Goodbye to Fixed Bandwidth Reservation: Job Scheduling with Elastic Bandwidth Reservation in Clouds

Haiying Shen[*], Lei Yu[§], Liuhua Chen [&], and Zhuozhao Li[*]

[*]Department of Computer Science, University of Virginia, USA

[§]College of Computing, Georgia Institute of Technology, USA

[&]Department of Electrical and Computer Engineering, Clemson University, USA

Outline

Introduction

- Cloud computing
  - multiplexes computation, storage and network resources among different tenants

- Network resource
  - bandwidth competitions among network flows
  - significantly varying data transmission latency
  - unpredictable application performance
  - may lead to long networking time and hence long job execution time

Related Work

- Bandwidth reservation in datacenter networks
- Little work exploits the relationship between job execution time and VM bandwidth in bandwidth reservation
- Bandwidth pre-specification is not suitable for ordinary tenants who may not have specialized network knowledge

Introduction

- Tenants need to provide
  - job description with the required job deadline
  - a reward value represents the worth of completing the job by its deadline
    - ➢ maximize the number of completed jobs
    - ➢ the number of service-level agreement (SLA) conformances
    - ➢ the social welfare of customers or the profit of the cloud provider
- Cloud provider needs to
  - decide the appropriate VM bandwidth in the virtual network abstraction

Goal

- Propose a new cloud service model to
  - leverage the elastic feature of the bandwidth resource
  - maximize the total rewards of jobs (referred to as system reward)
  - reduce job execution time

System Model

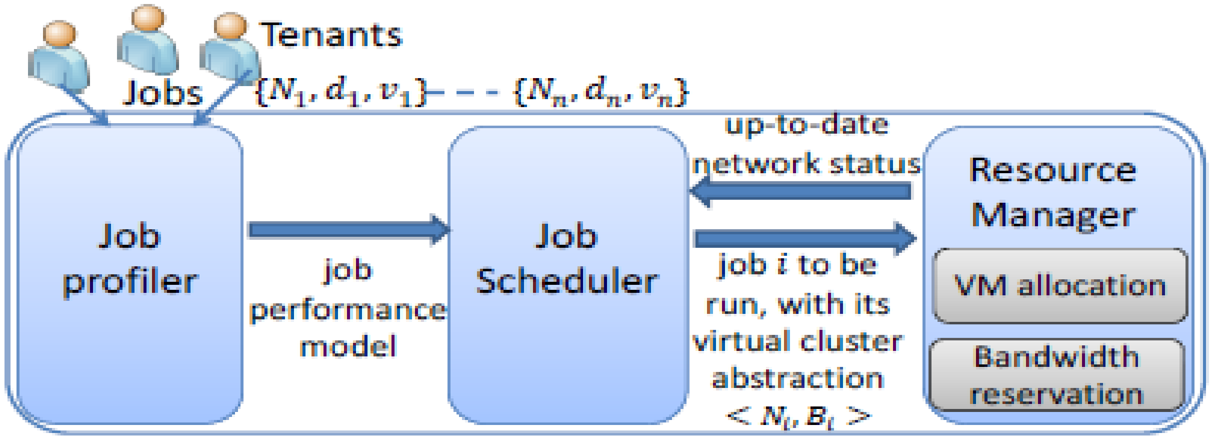
- Job model
  - each job is characterized by a tuple $\{N_j, d_j, v_j\}$
  - N_j, the number of VMs job j requires
  - d_j, the desired deadline of job j
  - v_j, the job's reward value that represents the worth of a successful completion of the job before its deadline d_j

- Virtual network abstraction
  - use a simple virtual cluster abstraction proposed in Oktopus

H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in Proc. of SIGCOMM, 2011.

UNIVERSITY of VIRGINIA

System Model


Tenants
Jobs
$\{N_1, d_1, v_1\}$ - - - $\{N_n, d_n, v_n\}$
Job profiler
job performance model
Job Scheduler
up-to-date network status
job i to be run, with its virtual cluster abstraction $< N_i, B_i >$
Resource Manager
VM allocation
Bandwidth reservation

- Job profiler is responsible for profiling each application offline, that is, determining the job completion time based on allocated VM bandwidth.
- Job Scheduler is responsible for determining the optimal schedule of jobs.
- Resource Manager maintains and reports to the job scheduler the up-to-date information of the datacenter resource.

