УДК 004.272.44:519.17

A.A. Bilenko

GRAPH PARTITIONING METHODS FOR COMPUTATIONS IN RECONFIGURABLE SYSTEMS

Abstract. *Graph partitioning problem for reconfigurable systems is reviewed. Analysis of methods of graph partitioning is done. Recommendations on theirs usage are given.*

Keywords: *reconfigurable systems, data flow graph, graph partitioning.*

А.А. Биленко

РАЗБИЕНИЕ ГРАФОВ НА ПОДГРАФЫ ДЛЯ ПОСТРОЕНИЯ ВЫЧИСЛЕНИЙ В РЕКОНФИГУРИРУЕМЫХ СИСТЕМАХ

Аннотация. *Рассматривается постановка задачи разбиения графа на подграфы. Проведен анализ методов разбиения графа вычислений на подграфы для применения в реконфигурируемых вычислительных системах. Даны рекомендации по использованию приведенных методов.*

Ключевые слова: *реконфигурируемые системы, информационный граф, разбиение графов.*

А.О.Біленко

РОЗБИТТЯ ГРАФІВ НА ПІДГРАФИ ДЛЯ ПОБУДОВИ ОБЧИСЛЕНЬ У РЕКОНФІГУРОВАНИХ СИСТЕМАХ

Анотація. *Розглядається постановка задачі розбиття графа на підграфи. Наведено аналіз методів розбиття графа обчислень на підграфи для використання в реконфігурованих обчислювальних системах. Надано рекомендації щодо використання наведених методів.*

Ключові слова*: реконфігуровані системи, інформаційний граф, розбиття графів.*

Introduction. Performance increase issue is the topic of interest for computer scientists. One approach to increase computer system performance is based on reconfiguration. Reconfigurable systems allow implementing arbitrary computations based on finite structure of processor elements. As far as it's needed to perform arbitrary computations on the finite structure, it's obvious that the algorithm of the computation should be split on several parts. These parts have to be mapped on the reconfigurable structure. Typically, graphs are used to represent computations for reconfigurable systems, that's why graph partitioning algorithms are very important for such systems.

Graph partitioning of weakly connected sub-graphs is a part of the method of computation preparation for reconfigurable system [1]. The choice of graph partitioning algorithm may affect on rational hardware resources utilization, increase of parallel parts of the computation, minimization of external memory usage.

Statement of the problem. The following definitions and limitations are used to set the graph partition in gtask:

$G(V,E)$ is a graph, where V is a set of its vertexes, E is a set of its edges.

$$E = \{e_{ij} = (i, j) \mid \forall v_i, v_j, adj(v_i, v_j)\},$$

where

$$adj(v_i,v_i)=\begin{cases} true, v_i \ connected\,to \ v_j \\ false, v_i \ isn't \ connected\,to \ v_j \end{cases}$$

Edges and vertexes have the following weights

$$W_E = \{w_e(e_{ij}) \in N_+ \mid e_{ij} \in E\},$$
$$W_V = \{w_v(v_i) \in N_+ \mid v_i \in V\}$$

Weights are positive natural numbers: $W_E, W_V \in N_+$.

One of the simplest examples of graph partitioning works with a graph which has edges and vertexes with weighs equal to one. And such a graph is partitioned onto two parts, containing two disjoint sets of vertexes V_1 and V_2. The power of sets V_1 and V_2 has to have close values and the number of edges which connect vertexes $v_i \in V_1$ and $v_j \in V_2$ should be minimal:

$$\begin{cases} V = V_1 \cup V_2 \ \grave{e} \ V_1 \cap V_2 = \varnothing \\ |V_1| - |V_2| \to \min \\ \left|\{e_{ij} \in E \mid v_i \in V_1 \ \grave{e} \ v_j \in V_2\}\right| \to \min \end{cases} \quad (1)$$

Generally, graph partitioning task is being set taking into account weights of edges and vertexes. In this case, sums of weights of all vertexes in sub-graphs ω_i should be equal and

sums of weights of all edges connecting different sub-graphs should be minimal:

$$\begin{cases} V = \bigcup_{i=1}^{k} V_i \ , \ V_i \cap V_j = \varnothing, \forall i, j, i \neq j \\ \omega_j = \sum_{v_i \in V_j} w_v(v_i) \ , \ \omega_p - \omega_q \to \min \\ p, q \in \{1,2..k\} \quad p \neq q \\ \omega_e = \sum_{e_{ij} \in E, v_i \in V_p, v_j \in V_q, q \neq p} w_e(e_{ij}) \to \min \end{cases} \quad (2)$$

Analysis of graph partitioning algorithms applications. Graph partitioning algorithms should take into account several properties and limitations while using for reconfigurable systems:

absence of cyclic dependencies between sub-graphs in order to support von-Neumann programming determinism [1,2];

input edges of vertexes with several input edges have to be connected with operational vertexes, but informational, of a subgraph [2].

