a puzzle problem from the master node using the processor part. The received puzzle problem is soon passed to the solver implemented on the FPGA part, and the solver tries to solve it. After the trial, the response is returned to the processor part, and is sent to the master node.

B. Solver Algorithms

In order to effectively solve a puzzle, we implement multiple solver algorithms. When we implement a solver algorithm into an FPGA, resource limitation is one of the problems. Since the resource and performance of a solver algorithm is in a trade-off relationship, we implement several types of algorithms with different data structures. Route information for each line is one of the most dominant data.

TABLE I
SOLVER ALGORITHMS AND THEIR DATA STRUCTURE.

Algorithm	Data Structure
Solver A	Two-dimensional array ($N \times L_{\max}$)
Solver B	One-dimensional array ($N \times L_{\text{avg}}$)
Solver C	Field data ($X \times Y \times Z$)

Table I shows the summary of the data structure. The implemented solver algorithms are described as follows:

1) **Solver A**: First, connect each number with the other one by A* algorithm. Then, randomly pick one of the lines, and re-route it. Repeat this until all the crosses are removed. The path data for each line is stored in a two-dimensional array. The size of the array becomes $N \times L_{\max}$ where $L_{\max}$ shows the maximum path length for each line.
2) **Solver B**: This algorithm is an updated version of the Solver A. The data structure adapted in Solver A is redundant. Therefore, Solver B stores the path data for each line in a one-dimensional array with no padding. The size of the array is $N \times L_{\text{avg}}$ where L_{avg} shows the average path length for each line.
3) **Solver C**: This algorithm only uses the field data. It tries to connect each line on the field.

IV. Implementation and Experimental Results

We implemented the proposed system into FPGA boards and a single-board computer. We used five Digilent PYNQ-Z1 [2] and two AVNET Ultra96 [3] as the worker nodes, and one Raspberry Pi 3B+ as the master node. Solver A is implemented into two PYNQ-Z1, Solver B is implemented into two PYNQ-Z1 and Ultra96, and Solver C is implemented into one PYNQ-Z1. Fig. 3 shows the picture of the proposed system.

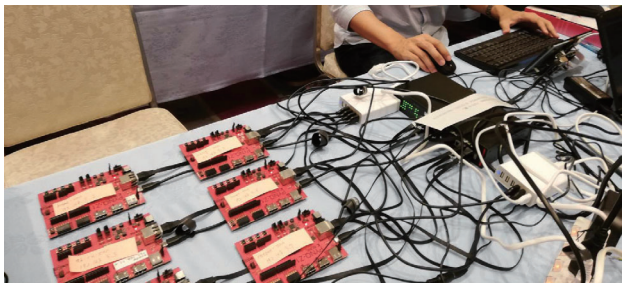

Fig. 3. Implemented system.

In the experiments, 34 puzzles are given to the proposed system. For comparison, we used a simple system where eight Solver A nodes are connected. We evaluate the number of solved problems and processing time. Table II shows the results of the experiments. The processing time in Table II shows the relative time based on the simple system. As shown in Table II, the proposed system outperforms the simple system in terms of both the number of solved problems and processing time.

TABLE II
EXPERIMENTAL RESULTS OF THE PROPOSED METHOD.

	Solved Problems (/34)	Processing Time
Simple System	20	1
Proposed System	28	0.42

V. Conclusion

In this paper, we proposed an FPGA-based heterogeneous solver for three-dimensional routing. The proposed system is implemented into multiple FPGA boards and a single-board computer. The experimental results demonstrate that the proposed system outperforms a simple system.

Acknowledgments

This paper was supported in part by Grant-in-Aid for Scientific Research (No. 19H04080 and No. 18J14325).

References

[1] "DA Symposium 2018 Algorithm Design Contest," (in Japanese). https://dasadc.github.io/adc2018/
[2] "Digilent PYNQ-Z1." https://reference.digilentinc.com/reference/programmable-logic/pynq-z1/reference-manual
[3] "AVNET Ultra96." https://www.avnet.com/opasdata/d120001/medias/docus/187/aes-ultra96-v2c (1).pdf