Efficiency enhancement method for distributed CAD systems in fog-computing environments

V M Kureychik, I B Safronenkova

Autonomous federal state institution of higher education «Southern Federal University», Institute of computer technologies and information security, 44, Nekrasovskiy Ave., Taganrog, 347900, Russia

E-mail: safronenkova050788@yandex.ru

Abstract. This paper presents the new method, which is promising for circuit partitioning problem (CPP) solving in distributed computer-aided systems (DCADs) in terms of fog-computing environments. The circuit partitioning problem belongs to NP-hard one. Nowadays parallel computing is widely used for NP-hard problem solving. One of the most important criteria of DCADs performance is the speed of a workload distribution problem solving. The time needed for this procedure can be long enough to affect the computational process. Fog-computing environments have some peculiarities, which affect the workload distribution problem solving. A proposed method allows getting a solution of a satisfactory quality. The fundamental difference of this method is an ontological analysis, which is based on applied algorithms and available resources. It allows decreasing the search space of candidate nodes for workload distribution problem. The novelty of this paper is an ontological approach to workload distribution problem solving in DCADs in fog-computing environments.

1. Introduction

The fog-computing paradigm is promising in terms of distributed problems solving, including the design problems. Circuit partitioning problem belongs to NP-hard problem. Parallel computing allows partitioning a problem. So, parallel computing is ubiquitous applied for getting a solution in case of high complexity problems. Distributed CAD systems operate on the basis of distributed computing [1]. One of the most important criteria of DCAD performance, which depends on many factors such as search optimization methods, parallelized algorithm methods and ways of algorithm segment partition [2,3]. It also depends on the speed of workload distribution problem solving, which is topical in the terms of fog-computing implementation [4]. In case of fog-computing a computational problem has a location and an opportunity to stop the process and to save context data [5]. In other words, the time needed for this procedure affects the computational process.

It must be to mentioned that a workload distribution problem belongs to NP-hard problem. The great number of possible solutions leads to the absence of a guarantee solution or to an unacceptable result in the available time in fog-computing environments.

Moreover, a workload distribution in the fog-computing environment has some peculiarities. For instance, the set of computational nodes is relatively large.

Consequently, it is needed to reduce a search space to get a solution of a satisfaction quality. In this paper an ontological approach, which is based on ontological analysis of parallel algorithm structures and available resources, is proposed. It allows making the choice of nodes for workload distribution in a fog layer. This approach is promising for distributed CAD system performance improvement.
2. Circuit partitioning problem solving in DCAD systems

The circuit partitioning problem is considered as the first step in VLSI physical design. So, its solution characterizes the quality of a designed project. Partitioning is a technique to divide a circuit into a collection of smaller parts with specified criterion (a set of criteria) and restrictions [6].

Let us consider the problem of stand-alone block partitioning against the criterion of interblock connections. This problem comes to oriented hypergraph \( H = (X, E) \) partition with weighed nodes and weighed edges into \( m \) subschemes. It is required to split a set \( X \) of hypergraph \( H \) into the subsets \( X_1, X_2, \ldots, X_m \), such that:

\[
X_i \neq \emptyset, i \in I = \{1, 2, \ldots, m\};
\]

\[
X_k \cap X_l = \emptyset \bigcup_{k \in I} X_k = X; |X_k| = n_k; k, l \in I.
\]

Let \( S_j \) be a \( j \)-route, which connects an initial edge of hypergraph \( H \) with a finite edge. \( l_j \) - the conventional length of \( j \)-route, which is defined by the number of parts of hypergraph \( H \) wherethrough the route \( S_j \) goes.

\[
l_j = \sum_{i=1}^{m} \sum_{k=1}^{j} \alpha_{ki},
\]

where \( t = |E| \), \( E_j = \{ e_j | e_j \in S_j \} \),

\[
\alpha_{ki} = \begin{cases} 1, & \text{if } X_{k-1} \subset P_i, \text{but } \exists x \in X_k : x \notin P_i \\ 0, & \text{otherwise} \end{cases}
\]

In such a manner some fixed partitions \( f \) are characterized by a value \( \alpha_{j} = l_j, \text{max} \), which is the maximum number of hypergraph \( H \) parts wherethrough routes go from the initial edges to the finite ones.

