HARDWARE ORIENTED CLASSIFICATION OF BINARY RELATIONS OF GRAPH-SCHEMES OF PARALLEL ALGORITHMS

I.A. Martynov, E.I. Vatutin, V.S. Titov

305040, Kursk, Russia Southwest State University, Department of Computer Sciences, 50 let Oktyabrya, 94 Phone: 8 (4712) 58-71-05

Abstract: The article describes the system of binary relations for graph-schemes of parallel algorithms, which are a special type of graphs with some special properties. Given a generalized parallel algorithm for building a matrix of binary relations based on private algorithms determine the connection, follow, alternative and parallelism relations. It is shown that using of specialized storage devices with a matrix structure can achieve a substantial gain in time during building the matrix of relations. In this case, the existing hardware with parallel structure (eg, GPU) are characterized by lower gets the win in time.

Keywords: graph-schemes of parallel algorithms, matrix of relations, binary relations classification, hardware accelerators design, logic multicontrollers design

In many problems associated with parallel representation and parallel processing of information there are a need to work with graph-schemes of parallel algorithms presented as a special kind of labeled graphs with some special properties [1, 2]. These include some problems from parallelization of computations, parallel programming and homogeneous multi-module systems design. It often requires the classification of binary relations between vertices of processed graph-scheme that sufficiently conveniently put into practice with using a matrix of relations M_R [3, 4].

This data structure is a square matrix with elements m_{ij} , $i, j = \overline{1, N}$, where N is a number of vertices within graph-scheme. Each element of matrix is a set of binary relations between pair of vertices $a_i \bowtie a_j$ that stored as a bit vector [11]. This set of binary relations can include follow relation ν , connect relation φ , parallelism relation ω and alternative relation ψ . Follow and connect relations can be used to determine the number and composition of links set during separating source graph-scheme to blocks in order to determine control dependencies and to calculate and minimize interconnect traffic control transfer and the number of interconnect commands [5]. Parallelism relation is important to comply with condition for the absence of parallel vertices consisting of one blocks of separation that is of interest during design of homogeneous multi-module systems of logic control within logical multi-controllers approach [1–3].

Follow relation has the property of transitivity (fig. 1) that allows for its transitive closure based on source information formed by considering of all arcs of graph-scheme and pairs of vertices incident to them [4].All another binary relations don't have the property of transitivity that illustrated at the fig. 2 at example of connect relation.

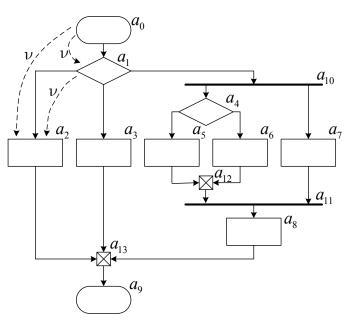


Figure 1.Example to explain the transitivity of the follow relation: $a_1 \nu a_0$, $a_2 \nu a_1 \Rightarrow a_2 \nu a_0$

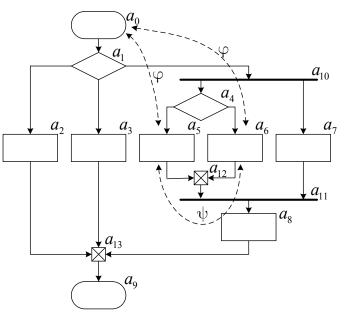


Figure 2. Example that shows none transitivity of connect relation: $a_0\varphi a_5$, $a_0\varphi a_6$, but $\neg(a_5\varphi a_6)$, as $a_5\psi a_6$

The required transitive closure can be implemented using Floyd–Warshall algorithm [6] that has time asymptotic complexity $O(N^3)$. During processing of large graph-schemes computing time can be up to 23% [7], therefor is of interest attempt to its decreasing. At articles [8, 9] are presented estimates of gain in time for matrices multiplication problem with various methods of algorithmic and high-level optimization (CPU cache using optimization, GPU global memory using optimization, loops unrolling). The problem consider edisvery similar to transitive closure problem which actually multiply Boolean matrices and resulting gain in processing speed (performance) ranged from several times to several hundred times depending on the hardware used and the composition of optimizations. Feature of the Floyd–Warshall algorithm is a special order of the nested loops, which from one hand allows finding desired transitive closure in one pass with three nested loops but on the other hand imposes

restrictions on the order of the sequential processing rows and columns of the matrix reducing the degree of parallelism of processed data and as a result reducing then theoretical gain from GPU using at least by N times negating usefulness of GPU in this problem. Instead, to minimize the amount of time for transitive closure it is possible to develop special hardware oriented accelerator based on using of special multi-port matrix memory [10] with structure that shown at fig. 3. This approach can significantly speed up the processing.

