

AHeuristic Method for Arrangement in Heterogeneous Disseminated Embedded Systems

A. R. Aravind, Assistant Professor,
Prince Shri Venkateshwara Padmavathy Engineering College, Ponmar Chennai-600127
Corresponding authors

Abstract:

This paper presents a heuristic approach for workflow scheduling in heterogeneous distributed embedded system (HDES). A genetic algorithm (GA) and ant colony optimization (ACO) modified with the greedy algorithm introduced to the system contains multiple heterogeneous embedded machines (HEMs) working as a cluster. Users can remotely access and utilize their computational power. The communications on different types of buses are taken into account to find an optimal solution. New meta-heuristic information based on forwarding dependency is proposed to build probability for ACO to generate task priorities. Besides, a greedy algorithm for machine allocation is incorporated to complete task scheduling. Experiments based on random task graphs running in the HEM cluster demonstrate the effectiveness of the modified greedy ant colony optimization algorithm which outperforms the others by 33% more result quality.

Keywords: Heterogeneous distributed embedded system, Network cluster, Meta-heuristic task scheduling, Genetic algorithm, Ant colony optimization, Greedy algorithm.

1. Introduction

The evolution and growth of the heterogeneous systems used for high-performance computing have been surprising for many years. Generally, heterogeneous systems [1] for parallel and distributed computing, consist of the system software, multiple processors, memories, and a communication network. Nowadays, more than 60 percent of electronic control unit parts in both automotive electronics and avionics systems employ the heterogeneous distributed embedded system (HDES) [2] because of the benefits in the resource utilization, weight, scale, and power consumption and high-performance. In the HDES architecture, the specialized processing elements are used to accelerate the complex tasks, also coprocessors for general purpose computing cooperated with single or multicore processors are exploited for tasks require flexibilities

Data-dominated applications in a variety of fields, such as computational, engineering, and scientific, are described by a DAG (directed acyclic graph). Nodes represent tasks and edges represent communication messages between tasks in the DAG. The scheduling tasks on HDES processors [3] to minimize the schedule length of a DAG-based parallel scientific application is a well-known nondeterministic polynomial-hard (NP-hard) optimization problem implement and orchestrate tasks in HDES, generally, message passing interface (MPI) is employed to make full use of distributed clusters and ensure high scalability. There are many application programming interfaces (APIs) or high-level programming languages used for MPI [3] such as open computing language (OpenCL), open multi- processing (OpenMP), and coprocessing threads (COPRTHR) SDK from Brown Deer Technology

2. Background and related work

In this paper, there are two study parts conducted for related work consisting of the embedded heterogeneous system (EHS) architectures and meta-heuristic applications on heterogeneous systems. The heterogeneous computing platform (HCP) for data processing in [3] presents the heterogeneous processors and partial reconfigurable hardware accelerators work together with efficiency through APIs consist of COPRTHR SDK and Epiphany SDK on Parallela board. The matrix-vector multiplication in a different size is the benchmark application for HCP, the results show that when data is increasing, the heterogeneous processor and a hardware accelerator are the best performance on the platform.

The work in [5] implements edge detection on heterogeneous system architecture (HSA) environment to compare an ARM multicore processor, Coprocessor and FPGA, and x64 architecture in terms of energy efficiency and optimal performance evaluated from image processing application. The ARM multicore processor with coprocessors (Epiphany multicore processor + FPGA) provides 50 times more energy efficiency better than an x64 architecture in experimental results.

Another interesting use of heterogeneous systems is related to clustering systems. The research in [4] presents a large-scale parallel method of moments on CPU/MIC heterogeneous clusters that proposes hybrid MPI/OpenMP APIs management for distributed CPU/MIC heterogeneous platforms. An implementation of a single board computer cluster for the crypto algorithm is presented in [6] to study 64-board. The cluster architecture is the Raspberry Pi 3 Model B to problem-solving of implementation of the DOZEN crypto algorithm. Towards a dataflow runtime environment for Edge, Fog, and In-Situ computing based on distributed Sucuri architecture [7], also presents the implement on multicore embedded

The hybrid heuristic-genetic algorithm with adaptive parameters for static task scheduling in a heterogeneous computing system is presented in [8] to further improve the performance of existing algorithms. In [9], a greedy ant algorithm is used for workflow scheduling for heterogeneous computing. This shows a novel workflow scheduling algorithm named Greedy-Ant to minimize the total execution time of an application in heterogeneous environments. The well-known genetic algorithm is employed for efficient parallel execution on Epiphany manycore processors. The work in [10] shows an evaluate Parallel a which is a small board with the Epiphany coprocessors consisting of sixteen MIMD cores connected by a mesh network-on-a-chip. The results are based on classical genetic algorithms.

