Analysis of Meta-level Communication for Distributed Resource Allocation Problems

(Extended Abstract)

Matthew Saponaro
Computer and Information Sciences
University of Delaware
Newark, Delaware
mattsap@udel.edu

Keith Decker
Computer and Information Sciences
University of Delaware
Newark, Delaware
decker@udel.edu

ABSTRACT

We examine the effects of meta-level-communication in the DSRAP (Distributed Sequential Resource Allocation Problem). In DSRAP, independent tasks are categorized into different types, where each task belonging to a particular task type shares a known distribution of task arrivals, durations, reward rates, maximum waiting times, and resource demands. We first look at a single task type DSRAP (SDSRAP) and develop an analytical model of the effect of meta-level communication about load on system performance for first-in-first-out (FIFO) local scheduling agents that forward tasks based on load. Through our analytical models and empirical results, we show how the frequency of communication affects performance for SDSRAP problems with one resource and task type. We then quantitatively measure the impact of meta-level communication on system performance with respect to the global measures of the system's load balance. We validate our analytical model's predictions experimentally, showing, e.g., as system load becomes unbalanced, performance decreases; organizational structure significantly impacts agent performance; and agents that can communicate and distribute tasks to neighbors significantly outperform agents working individually. Through our analysis on FIFO, routing algorithms, and our policy agents, we provide a framework for analyzing more complex task schedulers and task routing algorithms.

1. INTRODUCTION

Warehouse shipping, cloud computing clusters, and distributed sensor networks—all are large-scale systems requiring disparate entities working together to achieve goals. Participating entities are usually physically distributed and excessive communication is not always ideal due to overhead, network message saturation, and inefficient utilization of resources. Zhang et. al defines and provides several solutions to the DSRAP, but does not explore the problem analytically [7, 6, 8]. In order to optimize the amount of communication, we must understand how communication affects system performance. We provide a method that allows for organizations to describe system characteristics without spending resources testing system configurations by mathematically modeling system behavior. Our DSRAP model is comprised of a local task allocation mechanism and a task routing mechanism. We provide a general framework that allows for each component to be modeled individually. We analyze FIFO task allocation with both a meta-level communication router and then an evenly-distributed router. This paper's analyzed routing algorithms extend Arpaci-Dusseau and Culler's centralized proportional routing algorithm by combining it with meta-level communication to make a decentralized solution [1]. Meta-level communication has been extensively studied and is efficient in the sense that it summarizes necessary state information [2, 4]. Our agents communicate the number of tasks (load) with adjacent agents.

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

2. ANALYTICAL MODEL

We provide a set of difference equations to analytically describe the SDSRAP. The following equations describe the number of waiting, w_i^t, allocated, x_i^t, and failed, F_i^t, tasks at a given time t, and the number of available resources, R_i^t, for an arbitrary agent, A_i.

$$w_i^t = w_i^{t-1} - x_i^{t-1} - F_i^{t-1} + E_i + \sum_{j \in \text{adjacent}} S_{j\to i}^{t-d_{ij}} - S_{i\to j}^{t-1} \quad (1)$$

where w_i^{t-1} is the number of tasks that were previously waiting, x_i^{t-1} is the number of previously allocated tasks, F_i^{t-1} is the number of failed tasks from the previous time unit, $S_{j\to i}^{t-d_{ij}}$ is the number of tasks that A_j sent to A_i that have just arrived at A_i, $S_{i\to j}^{t-1}$ is the number of tasks A_i just previously sent away to A_j, and E_i is the number of new tasks that have arrived from the environment. At the beginning of an episode there are no tasks waiting—that is, $w_i^0 = 0$.