Graph partitioning problem is not a trivial problem and can be classified as a NP-problem. This means that there's no optimal algorithm working with the limiting behavior function $O(q^n) \ \forall \ n \in N$ for an arbitrary graph $G(V,E)$and$q = |V| + |E|$. That's why, all partitioning algorithms are estimated by performance and quality parameters. Tradeoff between performance and quality parameters depends on reconfigurable system structure.

To estimate the quality parameter of a graph partitioning, system (2) is used. According to it, sums of weights of edges connecting sub-graphs of a partitioned graph and quantity of vertexes in sub-graphs define the quality parameter of the partitioning

$$Q = \alpha_1(\omega_p - \omega_q)/\max(\omega_p - \omega_q) + \alpha_2\omega_e/\max(\omega_e) \quad (3)$$

where $\alpha_1+\alpha_2 = 1$, $\alpha_1$andα_2 define the value of power equality of sub-graphs' vertexes sets and quantity of connecting edges.

To estimate time parameter of a graph partitioning, limiting behavior function is used. Estimations of time and quality parameters of graph partitioning algorithms are presented in the table.

Graph partitioning algorithms are widely used in VLSI and FPGA design [2]. Typically, multilevel graph partitioning algorithms are used for inaccurate partitioning and Kernigan-Linn based algorithms for increasing the quality of given partitions.

Conclusion. Nowadays, several graph partitioning algorithms types are used to reach optimal quality/time tradeoff. To reach this tradeoff it's needed to combine usage of high-performance partitioning algorithms with algorithms working with a high-quality. The following approach can be used: perform graph partitioning in to $log(|V|)$ parts with a high-performance algorithm and increase partitions quality with high-quality algorithm. That's why graphs containing a big number of vertexes (more than 1000 vertexes) have to be partitioned with a combination of partitioning algorithms. It's optimal to apply spectral bisection following by Kernigan-Linn algorithm for such graphs. In practice, complexity of such algorithms combinations is considered to be $O(|V|log(|V|))$.

1. Timeandqualityestimationsforwell-knownpartitioningalgorithms

Algorithm	Time estimation	Quality estimation,Q (3)
Kernigan-Linn[3]	$O(\|V\|^3 log(\|V\|^2) \cdot k)$, k is a number of iterations to increase the quality.	The best partitioning quality based on iterative variant selection
Spectral bisection[3]	$O(\|V\|^2) + O(\sum_{i,j} \| Ek_{i,j} \|)$, Ek_{ij} is a set of edges between partitioned sub-graphs.	Good partitioning quality. Quality parameter is reached iteratively by setting the threshold of the sum of edges weights according to (2)
Multilevel partitioning [3]	Typical case: $O(\|V\|^2)$ Worst case:	Worst partitioning quality

	$O(\|V\|^2 log(\|V\|^2)) + O(\|E\|^2)$	

References

1. Bilenko A., Sitnikov V.S. Analysis of computation systems development based on reconfigurable systems. – Kharkov: Radioelectronic and computer systems – 2010. – P.212–214 [in Ukrainian].

2. Kalyaev I.A., Levin I.I. Semernikov E.A., Shmoilov V.I. Reconfigurable multipipeline computation structures–Rostov/D: SSC RAS. –2008. – 320 p. [in Russian].

3. Hendrickson B., Leland R.A Multilevel Algorithm for Partitioning Graphs. – NY: Proceedings of Supercomputing. – Vol. 1. – 1995. – P.15–25 [in English].

Received 14.02.2012

Bilenko Anatoliy A.,
aspirant, Odessa national polytechnic university.
Tel. +380919095405,
e-mail:
anatliy.bilenko@gmail.com.