The scheduling tasks on HDES processors [11] to minimize the schedule length of a DAG-based parallel scientific application are a well-known nondeterministic polynomial-hard (NP-hard) optimization problem, and numerous heuristic list scheduling algorithms have been proposed to generate near-optimal solutions of the problem.

In this paper, the motivation is to implement the meta-heuristic algorithm to workflow scheduling in the cluster-based HDES environment with heterogeneous embedded machines. We conduct the evaluation and implementation of the genetic algorithm (GA), the ant colony optimization algorithm (ACO), the modified greedy genetic algorithm. (MGGA), the modified

greedy ant colony optimization algorithm (MGACO), and genetic simulated annealing (GSA) on the proposed HDES system architecture. Rather comparing with conventional algorithms such as GA and ACO, the modified heuristic algorithm such as GSA are also included. The assumptions of HDES including task execution and communication models based on the architecture are described

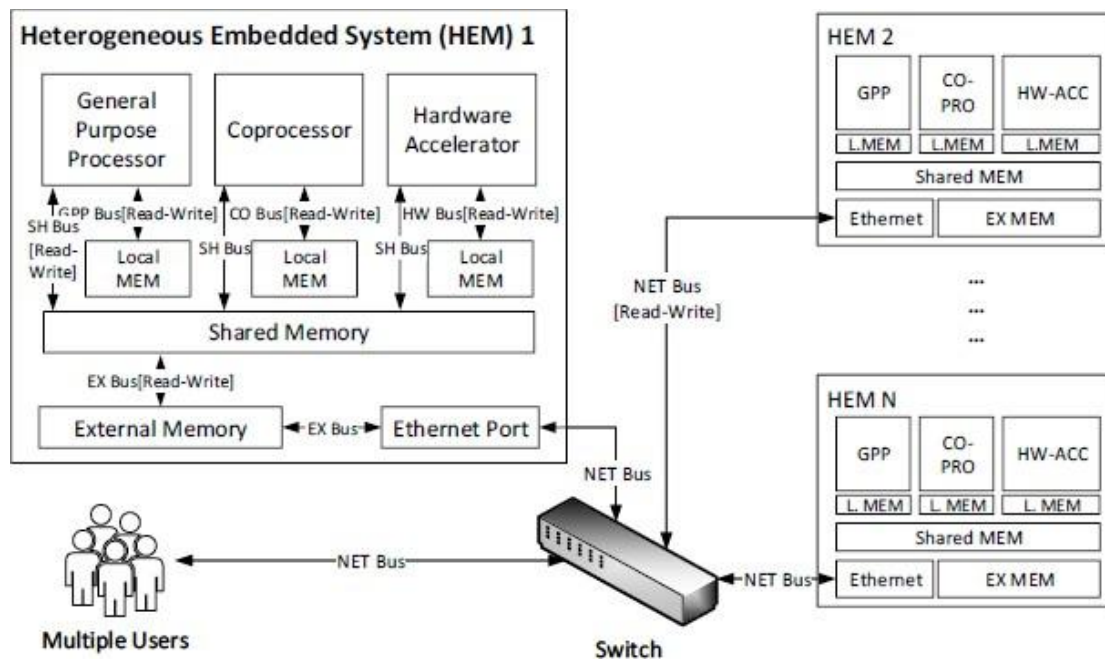


Figure.1 A cluster of heterogeneous embedded machine

3. System design overview

This section presents an overview of the system that includes the proposed HDES architecture, design, and heuristic applications that will be implemented and run in the system. HDES architectural design is based on computer architecture principles or embedded system principles [12-14]. We propose HDES working as a cluster that includes N -heterogeneous embedded machines (HEMs). Each embedded machine contains multiple processing elements such as processors, coprocessors, hardware accelerators also memories, and interconnections. They connect to the network devices for multiple-user services. Rationally, the system performance can be limited by the speed of the network. For this reason, it makes sense to give only parallel processes access to the HDES cluster connection network.

