Multi-Tier Resource Allocation in Distributed Systems

THE UNIVERSITY OF SYDNEY

Thomas Ryan
Dr. Young Choon Lee
School of Information Technologies
FACULTY OF ENGINEERING & INFORMATION TECHNOLOGIES

INTRODUCTION

- Scheduling and resource allocation strategies for distributed computing systems such as MapReduce and scientific workflows like Montage [1] are an ongoing research problem; many have been published, but most-real world systems use some variation of a simple FIFO queue.
- Users are often more concerned with execution times of their jobs than scheduling properties such as strict fairness. Additionally, many schedulers often result in resource under-utilization.
- The Local Resource Shaper (LRS) proposed by Lu et. al. [2] introduces shaping resources for 'Active' and 'Passive' slots for Map Reduce tasks in Classic Hadoop.
- We generalize this into Multi-Tier Resource Allocation (MTRA), a resource allocation strategy for distributed jobs that are each made up of many small tasks. By allocating resources to tasks using a tier system that shapes resource usage of tasks in lower tiers, MTRA improves execution times by increasing resource utilization while minimizing resource contention. MTRA functions independently of the distributed system scheduler.

Applications

- Hadoop YARN [3] is a framework for distributed processing of jobs such as MapReduce over large scale datasets. Jobs are typically composed of many tasks that perform the same computation on different portions of a dataset.
- Scientific Workflows such as Montage also operate on large datasets, but are characterized by more complex precedence constraints than typical YARN applications.
- In both examples, I/O and communications between tasks, especially transfer of data between cluster nodes, can be a significant source of overhead in a distributed system.