CPP involves the searching procedure of a partitions \( f \) from the set of possible partitions \( F \) of hypergraph \( H \), when a value \( \alpha_{j} \) is minimized under the given constraints:

\[
\Phi = \min \alpha_{j} \text{ when } \begin{cases} |P| \leq B \\ C_i \leq C \end{cases},
\]

where \( B \) – the number of elements in a block; \( C \) – the number of primary outputs in a block [7].

Advanced VLSI and information technologies are the basis for distributed problems solving. Distributed CAD systems operate on the basis of parallel computing paradigm. Modern VLSI have the millions of transistors, so time for getting a solution can increase rapidly. Parallel computing can reduce the time of getting a solution in DCADs. There are some advantages of the parallel computing paradigm:

- solving the problems of high dimensionality,
- getting the solution of high quality,
- the inexpensiveness of multiprocessor systems [8,9].

The high complexity of microelectronics objects involves parallel computing as the basis of DCADs. It presupposes to reduce the time of getting a solution.

French corporation “Dassault Systems” released a new version of CAD “CATIA on the Clouds” based on the V6 PLM planform in 2010. Huawei and Siemens presented a collective decision “Manufacturing Cloud Desktop” based on Siemens NX software and Huawei FusionAccess Desktop Cloud platform in 2018 [11]. Autodesk corporation afforded a free access to some “cloud” applications: “Project Butterfly”, “Project Neon”, “Inventor Optimization”, “Fusion 360” and others [12]. Conspicuously, a lot of significant corporations of CADs moved their projects to the “clouds.”
3. **Fog-computing paradigm**

The new direction of information technologies, called the Internet of things, is characterized by a huge data volume. It forms the basis of a fog-computing. Such paradigm was proposed by Cisco in 2012. It can be considered as a cloud-computing development. Fog-computing focuses on the problems of real-time big data processing [13]. The main difference between cloud-computing and fog-computing is that fog shifts a computational problem to the edge of network. User devices are also used for data processing [14]. We can illustrate a fog-computing paradigm in the scheme (see Fig. 1). One can see that a fog layer locates between the network edge (user devices) and the cloud.

![Figure 1. A fog-computing framework.](image_url)

The main factor of fog-computing development is an opportunity to relocate a computational problem. For instance, the problem can be relocated from a cloud to the fog. Such peculiarity makes a fog-computing paradigm more attractive for a workload distribution problem solving in distributed systems.

4. **The Analysis of Parallel GA models**

The main problems of CPP solving are time consuming and big data storage. Methods, which are based on genetic algorithms (GA), are rather effective [15]. But nontrivial problems can require significant computing resources (high volume memory and along search time) while a simple GA is being executed. Parallel GA implementations is considered be more effective in many cases. It is so because a GA has a natural peculiarity to be parallelized.

Let us consider the conceptual models of parallel GA, which were presented in literature [16, 17].

**Independent Runs Model.** This model consists in executing in parallel a same sequential algorithm, with no interaction among the independent runs. Such approach can be rather useful. For example, it can be used to run several versions of the same problem with different initial conditions, thus allowing statistics on the problem to be gathered. An independent runs model can be considered as a special case of the master-slave model, where there is no migration at all.

**Master-Slave Model.** The main idea of this model involves distributing the objective function evaluations among several slave processors while the main loop of the GA is executed in a master processor. The master processor sends parameters (those necessary for the objective function evaluations) to the slaves; objective function values are then returned when computed. The master processor controls the parallelization of the objective function evaluation tasks performed by the slaves. This model is effective in the case of the high complexity of objective function.