Generally, there are 4 layers involved in implementing applications in heterogeneous systems, namely an application layer, an API layer, an operating system (OS) layer, and the hardware layer. The proposed heterogeneous cluster is shown in Fig.1. In this research, the application layer represents data-dominated applications. The API layer is the set of functions or libraries that are available for the users to manage system resources and run applications. For example, the OpenMP is designed for shared memory, which can run on a HEM cluster using both OpenMP and MPI designed for distributed memory [13-17] such as local memory, external memory, and bus system. There are many buses used as the communication paths between

processing units in this system such as the processor bus (GPP bus), coprocessor bus (CO bus), hardware bus (HW bus), external bus (EX bus), and network bus (NET bus) as shown in Fig. 1.

4. Proposed heuristic algorithms based on the HDESarchitecture

We introduce four meta-heuristic algorithms for scheduling problem based on the proposed HDES architecture. The algorithms include genetic algorithm (GA), ant colony optimization algorithm (ACO), modified greedy genetic algorithm (MGGA), and modified greedy ant colony optimization algorithm (MGACO). The second step, an initial population is generated randomly as a sequence of integers inside a chromosome representing the class indices in one of the matched DAG.

The crossover occurs between the two parents. We use a simple technique, random one-point crossover. A random rate is 25 percent. The crossover operator builds an offspring by random choosing to cut-points in the two parents, copying the subsequences between the cut points in the two parents into new two offspring, one each, and then the remaining indices are filled, position wise, from the otherparent.

The modified greedy genetic algorithm (MGGA) is shown on the left side of Fig. 2, to a matching problem in DAGs. As it can be seen that MGGA is similar to the GA, but one greedy step is added on.

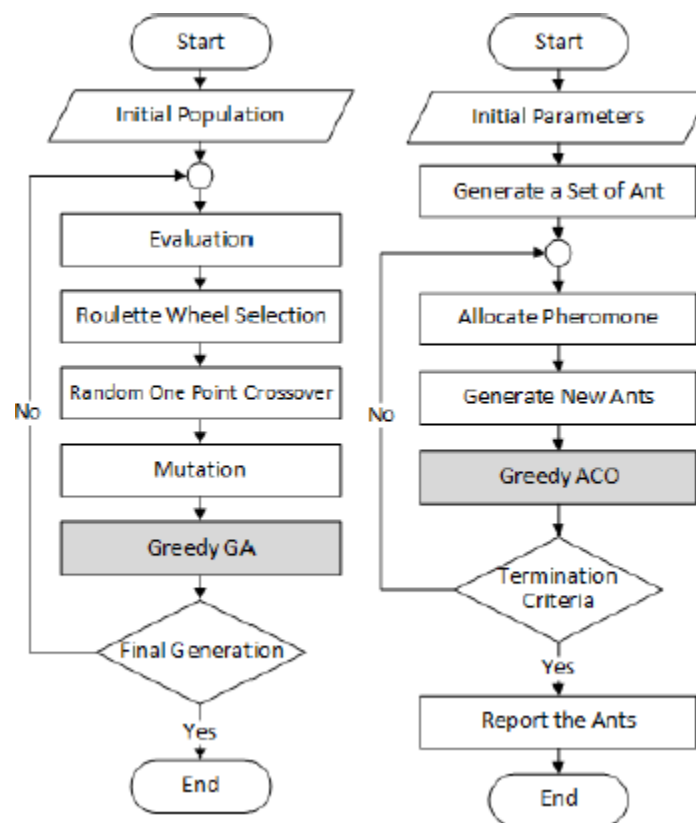


Figure.2 The flow chart of proposed MGGA (left) and MGACO (right)

5. Experiments and results and Discussion

In this section, experiments are carried out with the comparison of the results using the heuristic algorithms to find optimal solutions for HDES. The experimental design of HDES that includes the 9- layer of HEM connected by a network switch based on a star topology. The algorithms with different task graphs are simulated in this heterogeneous embedded system. With MPI, the meta-heuristic algorithms can be managed to run on selected HEM using master-slave technique. The results from different sizes of task graphs in this heterogeneous embedded system types of algorithms are collected at the HEM master. The best solutions are averaged and compared using makespan, speedup, and SLR metrics.