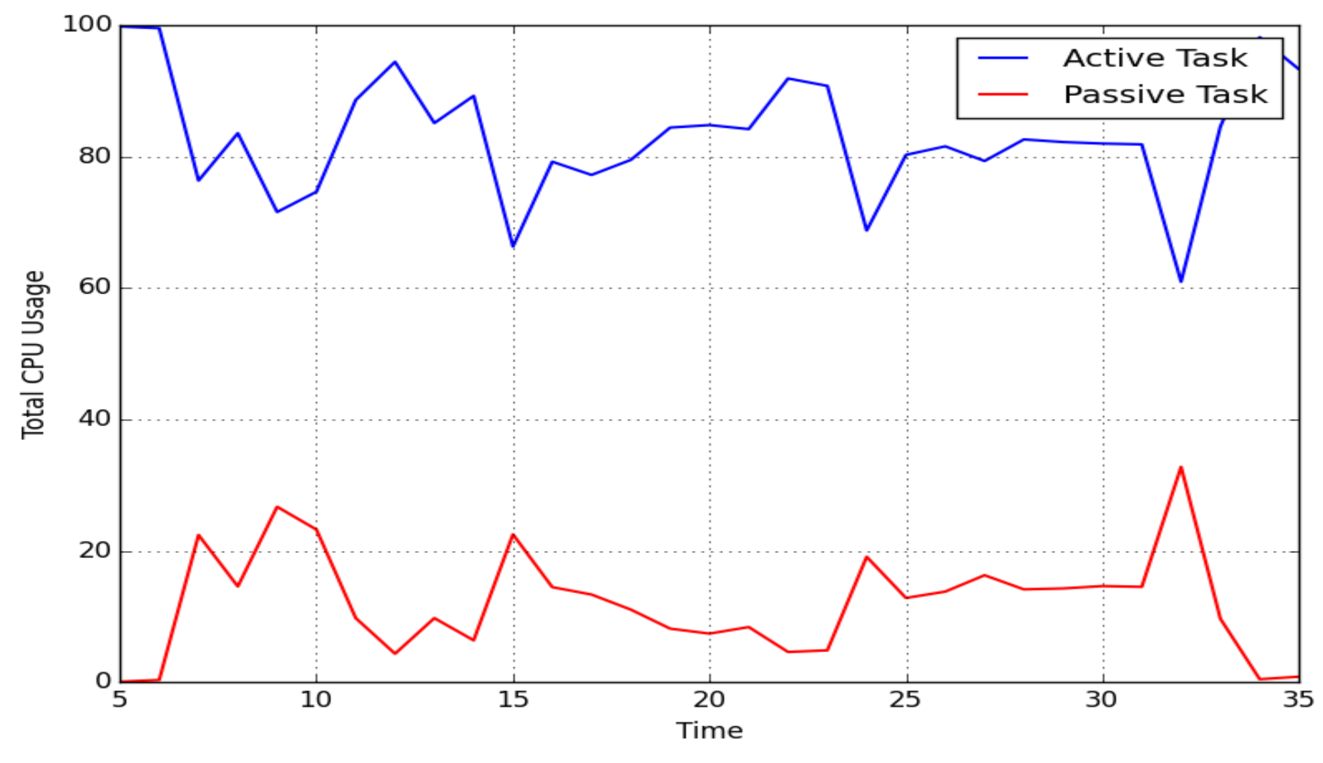

Fig. 1. Resource usage of Active and Passive tasks on one CPU core

MULTI-TIER RESOURCE ALLOCATION

- MTRA is based on the idea of allocating tasks in tiers; for example, an 'Active' and a 'Passive' task are co-located on each CPU core, and the Passive task's resource consumption is shaped such that it only consumes unused resources. Fig. 1 shows this complementary CPU usage. Cases with more than two tiers have also been investigated.

CONTRIBUTION

- We present Multi-Tier Resource Allocation (MTRA), a generalization of the use of Active and Passive tasks in LRS, and implement it in two new environments:
- MTRA in Hadoop YARN shows that two-tier resource allocation is effective in the YARN environment, despite the changes between Classic Hadoop and Hadoop YARN, and can create performance improvements of up to 49% in I/O bound tasks.
- When generalized for use with a Montage workflow, MTRA improves performance by up to 9%.

