Deadline Guaranteed Service for Multi-Tenant Cloud Storage

Guoxin Liu and Haiying Shen

Presenter: Haiying Shen

Associate professor

*Department of Electrical and Computer Engineering, Clemson University, Clemson, USA

Outline

- Introduction
- Related work
- PDG design
- Evaluation
- Conclusion

Introduction

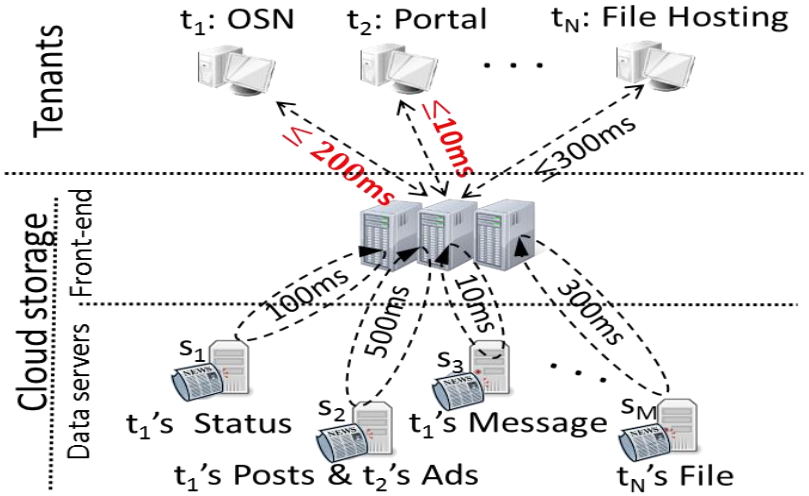
- Cloud storage
  - Tenant perspective
    - Save capital investment and management cost
    - Pay-as-you-go
    - Service latency vs. revenue
      - Amazon portal: increasing page presentation by 100ms reduces user satisfaction and degrades sales by 1%.
      - Challenge: Reduce the fat-tail of data access latency

Introduction

- Cloud storage
  - Provider perspective
    - Cost-efficient service
      - Cost saving
        - Resource sharing between tenants
      - Energy saving
        - Workload consolidation
  - Encounter problem
    - Unpredictable performance to serve tenants' data requests (e.g. service latency)

Introduction

- Problem harmonization
  - Service level agreements (SLAs) [1] (e.g. 99% requests within 100ms) baked into cloud storage services
- Challenge
  - How to allocate data: non-trivial



[1] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowstron. Better Never than Late: Meeting Deadlines in Datacenter Networks. In Proc. of SIGCOMM, 2011.

Introduction

- Our Approach:
  - PDG: Parallel Deadline Guaranteed scheme
    - Goals: traffic minimization, resource utilization maximization and scheme execution latency minimization
    - Assurance: Tenants' SLAs
    - Operation: serving ratios among replica servers and creating data replicas
    - Enhancement: prioritized data reallocation for dynamic request rate variation

Outline

- Introduction
- Related work
- PDG design
- Evaluation
- Conclusion

Related work

- Deadline-aware networks
  - Bandwidth apportion
    - According to deadline
  - Dataflow schedule
    - Prioritize different dataflows
  - Caching system
    - Cache recent requested data
  - Topology optimization
- Optimized cloud storage
  - Throughput maximization
  - Data availability insurance
  - Replication strategy to minimize cost
- Problem
  - None of them achieve multiple goals as PDG in cloud storage

Outline

- Introduction
- Related work
- PDG design
- Evaluation
- Conclusion

PDG design

- Data allocation problem
  - Heterogeneous environment
    - Different server capacities
    - Different tenant SLAs
    - Variations of request rates
  - Multiple constraints
    - SLA insurance
    - Storage/service capacity limitation
  - Multiple goals
    - Network load, energy consumption and computing time minimization
  - Time complexity
    - NP-hard

- Data reallocation for deadline guarantee as a nonlinear programming

$$\min \left(|\mathcal{M}_u| + \beta \Phi_{f'}\right) \tag{1}$$

subject to

$$\forall t_k, \mathcal{P}_{t_k}/(1-\epsilon_{t_k}) \geq 1 \tag{2}$$

$$\sum_{s_n \in \mathcal{M}_u} X_{s_n}^{\mathcal{D}_i} \cdot \mathcal{H}_{s_n}^{\mathcal{D}_i} = 1 \quad \forall \mathcal{D}_i \in \mathbf{D} \tag{3}$$

$$\sum_{\mathcal{D}_i \in \mathbf{D}} S_{\mathcal{D}_i} \cdot X_{s_n}^{\mathcal{D}_i} \leq C_{s_n} \quad \forall s_n \in \mathcal{M} \tag{4}$$

$$\sum_{s_n \in \mathcal{M}} X_{s_n}^{\mathcal{D}_i} \geq r \quad \forall \mathcal{D}_i \in \mathbf{D} \tag{5}$$

$$X_{s_n}^{\mathcal{D}_i} \in \{0,1\} \quad \forall s_n \in \mathcal{M}, \forall \mathcal{D}_i \in \mathbf{D} \tag{6}$$

$$0 \leq \mathcal{H}_{s_n}^{\mathcal{D}_i} \leq 1 \quad \forall s_n \in \mathcal{M}, \forall \mathcal{D}_i \in \mathbf{D} \tag{7}$$