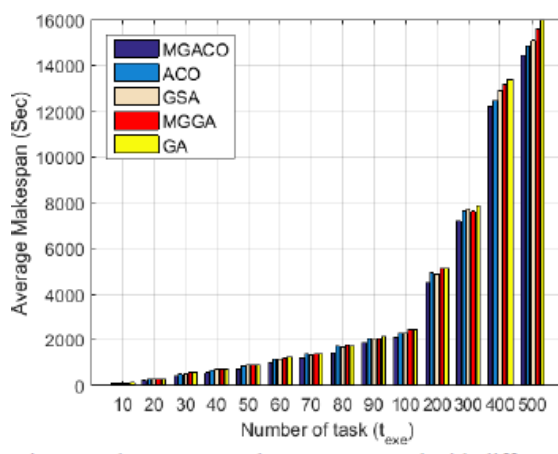


Figure.3 The average makes pan compared with different algorithms on random task graphs.

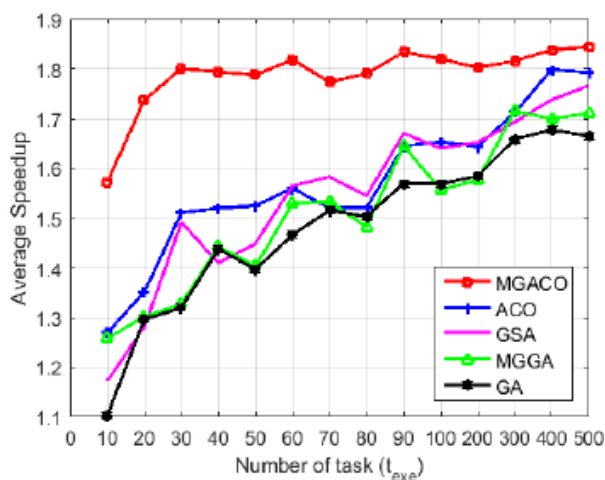


Figure.4 the average speedup compared with different algorithms on random task graphs.

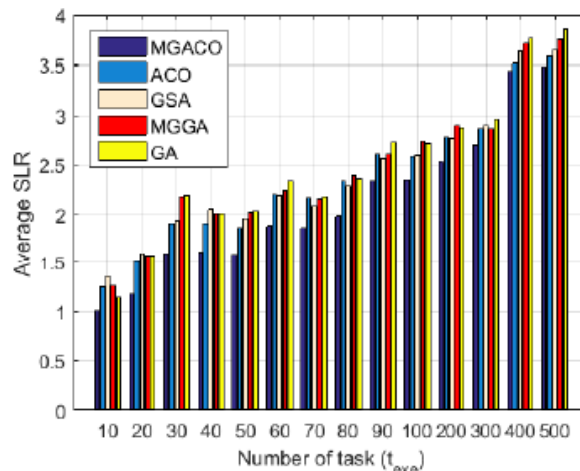


Figure.5 the average SLR compared with different algorithms on random task graphs

Fig. 3 shows the average makespan from all DAG graph sizes obtained from GA, ACO, MGGA, MGACO, and genetic simulated annealing (GSA). The GSA algorithm [18] utilizes the simulated annealing (SA) algorithm in the part of mutation operation based on GA for finding the optimization value. The average makespan of MGACO is 240 minutes which is the best of overall average makespan while ACO is 247 minutes, GSA is 251 minutes, MGGA is 259 minutes, and GA is 266 minutes, respectively.

In Fig. 4, the proposed MGACO performs better than other algorithms. Compared with GA and GSA, MGACO improves speedup by 30 percent for a small graph size. When the task number increases, MGACO shows dramatically in advantage. It is found that the performance of MGACO tends to converge rapidly at the steady state, when the number of tasks is larger than 30. Although, ACO performs better than GSA, MGGA, and GA, when the task number exceeds 50, its performance decreases afterwards. When number of tasks is equal to 80, MGACO improves speedup about 33 percent. For the SLR, Fig. 5 shows the details of the comparisons on random tasks graphs. It is obvious

By introducing a simple greedy algorithm in meta-heuristic searching for a shortest makespan in this paper, MGACO can generate better scheduling results compared to GA and MGGA because an ant colony can help to find the best pathway. As the complexity of the task graph grows, the task graph becomes harder for them to produce consistent results on a variety of graphs. The resource constraints are addressed in the cost function since the number of reconfigurable cells and coprocessors are limited.