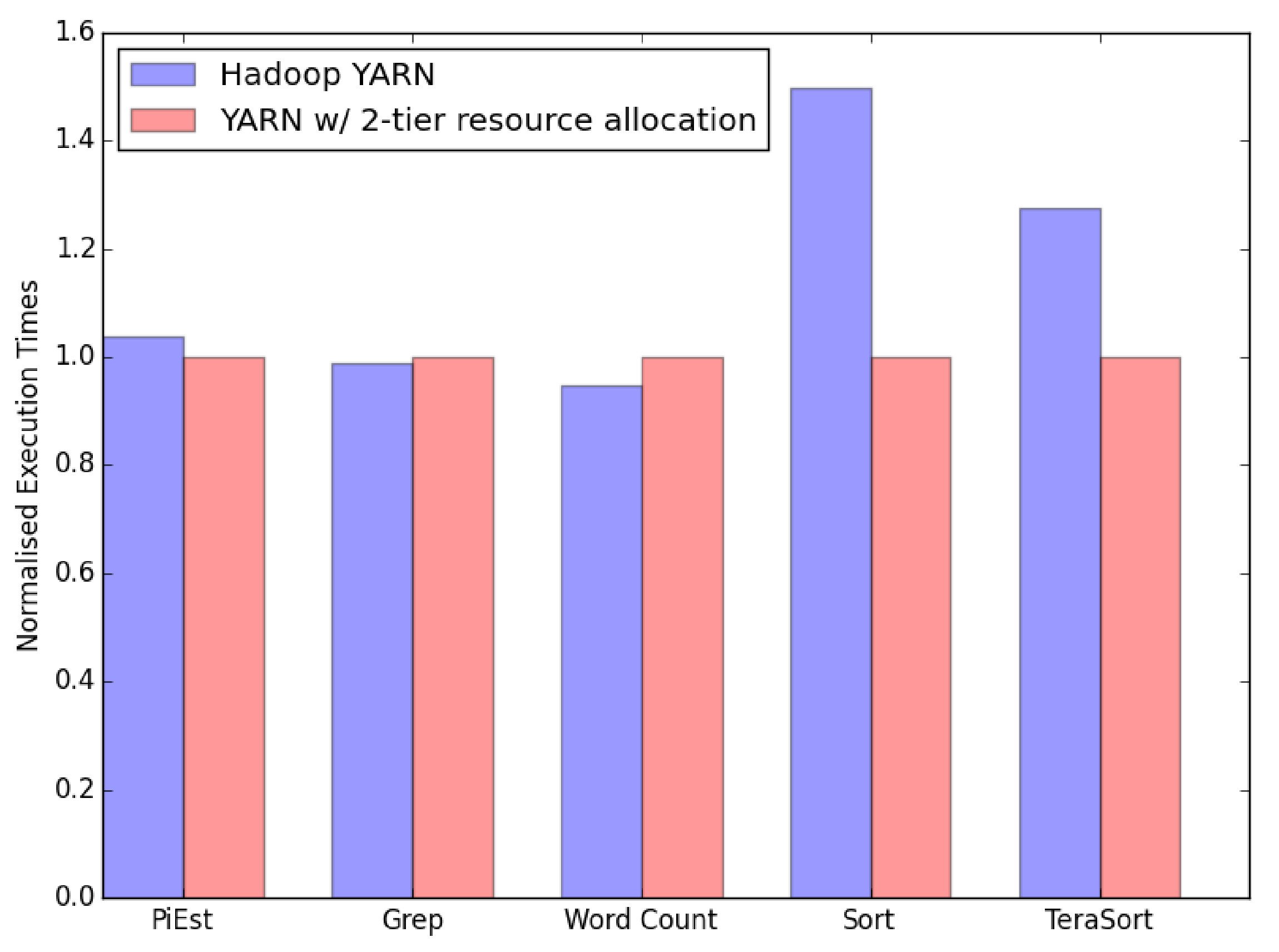

Fig. 2. Comparison of execution times for MapReduce benchmarks

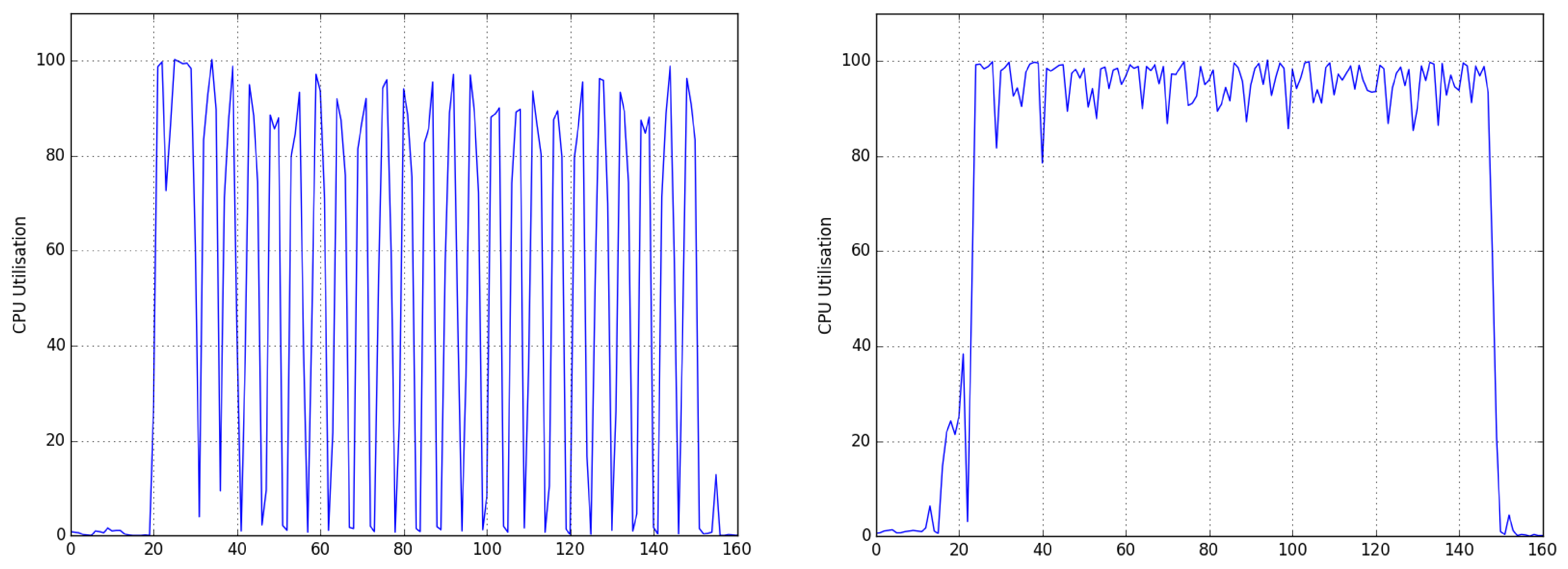

Fig. 3. CPU utilization in the PiEst benchmark : YARN (left) and YARN w/ MTRA (right)

EVALUATION

MTRA for Map Reduce in Hadoop YARN

- MTRA was implemented on top of YARN with one Active and one Passive task per CPU core. Resources were allocated in the ratio 100:1 Active to Passive using Linux Cgroups.
- Experiments were performed on a cluster of 11 m3.2xlarge nodes in Amazon EC2. 30GB input data was used for each job. The benchmarks are taken from the MapReduce literature and represent a range of resource usage profiles.
- As shown in Fig. 2, MTRA produces a slight performance increase for the CPU-bound job PiEst, and large performance increases for I/O-bound jobs Sort and TeraSort.
- Fig. 3 shows how the two-tier system increases resource utilization; we see significant drops in CPU utilization in YARN as each task finishes, while MTRA maintains much higher utilization.

MTRA for Montage Scientific Workflows

- Rather than running a second task as Passive, here the Passive idea was generalized, and we define a loading job to run as the Passive task, which loads the data for the next task in the workflow.
- Amazon EC2 c3.xlarge nodes were used over a range of cluster sizes, as shown in Fig. 4, with a 2.0 degree Montage workflow.
- Fig. 4 shows how MTRA allows for increased performance in a 2-node cluster. Performance converges in larger cluster sizes as the critical path length is approached.