Job Profiler

- Job profiler
  - Good predictor of the applications, i.e., high prediction accuracy
  - Well exploited to optimize the resource allocation

- How the job profiler works?
  - No-elongation threshold bandwidth
    - ➢ Once the VM bandwidth drops below the threshold, the execution time is elongated monotonically
  - For job j, output a sequence of tuples $\left\{\left(B_1^j, \tau_1^j\right), \dots, \left(B_k^j, \tau_k^j\right)\right\}$
  - $B_i^j, i \in \{1,2,\dots,k\}$, bandwidth option for job j
  - $\tau_i^j, i \in \{1,2,\dots,k\}$, corresponding execution time of job j
  - Assume B_k^j is the largest bandwidth, which is the no-elongation threshold

Proposed Algorithm

- Assume that the scheduler has full knowledge of n jobs in a batch before computing
- Formulate the job scheduling problem as an optimization problem
- Input -- a set of jobs ready to be scheduled, each with
  - job information $\{N_j, d_j, v_j\}$
  - profile tuples $\left\{\left(B_1^j, \tau_1^j\right), \ldots, \left(B_k^j, \tau_k^j\right)\right\}$
- Output – a schedule for all the jobs which
  - maximizes the total reward value of jobs that are completed before their deadlines, namely system reward
- An offline and heuristic algorithm

Proposed Algorithm

- Let J_{ready} be a set of n jobs ready to be scheduled in the system
- First, at the beginning $t = t_0$, the algorithm determines a set of jobs, which can be concurrently allocated and executed, denoted by J_{run}
  - Choose the smallest bandwidth option for each job in J_{ready} that can satisfy the deadline requirement
  - Find valid allocation for each job in the virtual cluster abstraction
  - Give higher priority for the jobs with higher reward and smaller deadline
  - For every job that has a valid virtual cluster placement and bandwidth allocation, put the job j into J_{run}
  - Receive the cluster resource status from the resource manager
  - Re-calculate the VM placement after a job finishes and a new job is added

Proposed Algorithm

- So far, J_{run} includes all jobs scheduled with their smallest bandwidth options that can meet their deadlines.

- Next, the algorithm optimizes the VM bandwidth selections for jobs in J_{run} by minimizing their execution time
  - Determine the longest job that has the largest execution time, say J_L
  - Increase the bandwidth selection of J_L to the next highest bandwidth in the profile
  - See if the bandwidth selection is admissible under its VM placement, i.e., the bandwidth will not exceed the link capacity
  - If yes, the execution time is reduced
  - Repeat the above steps until it cannot further reduce the execution time of the longest job

Proposed Algorithm

- Finally, the algorithm determines the set of jobs J_{run} that can be concurrently allocated and run at time $t = t_0$. The algorithm also reduces the execution time of the jobs in J_{run}.
- Then, the algorithm schedules the remaining jobs in J_{ready} for time $t = t_1$.
- Repeat the above steps until all the jobs are scheduled.
- Accordingly, the resource manager starts scheduling the jobs.