6. Conclusion

This paper considers the genetic algorithm and the ant colony optimization modified with the greedy algorithm and presents an approach for ACO based workflow scheduling. New meta-heuristic information based on forward dependency is proposed to build probability for ACO to generate task priorities. Additionally, a greedy algorithm for machine allocation is incorporated to complete scheduling. Based on the random task graphs, experimental results in the HEM

cluster demonstrate the effectiveness of the modification of greedy ant colony optimization algorithm which outperforms the others by 33% more result quality. Future work would be a real implementation of data-dominated applications in this HDES based on the optimal scheduling results.

References

1. S. Prongnuch and T. Wiangtong, "Heterogeneous Computing Platform for data processing", In: *Proc. ISPACS*, pp.1-4,2016.
2. Y. Chen, S. Zuo, Y. Zhang, X. Zhao, and H. Zhang, "Large-Scale Parallel Method of Moments on CPU/MIC Heterogeneous Clusters", *IEEE Trans. on Antennas and Propagation*, Vol.65, No.7, pp.3782-3787, 2017.
3. S. Prongnuch and T. Wiangtong, "The Implementation of Edge Detection on HSA Environment", In: *Proc. iEECON*, pp.1-4,2017.
4. S. N. Agathos and V. V. Dimakopoulos, "Adaptive OpenMP Runtime System for Embedded Multicores", In: *Proc. BUC*, pp.174-181, 2014.
5. S. Ding, J. Wu, G. Xie, and G. Zeng, "A Hybrid Heuristic-Genetic Algorithm with Adaptive Parameters for Static Task Scheduling in Heterogeneous Computing System", In: *Proc. IEEE Trustcom/ BigDataSE/ICSS*, pp.761-766, 2017.
6. B. Xiang, B. Zhang, and L. Zhang, "Greedy- Ant: Ant Colony System-Inspired Workflow Scheduling for Heterogeneous Computing", *IEEE Access*, Vol. 5, pp.11404-11412,2017.
7. L. Faber and K. Boryczko, "Efficient Parallel Execution of Genetic Algorithms on Epiphany Manycore Processor", In: *Proc. FedCSIS*, pp.865-872, 2016.
8. G. Xie, G. Zeng, X. Xiao, R. Li and K. Li, "Energy-Efficient Scheduling Algorithms for Real-Time Parallel Applications on Heterogeneous Distributed Embedded Systems", *IEEE Trans. on Parallel and Distributed Systems*, Vol.28, No.12, pp.3426- 3442, 2017.
9. J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, Vol.6, Morgan Kaufmann, San Francisco,C.A.2017.
10. P. Czarnul, *Parallel Programming for Modern High Performance Computing Systems*, Vol.1, Chapman and Hall/CRC, New York,N.Y.2018.
11. L. Howes, *TheOpenCL Specification, Version: 2.1, KhronosOpenCL Working Group*, [Online] Available: <https://www.khronos.org/registry/OpenCL/specs/opencl-2.1.pdf>. Accessed on: Sep. 2, 2011.
12. Brown Deer Technology, *COPRTHR-2 Overview*, [Online]. Available: <http://www.browndeertechnology.com/docs/COPRTHR-2 Overview rev-b-20160629.pdf>. Accessed on: Sep. 2, 2016.
13. Adapteva, *Parallella-1.x Reference Manual*, [Online]. Available:https://www.parallella.org/docs/parallella_manual.pdf. Accessed on: Sep.2, 2010.
14. O. Terzo, K. Djemame, A. Scionti, and C. Pezuela, *Heterogeneous Computing Architectures: Challenges and Vision*, Vol.1, CRC Press, Florida,F.L.2011.
15. M. Zhao, X. Yin, and H. Yue, "Genetic Simulated Annealing-Based Kernel Vector Quantization Algorithm", *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.31, No.5, pp. 1-28,2017.
16. L. C. Thanh, "Performance Analysis of Greedy Algorithms for Max-IS and Min-Max1-Match", *Vietnam Journal of Mathematics*, Vol.36, No.3, pp.327-336, 2008.