Optimizing End-to-End Performance of Distributed Workflows in a Heterogeneous Environment using resource sharing

Eti Aggarwal, Dr. Joshua Philips, Dr. Yi Gu

College of Basic and Applied Sciences, Middle Tennessee State University

ABSTRACT

Next-generation computation-intensive applications in various domains feature large-scale computing workflows that are often modeled as directed acyclic graphs and deployed in distributed network environments. With limited computing resources, it is often unavoidable to map multiple modules to the same computer node which results in a resource contention among concurrently executing independent modules. Considering the implications of such resource sharing in workflow mapping algorithms is crucial for improving the end-to-end performance of these distributed applications. We conduct a deep investigation into workflow execution dynamics under a fair share-based resource-sharing system and propose Recursive Critical Path with Fair Share-RCPS algorithm. We will compare its performance with the existing RCP algorithm by using an extensive set of simulated data and comparing the resulting minimum end-to-end MED values. RCPS will provide a lower value of MED as compared to RCP by including the exact resource sharing dynamics in workflow mapping.

1. BACKGROUND

• Applications processing huge amounts of data that varies in volume, velocity, variety and veracity (also known as Big Data) are gaining an industry-wide importance. [1, 2]

• Such applications are considered as a workflow of interdependent computing modules represented as a directed acyclic graph and mapped on a heterogeneous network. [3, 4, 5]

• The objective of such mapping is to select a set of computing nodes in the physical computer network and assign each computing module in the virtual task graph to one of the selected nodes to achieve minimum end-to-end delay (MED).

• The Efficient Linear pipeline configuration- ELPC algorithm is a dynamic programming based approach which models the computing workflow as a linear pipeline, partitions the modules of the task graph into groups and maps them onto a set of selected computing nodes in a network. [6] (Fig. 4)

• The recursive critical path-RCPS algorithm improves upon ELPC algorithm by considering the task graph as a directed acyclic graph instead of a pipeline, which gives more realistic representation. [7] (Fig. 1, Fig. 5)

• When multiple modules get mapped to a common node they can compete for the processing power and bandwidth of the communication link. (Fig. 2)

• Including the implications of such resource sharing in the mapping algorithm can improve the MED value obtained from the mapping. (Fig. 3)

2. METHODS

• We utilized the task and network graphs used in the RCP algorithm and generated simulation datasets by varying the number of modules, module complexity, network nodes, node processing power and network bandwidth within nodes.

• We implemented the Fair share based RCPS algorithm to calculate the exact amount of processing power and bandwidth received by a module when mapped on a node with multiple other modules.

• We modified the RCP algorithm with above calculations for the mapping of the non-critical modules on heterogeneous network.

• We will run the modified datasets over the RCP and RCPS algorithms and compare the MED values. (Fig. 6)

3. RESULTS

• We anticipate that the RCPS algorithm will provide an improved value of MED over the RCP algorithm for cases when resource sharing occurs. (Fig. 6)

• RCPS will provide a substantial performance speedup over the RCP algorithm. (Fig. 7)

• Small standard deviation values in the MED demonstrates the performance robustness and optimization stability of RCPS in achieving MED in various task graphs and computer networks of disparate topologies and different scales. (Fig. 8)

4. CONCLUSIONS AND FUTURE WORK

• Performance analysis results will prove that RCPS algorithm provides an improved MED value over RCP algorithm for resource sharing scenarios. (Fig. 6, Fig. 7)

• The future work will focus on designing a priority based dynamic resource sharing algorithm that will give more resources to critical modules over branch modules. [9]

REFERENCES

ACKNOWLEDGEMENTS

This research is sponsored by National Science Foundation under Grant No. 536958 with Middle Tennessee State University.