We represent the number of tasks allocated at time t, x_i^t as the sum total of tasks allocated at time t that arrived at some earlier time $a \le t$ ($x_i^t[a]$). We can also represent $S_{i\to j}^t$ and w_i^t that are described below similarly.

$$x_i^t = \sum_{a=1}^{t} x_i^t[a] \quad (2)$$

A FIFO agent will allocate tasks that arrive earlier first. The number of allocated tasks from a particular time is constrained by the number of waiting tasks from that time, $w_i^t[a]$, and the number of resources still remaining after serving tasks that arrived before time a, $R_i^t[a]$.

$$x_i^t[a] = \min\left(w_i^t[a], R_i^t[a]\right) \quad (3)$$

$$R_i^t[a] = \max\left(R_i^t - \sum_{k=1}^{a-1} w_i^t[k], 0\right) \quad (4)$$

An agent will have as many resources as it had in the previous time unit that have not been allocated in addition to the resources freed from completed tasks, x_i^{t-s}, where s is the service time of a task:

$$R_i^t = R_i^{t-1} - x_i^t + x_i^{t-s} \quad (5)$$

We describe the number of failed tasks at a particular time, t, as the number of tasks that have exceeded their maximum waiting time, m. A task has exceeded its maximum waiting time if it arrived m time units before t, and has not been serviced. Therefore, the number of failed tasks at time t, F_i^t, is $w_i^t[t-m]$.

We use Hamilton's apportionment algorithm in order to route tasks [3]. Essentially, the number of sent tasks, $S_{i\to j}^t$, that each neighboring agent, A_i, will be sent is a proportional whole value, $\lfloor p_{j\to i}^t \rfloor$; additionally, any remaining tasks, $O_{j\to i}^t[a]$ will be divided to the agents with highest fractional component of p.

For evenly-routing agents, we describe $p_{j\to i}^t[a]$ as the proportion of excess tasks that should be sent to a neighboring agent based on the number of neighbors.

$$p_{j\to i}^t[a] = \frac{1}{|\text{adjacent}|} * O_j^t[a] \quad (6)$$

For proportional-routing MLC agents, we describe $p_{j\to i}^t[a]$ as the proportion of excess tasks that should be sent to a neighboring agent based on its beliefs, B_{ij}^t, about its current availability of resources.

$$p_{j\to i}^t[a] = \frac{B_{ji}^t}{\sum_{l\in\text{adjacent}} B_{jl}^t} * O_j^t[a] \quad (7)$$

Agent A_j will update A_i's belief about the resources A_j has when the difference between its current available resources and what A_j previously communicated to A_i is larger than a threshold, θ.

$$B_{ij}^t = \begin{cases} R_j^t, & \text{if } \left|R_j^t - B_{ij}^{t-1}\right| \geq \theta \\ B_{ij}^{t-1}, & \text{otherwise} \end{cases} \quad (8)$$

The number of excess tasks that A_i has at time t that arrived the remaining tasks in the waiting queue after its resources have been allocated.

$$O_i^t[a] = \max\left(w_i^t[a] - R_i^t[a], 0\right) \quad (9)$$

3. EXPERIMENTS AND RESULTS

We consider a k-connected topology in DSRAP where the first half of agents are heavily loaded and the other half are lightly loaded and examine under which connectivity levels the evenly-route algorithm outperforms the proportional-route algorithm. For this experiment, we investigate a 30 agent system where each agent has 120 resource A units and 140 resource B units. The average system load is kept at 20 where each agent's individual load is pulled from a uniform distribution centered on 35 or 5 with a variance of $\frac{4}{3}$. Our proportional-route agents always communicate (i.e. θ is 0). We show the effect of connectivity of the system with respect to system performance in figure 1. Our experiments show the best performance is when the heavily loaded