**Distributed Model.** In this model, the population is structured into smaller subpopulations relatively isolated one from the others. Parallel GA based on this paradigm are sometimes called multipopulation or multi-deme GAs. The key characteristic of this kind of algorithm is that individuals within a particular subpopulation (or island) can occasionally migrate to another one. Here, genetic operators (selection, mutation, and recombination) take place within each island, which means that each island can search in very different regions of the whole search space with respect to the others.
Cellular Model. The parallel cellular (or diffusion) GA paradigm normally deals with a single conceptual population, where each processor holds just a few individuals. The main peculiarity of cellular model is the structuring of the population into neighborhoods, and individuals may only interact with their neighbors. Thus, since good solutions (possibly) arise in different areas of the overall topology, they are slowly spread (or diffused) throughout the whole structure (population).

5. Efficiency enhancement method for distributed CAD systems in fog-computing environments

In this work, CPP solving efficiency enhancement is the time reduction of workload distribution problem solving. Ontological approach to solve this problem is proposed here. A new method involves ontological analysis, which tends to limit the search space of workload distribution.

The initial data of parallel algorithm is as follows:
- an algorithm class (genetic algorithm, bio-inspired algorithm, algorithm inspired of the lifeless nature);
- a parallel model (independent runs, master-slave, distributed, cellular);
- the description of a subgraph, which has to be relocated (initial placement and partitioning method);
- node characteristics (computation capacity and channel capacity).

We need to impose some restrictions of graph partition methods to avoid NP-hard problem solving. Then on the basis of ontological analysis it is possible to reduce the search space of candidate nodes for a workload distribution problem. The scheme of proposed method is shown below (see Fig. 2).

![Figure 2. The scheme of developed method.](image)

6. Ontology Development

As it was mentioned above, the distributed CAD performance in fog-computing environments is considered as the main criterion in the current work. This criterion depends on some factors including search optimization methods, parallelized algorithm methods and ways of algorithm subtask distribution between nodes. If a computational problem has to be relocated, a computational process is paused. Therefore, the speed and quality of workload distribution affects the DCADs problem solving.

As far as the number of fog-layer nodes is big, it takes a lot of time to consider all of them. In this work the search space reduction is proposed. It’s necessary to choose a set of nodes which is appropriate for computational problem solving. For this purpose an ontological model is proposed. The model allows
representing information about structure and characteristics of subtask, which has to be relocated, formally. On the basis of ontological model can be decided if the candidate node should be included into the search space or not.

Let us conduct the analysis of subtask, which has to be relocated, according to its graph structure to specify a set of characteristics.

We use Protégé 4.2 for ontology development [18]. In terms of ontology algorithm class for CPP solving is defined as class “Class”, which is the class of a superclass “Thing”. In this work the class “Class” includes the following subclasses. They are “GA”, ”Bio-inspired algorithm”, “Lifeless nature algorithm”. The number of these subclasses can be extended. It depends on algorithm, which was chosen for CPP solving.

In this work subclass “GA” is considered. The next class includes GA parallel models (“Parallel_models”). The “Parallel_models” class is connected with the subclass “GA” through the “Object Propeties” – “has_parallel_models”. The class “Parallel_models” includes the following subclasses “Independent_runs”, “Master_Slave”, “Distributed”, “Cellular”.

To represent the subgraph of workload distribution problem it is necessary to create the classes of initial subgraph placement (“Initial_placement”) and graph partitioning methods (“Partitioning_method”). Moreover, each parallel model has a unique partitioning method. For example, the “Independent_runs” model has a partitioning method (“Indeped_runs_partitioning”), which includes two different ways. The first one is algorithm fragment relocation (“Instance_transf”). The second one is algorithm fragment partitioning (“Instance_partitioning”). There are two partitioning methods for the “Master_Slave” model: master relocation (“Master_transfer”), slave relocation (“Slave_transfer”) or master and slave relocation (“M_S_transfer”). There is only one way of partitioning possible in case of the distributed model – island relocation (“Island_transfer”). In case of the cellular model a procedure of graph partitioning is not sensible because it leads to distributed (island) model. It’s necessary to create a class “Node_features” for the description of node characteristics. The developed ontology is shown below (see Fig. 3).