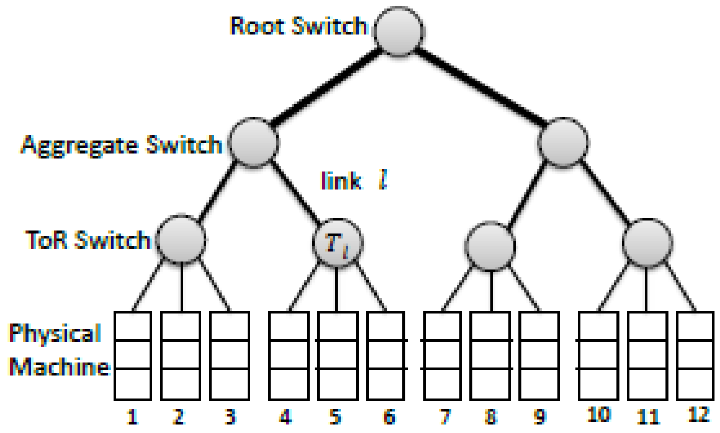
Performance Evaluation

- Experiments
  - We ran Facebook-2010 workload in simulation and on a 24-node CloudLab cluster and collected the job information for each job
    - Execution time, denoted by τ_1^j
    - Number of VMs needed by the job
    - Bandwidth usage B over the execution time period
  - Reward values for each job were generated from a uniform distribution within an interval of [100,1000]
  - The bandwidth options were generated from B by multiplying a set of ratios, $\{0.2, 0.4, 0.6, \ldots, 2.0\}$
  - The execution time under different bandwidth options was determine by a linear model
  - The results are the average over 20 runs

- Comparison method
  - Earliest Deadline First (EDF)
  - Oktopus
  - PS-P in FairCloud

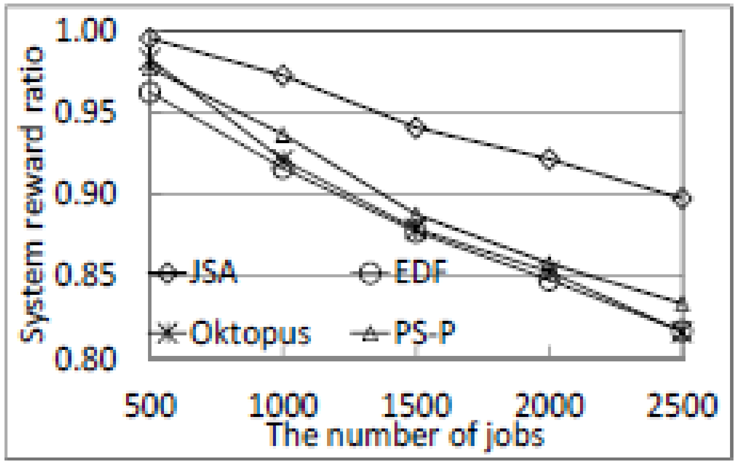
Performance Evaluation

- Three-level tree topology
  - 4 VM slots each machine, and 1Gbps link connected to ToR switch
  - 40 machines in a rack, 40 racks
  - 20 ToRs connected a level-2 aggregation switch
  - 2 aggregation switches connected a root switch
  - In total 1600 machines
  - 8Gbps link bandwidth between ToR and aggregation switch
  - 32Gbps link bandwidth between aggregation and core switch
  - Oversubscription ratio 5:1

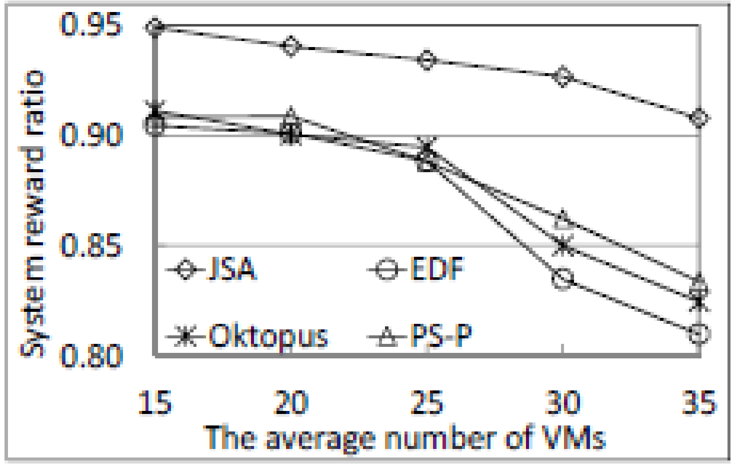


Performance Evaluation

- System reward ratio
  - Defined as the total reward value of the jobs that complete before their deadlines divided by the total reward value of all the jobs



(a) System reward ratio vs. the number of jobs.



(b) System reward ratio vs. the average number of VMs of each job.