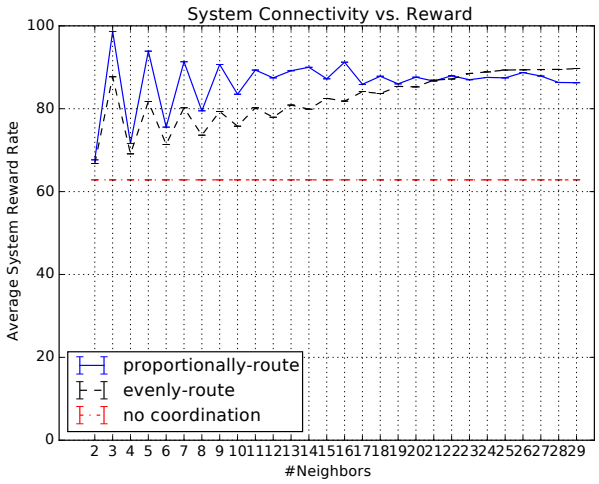

Figure 1: Utility Rate for a k-connected topology where the first half agents are heavily loaded and the other half are lightly loaded.

agents may be able to shed their excess tasks to a lightly loaded agent without their tasks competing with others to be allocated (i.e. k is 3). We note the low performance spikes is when heavily loaded agents are not connected to many lightly loaded agents. We notice that the proportionally routing agents' performance converges to a value in between the least connected system (i.e. k = 2) and the best connectivity level (i.e. k = 3). Highly-connected systems may experience inconsistency with actual load due to the high flux of sent tasks. We note that the evenly routing system's performance increases since the heavily loaded agents have more of an opportunity to forward tasks to more lightly loaded agents.

We demonstrate the versatility of our equations by considering the k-connected topology such that agents alternate between heavy loads and light loads. We show that the percentage of sent tasks completed for this system using the evenly-routing algorithm. All tasks sent from a heavily loaded agent to a lightly loaded agent would be completed; thus, if we calculate the number of connections, we can calculate the number of completed tasks. The number of connections for a single heavily loaded agent to a lightly loaded agent follows a variation of the even numbers repeated sequence where the nth term is $2 * \lfloor \frac{n}{2} \rfloor$ [5]. For this system, if k is even, each heavily loaded agent will be connected to $2 * \left\lfloor \frac{\frac{k}{2}+1}{2} \right\rfloor$ lightly loaded agents and, if k is odd, there would be 1 more lightly loaded agent. Our equation for the percentage of sent tasks completed follows for when k is odd. Consider that k is 8; Each heavily loaded agent will have 4 lightly loaded neighbors and 4 heavily loaded neighbors, thus half the tasks can be completed.

$$\frac{2 * \left\lfloor \frac{\frac{k}{2}+1}{2} \right\rfloor + 1}{k} \quad (10)$$

By understanding the schedulers performance analytically, we can determine when certain schedulers and routing algorithms are useful for a given environment.

REFERENCES

[1] A. C. Arpaci-Dusseau and D. E. Culler. Extending proportional-share scheduling to a network of workstation. 1997.

[2] K. Decker and V. Lesser. An approach to analyzing the need for meta-level communication. In *Proceedings of the 13th International Joint Conference on Artifical Intelligence - Volume 1*, IJCAI'93, pages 360–366, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[3] J. Gonzalez and N. Lacourly. A family of hamilton type methods for congressional apportionments. *Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle*, 26(1):31–40, 1992.

[4] D. E. Neiman, D. W. Hildum, V. R. Lesser, T. Sandholm, et al. Exploiting meta-level information in a distributed scheduling system. In *AAAI*, pages 394–400, 1994.

[5] J. Sellers. Encyclopedia(at) pommard.inria.fr, https://oeis.org/a052928, 2000.

[6] C. Zhang and V. Lesser. Coordinating multi-agent reinforcement learning with limited communication. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1101–1108. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[7] C. Zhang, V. R. Lesser, and P. J. Shenoy. A multi-agent learning approach to online distributed resource allocation. In *IJCAI*, pages 361–366, 2009.

[8] C. Zhang and J. A. Shah. Fairness in multi-agent sequential decision-making. In *Advances in Neural Information Processing Systems*, pages 2636–2644, 2014.