![Figure 3. The developed ontology.](image)

Ontology, which is presented above, is a key component of proposed method. The ontology includes the formal description of parallel GA structure and their characteristics. It also involves required and available resources. The developed ontology makes possible to limit the search space of candidate nodes for workload distribution problem solving. It’s worth noting that ontology can be extended by adding the new classes of algorithms.
7. Conclusion
The issue of efficiency enhancement for distributed CAD systems in fog-computing environments was examined in the current paper. As a circuit partitioning problem belongs to NP-hard problems, the time needed for solving can be long enough, so it affects the computational problem. In the view of fog-computing paradigm the problem of workload distribution is particularly topical. The method, which is focused on the limitation of candidate search space, was proposed for this problem solving. Ontological analysis is a key component of this method. It allows us to reduce a set of nodes, which is considered for distribution. As a result, inappropriate nodes are eliminated from consideration. We have a limited set of nodes for workload distribution. In that way we reduce the time of getting a solution of the workload distribution problem. The analysis of parallel GA models was conducted in this work. The results were the basis for ontology development.

8. Acknowledgments
The reported study was funded by the Russian Foundation for Basic Research No. 18-07-00050, No. 18-29-22019/18.

References
[1] Glushan V M, Lavrik P V 2015 Distributed CAD systems. Architecture and possibilities. (Oskol: TNT)
[2] Gridin V N, Dmitrevich G D, Anisimov D A 2012 Construction of web - services of systems of automation circuitry designing Journal of Information Technologies and Computing Systems 4 79-84
[3] Dmitrevich G D, Anisimov D A 2014 Computer-aided design systems engineering based on service-oriented architecture ETU "LETI" 2 14-18
[4] Moysiadis V, Sarigiannidis P, Moscholios I 2018 Towards Distributed Data Management in Fog Computing Wireless Communications and Mobile Computing 2018 1-14
[5] Melnik E V, Klimenko A B, Ivanov D Y 2018 Distributed Information and Control system reliability enhancement by fog-computing concept application Proceedings of IOP Conference Series: Materials Science and Engineering (vol. 327) pp.165 – 168
[6] Kurejchik V M 1990 The mathematical support of design via CAD systems (Moscow: Radio i svyaz')
[7] Kurejchik V V, Kurejchik V M 2006 The evolutionary methods of commutation circuit partitioning problem (Taganrog: TRTU)
[8] Korneev V V 1999 Parallel computing system (Moscow: Nolidzh)
[9] Lavrik P V 2010 Conceptual approaches of parallel processing of the information in the context of realization distributed CAD Proceedings of the Congress on intelligent systems and information technologies "IS&IT'11" (vol. 3) (Moscow: Fizmatlit) pp. 112-118
[10] Levin D CAD’s russian market at the end of 2014 2014 Your window to the CAD’s world Retrieved from: http://isicad.ru/ru/digest/isicad.ru_articles_8_2014.pdf
[11] The announcement of a contemporary solution 2018 Tadviser Retrieved from: http://www.tadviser.ru/index.php/Продукт:Huawei_u_Siemens_Manufacturing_Cloud_Desktop
[12] Stremnev A 2018 Autodesk Fusion 360 – 3D is under control Retrieved from: https://sapr.ru/article/25279
[13] Stojmenovic I, Wen S. 2014 The Fog Computing Paradigm: Scenarios and Security Issues Proc. 2014 Fed. Conf. Comput. Sci. Inf. Syst. (vol. 2) ed. Maria Ganzha, Leszek A. Maciaszek, Marcin Paprzycki (New York City: IEEE) pp.1–8
[14] Vaquero L M, Rodero-merino L 2014 Finding your Way in the Fog: Towards a Comprehensive Definition of Fog Computing ACM SIGCOMM Comput. Commun. Rev. 44 (5) 27–32
[15] Gladkov L A, Kurejchik V V, Kurejchik V M 2006 Genetic algorithms (Moscow: FIZMATLIT)
[16] Alba E 2005 Parallel metaheuristics. A New Class of Algorithms. (New York: Wiley)
[17] Levine D A 1996 A Parallel Genetic Algorithm for the Set Partitioning Problem Meta-Heuristics ed. Osman I.H., Kelly J.P. (Boston: Springer) pp. 23-35
[18] Protégé 4.2 Retrieved from:https://protege.stanford.edu