The number of jobs varies from 500 to 2500.
$EDF < Oktopus \approx PSP < JSA$

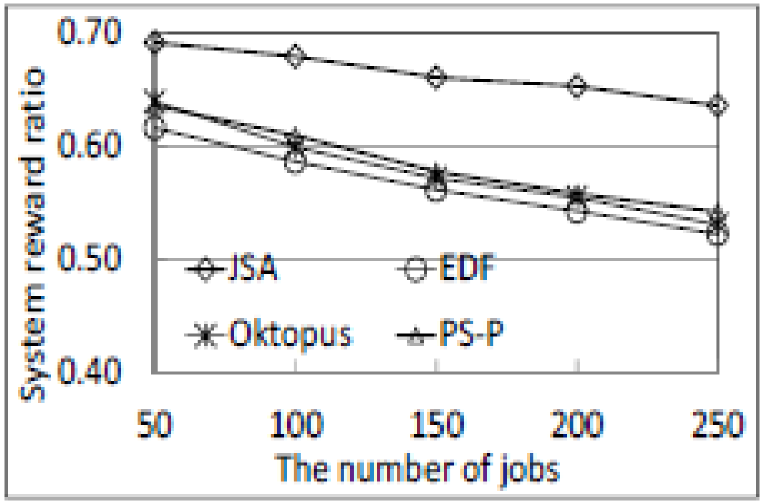
The average number of VMs for each job varies from 15 to 35.
$EDF < Oktopus \approx PSP < JSA$

Performance Evaluation

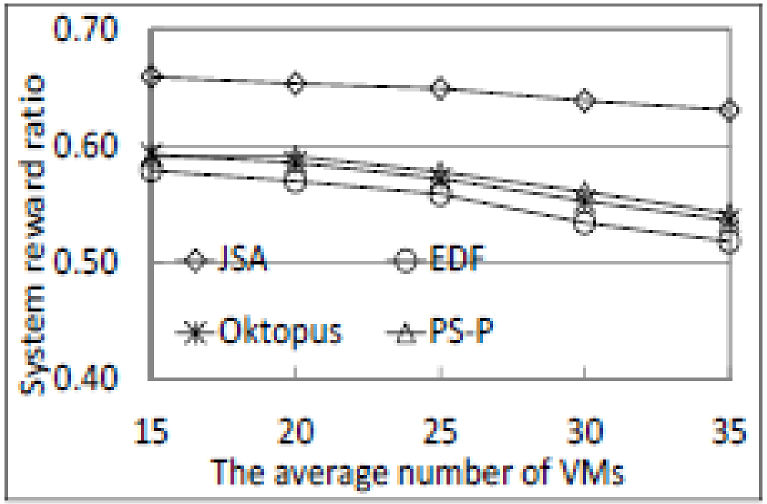
- Real implementation of the job scheduler
- Three-level tree topology
  - 1 VM slot each machine, and 1Gbps link connected to ToR switch
  - 6 machines in a rack, 4 racks
  - 2 ToRs connected a level-2 aggregation switch
  - 2 aggregation switches connected a root switch
  - In total 24 machines
  - 1.2Gbps link bandwidth between ToR and aggregation switch
  - 0.48Gbps link bandwidth between aggregation and core switch
  - Use the network utility tools *tc* and *iptables* to implement these bandwidth limits
  - Oversubscription ratio 5:1
- Workloads
  - Data transfer jobs
  - Transfer size is based on the job information collected from Facebook workload

UNIVERSITY of VIRGINIA

Performance Evaluation



(a) System reward ratio vs. the number of jobs.



(b) System reward ratio vs. the average number of VMs of each job.

The number of jobs varies from 50 to 250.

$EDF < Oktopus \approx PSP < JSA$

The average number of VMs for each job varies from 15 to 35.

$EDF < Oktopus \approx PSP < JSA$

Conclusion

- We design a new cloud service model, in which a tenant only needs to specify the job deadline.
- We aim to find a job schedule (the reserved bandwidth in the virtual cluster abstraction, the running start time of each job, and the VM placement in the datacenter) to satisfy the deadline requirements while maximizing the total rewards of jobs that are finished by their deadlines.
- Due to the NP hardness of this problem, we propose a heuristic scheduling algorithm to find the job schedule.
- In the future, we will explore on-line scheduling algorithms with automatic bandwidth reservation.

UNIVERSITY of VIRGINIA

Thank you!

Questions & Comments?

Dr. Haiying Shen

hs6ms@Virginia.edu

Associate Professor

Pervasive Communication Laboratory

University of Virginia