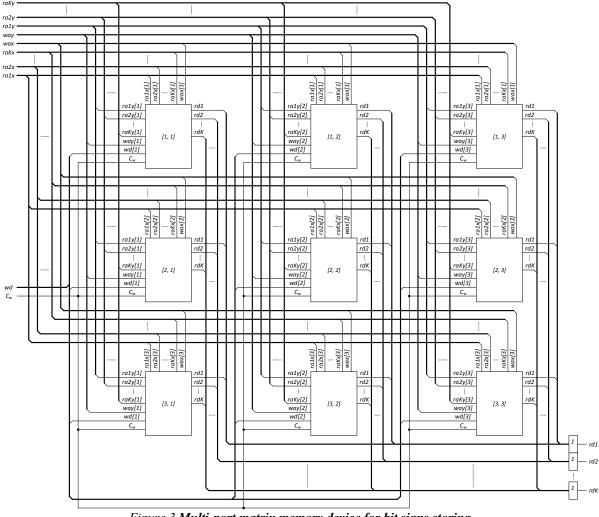


Figure 3. Multi-port matrix memory device for bit signs storing

Connect relation can be obtained directly after follow relation determining using the identity $a_i \varphi a_j \Leftrightarrow (a_i \nu a_j) \lor (a_j \nu a_i)$ (in other words pair of vertices if connected by control when one of them is accessible from another or visa versa). At the practice implementation determining of connect relation can be partially aligned at time with the elucidation of follow relation. Using discussed above matrix memory determining of connect relation can require additional diagonal links between memory cells which symmetric with respect to main diagonal that provide fast identifying of connect relation in constant time independent of dimension *N* of the problem.

Clarification of the alternative relation started by finding pairs of alternative vertices and vertices of combining alternative arcs corresponding to each other within current alternative fragment of graph-scheme. After that builds the paths between selected pair of vertices and adds desired alternative relation between vertices that make up different alternative ways. Shown approach has some parallel operations, however, taking into account a number of restrictions it is advisable to implement this operation on CPU passing at the hardware level only result of classification of alternative relation. This can significantly decrease complexity of hardware accelerator structure without departing of matrix organization of memory and slightly increase the total time of processing. In this case, alternative relation explanation can be aligned in time with elucidation of follow and connect relations.

Parallel relations can be obtained immediately after the explanation of connect and alternative relations using identity $a_i \omega a_j \Leftrightarrow \neg(a_i \varphi a_j) \land \neg(a_i \psi a_j)$. With using matrix memory this operation can be efficiently implemented by logical combining of cells that store respective bit signs of relations as shown at fig. 4 without significant increasing of matrix storage complexity.

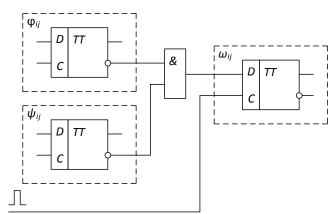


Figure 4. Circuit implementation of operation for determining the parallelism relation

Thus, as in case with connect relation, time complexity of operation do not depend from dimension N of a problem.

Common parallel hardware oriented algorithm for classification of binary relations based on shown above separate partial algorithms with its features presented at fig. 5.

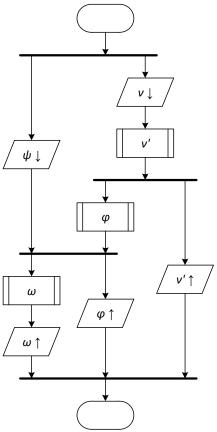


Figure 5. Parallel hardware oriented algorithm for matrix of relations filling. Arrow down at IO operators denotes data loading from RAM to matrix storage, arrow up – resulting data storing from matrix memory to RAM. Operator vertices are private algorithms for determining of specific relation using hardware accelerator

With its using can be implemented transfer of the time-consuming operations during filling matrix of relations from program to hardware level that taking into account their specific and provide significant decreasing of computing time.

REFERENCIES

1. Vatutin E.I., Zotov I.V., Titov V.S. et al. Combinatorial-logical problems of synthesis separations of parallel logic control algorithms during logical multi-controllers design / Kursk: KurskSTU, 2010. 200 p.

2. Vatutin E.I. Logical multi-controllers design. Getting separations of parallel graph-schemes of algorithms. Saarbrucken: Lambert Academic Publishing, 2011. 292 p.

3. Organization and synthesis of micro program multimicrocontrollers / I.V. Zotov, V.A. Koloskov, V.S. Titov et al. Kursk: Kursk, 1999. 368 p.

4. Vatutin E.I., Zotov I.V. Relation matrix building applied to the problem of optimal separation of parallel logic control algorithms // Proceedings of Kursk State Technical University. Kursk, 2004. № 2. pp. 85–89.

Vatutin E.I. The problem of interblock traffic estimation during getting separations of parallel logic control algorithms // Education, Science, Production. Belgorod, 2006.
<u>https://en.wikipedia.org/wiki/Floyd–Warshall algorithm</u>

7. Vatutin E.I. An analysis of bottlenecks of program implementation of parallel-sequential method for getting separations of parallel algorithms // Optoelectronic devices in recognition

systems, image processing, and symbolic information processing (Recognition – 2013). Kursk: Southwest State University, 2013. pp. 235–237.

8. Vatutin E.I., Martynov I.A., Titov V.S. The CPU real performance estimation for matrices multiplication problem using single-threaded software implementation // Proceedings of Southwest State University. Series: Control, Computer Sciences, Informatics, Medical devices. 2013. № 4. pp. 11–20.

9. Vatutin E.I., Martynov I.A., Titov V.S.The GPU real performance estimation for matrices multiplication problem using CUDA // Proceedings of Southwest State University. Series: Control, Computer Sciences, Informatics, Medical devices. 2014. № 2. pp. 8–17.

10. Martynov I.A., Vatutin E.I., Titov V.S. Multi-port matrix memory device for bit signs storing // Medical and Ecological Information Technology. Kursk, 2014. pp. 124–126.

11. Vatutin E.I. Optimizing of sets processing // Medical and Ecological Information Technology. Kursk, 2005. pp. 145–147.