2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines

Megrez: Parallelizing FPGA Routing with Strictly-Ordered Partitioning

Minghua Shen* and Guojie Luo*†
*Center for Energy-efficient Computing and Applications, School of EECS, Peking University, China
†Collaborative Innovation Center of High Performance Computing, NUDT, China
Email: {msung, gluo}@pku.edu.cn

I. Introduction

FPGAs play a crucial role in the space of customizable accelerators over the next few years. A chief limiting factor is that FPGA CAD tools are cumbersome and time-consuming to most application developers. Routing is the most complex step in FPGA design flow and NP-complete problem. The PathFinder routing algorithm [1] is in dominant use in FPGA CAD research. However, PathFinder is sequential in nature and lengthy in runtime. Parallelization has the potential to solve the issue but faces non-trivial challenges. In this work we introduce Megrez that uses strictly-ordered partitioning to explore the parallelism on GPU. Experimental results show that Megrez achieves an average of 15.13× speedup on GPU with negligible influence on the routing quality.

II. Methodology

The methodology of Megrez consists of two phases: partitioning and parallelization. The partitioning phase separates the nets into multiple subsets while maintaining ordinary net ordering for same routing quality. In the parallelization phase, the nets in the same subset are routed in parallel, and the subsets are routed in serial.

We partition the nets into M subsets s_1, s_2, ..., s_m with r_1, r_2, ..., r_n nets respectively. Note that r_1+r_2+...+r_n = n, and the relative ordering for the subsets must be preserved. Specifically, the nets in the same subset are independent, and the dependent nets are distributed in different subsets. And it is completely different from the previous partitioning methods [2], [3]. The net partitioning can be solved by dynamic programming in quadratic time. We will analyze the time complexity of the partitioning below. Table I gives the related notations.

TABLE I
Notations for the partitioning problem

Notation	Description
$E[j][i]$	The feasibility indicator whether the nets $r_j, r_{j+1}, ..., r_i$ are independent.
$C[i]$	The minimum number of subsets for the nets from r_1 to r_i with the strictly-ordered and independent properties.

Specifically,

$$E[j][i] = \begin{cases} 1, & \text{independent} \\ +\infty, & \text{otherwise} \end{cases}$$

The algorithm consists of two stages: the precomputation stage and the dynamic programming stage. We can precompute the $E[j][i]$ using simple pair-wise testing in worst-case quadratic time. In practice, even this simple algorithm terminates fast, because the size of independent subsets is limited.

Based on the precomputed $E[j][i]$, we can start the dynamic programming algorithm. The minimal number of strictly-ordered subsets of the first i nets satisfies

$$C[i] = \begin{cases} 1, & i = 0 \\ \min_{j=0}^{i-1}\{C[j] + E[j+1][i]\}, & i \geq 1 \end{cases}$$

The solution to our problem is $C[N]$. It is obvious that computing $C[N]$ takes $O(N^2)$, given $E[i][j]$. Thus, the time complexity of the overall dynamic programming algorithm is quadratic.

We use dynamic parallelism to parallelize the FPGA routing on GPU. The previous work [4] has also demonstrated the effectiveness of dynamic parallelism with GPU techniques. We collect the independent nets according to their original ordering, and make sure that their concurrent routing will not affect their routing results, comparing to the sequential routing. With optimal partitioning, we start to route the first subset of independent nets concurrently and then route the next subset until all the subsets are processed. It is evident that this subset can be routed in parallel without affecting the routing results.

III. Evaluation

The parallel routing method described in this paper is implemented with C++ and CUDA. The experiments are performed on a Linux server with a 6-core Intel Xeon E5-2620 CPU at 2.2GHz and 32 GB shared memory, equipped with a Tesla K40c GPU having 2880 cores in 15 streaming multiprocessors and 12 GB video memory. The baseline for comparison is the original VPR 7.0 router [5]. Experimental results show that Megrez achieves an average speedup of 15.13× on a single GPU. It effectively maintains deterministic results on different platforms and always produces the same solutions as the serial version.

IV. Acknowledgement

This work is partly supported by National Natural Science Foundation of China (NSFC) Grant 61520106004.

References

[1] L. McMurchie and C. Ebeling. Pathfinder: A negotiation-based performance-driven router for FPGAs. FPGA 1995.
[2] M. Gort and J. Anderson. Deterministic multi-core parallel routing for FPGAs. FPL, 2010.
[3] M. Shen and G. Luo. Accelerate FPGA routing with parallel recursive partitioning. ICCAD 2015.
[4] M. Shen and G. Luo. Corolla: GPU-accelerated FPGA routing based on subgraph dynamic expansion. FPGA 2017.
[5] J. Rose et al. VPR 7.0: Next generation architecture and CAD system for FPGAs. TRETS 2014.

978-1-5386-4037-1/17 $31.00 © 2017 IEEE
DOI 10.1109/FCCM.2017.18

IEEE computer society