A Path Clustering Heuristic for Scheduling Task Graphs onto a Grid

L. F. Bittencourt, E. R. M. Madeira, F. R. L. Cicerre, L. E. Buzato
{luiz.bittencourt,edmundo,fcicerre,buzato}@ic.unicamp.br
Institute of Computing, State University of Campinas
PO 6176, Campinas, São Paulo, Brazil

1. INTRODUCTION

Task scheduling is an NP-Complete problem and efficient scheduling is very important for achieving good performance. We propose a heuristic for scheduling task graphs onto a grid infrastructure that supports dependent task execution. The baseline of the *Path Clustering Heuristic* (PCH) algorithm is to select a path from the DAG and schedule the nodes on this path onto the same processor.

The proposed algorithm was developed based on Xavantes grid middleware and its programming model [1]. The middleware arranges the resources in groups, aiming to execute dependent tasks in near resources, reducing the communication costs and increasing the performance on process execution. In Xavantes, *Controllers* are control elements that specify the tasks execution order and they are also responsible for transmitting data between dependent tasks. Controllers are specified in the programming model and each task is subordinated to a controller.

2. PROPOSED ALGORITHM

The PCH algorithm uses some attributes calculated for each task: *Priority*, *Weight* (Computation Cost), *Communication Cost*, *Earliest Start Time* (EST) and *Estimated Finish Time* (EFT). All the information necessary to compute these attributes is given by the programming model or by the infrastructure. Initially, we assume a virtual homogeneous system composed of an unbounded number of the best processor available connected by links with the highest bandwidth available. Each task is scheduled on a different processor on the virtual system, then the algorithm computes the initial attribute values of each node.

In the task selection and clustering step, the node n_i with the highest Priority is selected and added to the cluster cls_k. Next, the algorithm performs a depth-first search on the task graph starting on n_i, always selecting the not scheduled successor n_s that has the highest $P_s + EST_s$ and adding it to the cluster cls_k, until n_s has no succesors.

To select a resource to a cluster, the EFT of each node in the cluster is calculated and the EST of the successor of the last node in the cluster is calculated. A cluster cls_k is scheduled on the resource that gives the smallest EST for the successor of cls_k. Finally, the Weights and ESTs are recomputed. The task selection and the processor selection steps are repeated while there are unscheduled nodes.

With all nodes scheduled, each controller must be assigned to a processor that minimizes the communication with its nodes and with its subcontrollers. The policy of creating clusters based on sequential tasks in the task graph helps in reducing the controller communication overhead. The controller selected to be scheduled is a random choice among that which have all subcontrollers already scheduled. The resource selected to suit a controller is that which minimizes the communication between the controller and its tasks, and the controller and its subcontrollers.

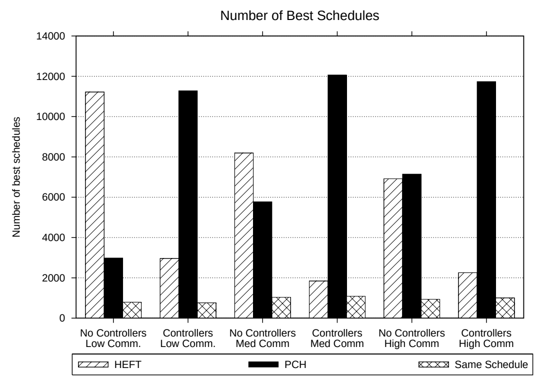

Figure 1: **Number of best schedules of each algorithm.**

We compared PCH algorithm with HEFT algorithm [2]. The results in Figure 1 show that PCH gives good results when controllers are considered, as expected and desired.

3. CONCLUSION

We present a heuristic for task scheduling in the Xavantes grid middleware. For the task graphs supported by Xavantes, the strategy gives good performance when controllers are considered with time complexity $O(pv^3)$.

Future works include making the algorithm schedules only part of the DAG in advance, then scheduling the remaining nodes in real time. Here, how many nodes the algorithm is supposed to schedule in advance is the question that comes up. Also, rescheduling nodes that are in resources with poor performance can minimize the running time.

The authors thank CNPq, and WebMaps and Agroflow projects for the financial support.

4. REFERENCES

[1] F. R. L. Cicerre, E. R. M. Madeira, and L. E. Buzato. A hierarchical process execution support for grid computing. *To appear in Concurrency and Computation: Practice and Experience*, 2005.

[2] H. Topcuoglu, S. Hariri, and M.-Y. Wu. Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Trans. Parallel Distrib. Syst.*, 13(3):260–274, 2002.