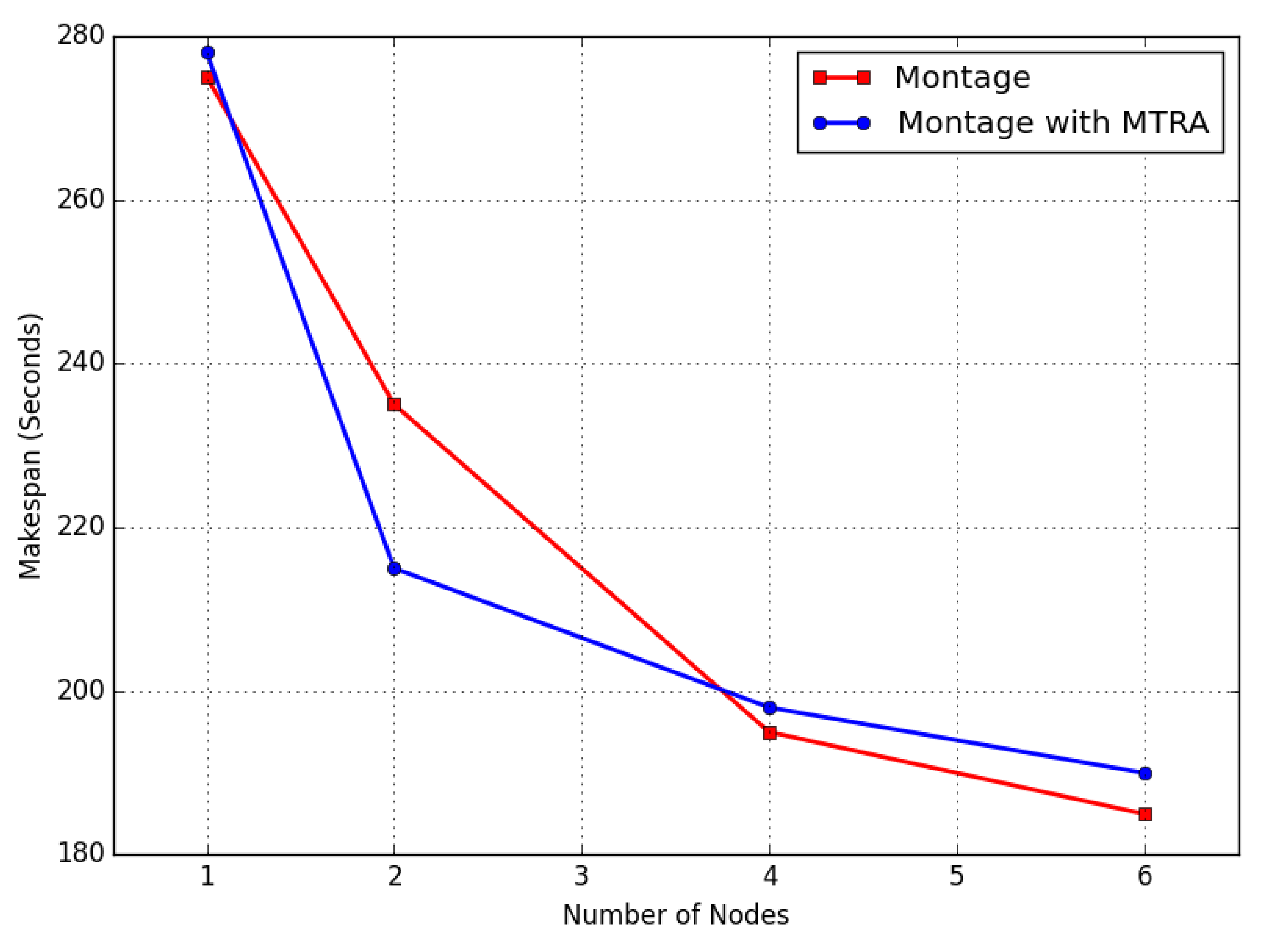

Fig. 4. Comparison of Montage workflow makespans for different cluster sizes

FUTURE WORK

- While we have demonstrated performance improvement in several cases, it is largely in I/O bound jobs. Further work into effective use of multi-tier resource allocation systems for jobs with different resource usage profiles is still required.

REFERENCES

- [1] Montage: An Astronomical Image Mosaic Engine, http://montage.ipac.caltech.edu/index.html
- [2] Lu et. al., *Local Resource Shaper for MapReduce,* CloudCom 2014, IEEE
- [3] Vavilapalli et. al., (2013), *Apache Hadoop YARN: Yet Another Resource Negotiator,* SOCC 2013, ACM