PDG design

- System assumption
  - Each server = M/M/1 queuing system
    - Request arrival rate follows Poisson process
    - The service time follows an exponential distribution
    - Single queue
  - Based on the model, we can derive the CDF of service time of requests
    - Sn: server n; $F()_{sn}$: CDF of service time; λ_{sn}: request arrival rate, μ_{sn}: service rate

$$F(t)_{s_n} = 1 - e^{-(\mu_{s_n} - \lambda_{s_n}) \cdot t}$$

PDG design

$f_{t_k}(j)$: probability density function that tenant t_k's request targets j servers

To guarantee SLA:

$$f(b_{t_k}) = \sum_{j \in [1,n]} (b_{t_k})^j \cdot f_{t_k}(j) = 1 - \epsilon_{t_k}$$

We use x_{t_k} to denote the solution for $b_{t_k} \in (0, 1)$

Lemma 2. If $\forall t_k \forall s_n, s_n \in \Re_{t_k} \Rightarrow F(d_{t_k})_{s_n} \geq x_{t_k}$, then the SLAs are guaranteed.

Definition 1. *We use* $\mathcal{K}_{t_k}$ *to denote* $|ln(1 - x_{t_k})/d_{t_k}|$, *and call* $\mathcal{K}_{t_k}$ *the* deadline strictness *of tenant* t_k, *which reflects the hardness of* t_k*'s deadline requirement.*

PDG design

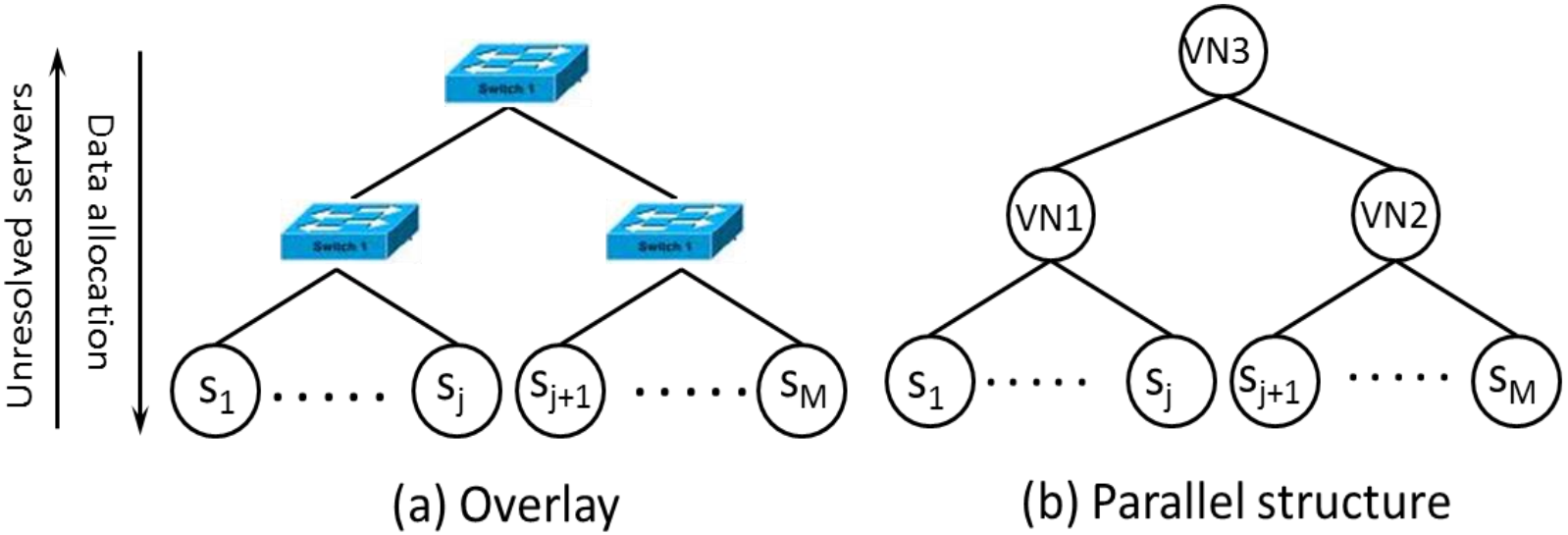
- System assumption
  - To guarantee SLA
    - λ_{sn}^{g}: maximum arrival rate to Sn; K_{tk}: tenant k's deadline strictness, a variable related to the deadline and allowed percentage of requests beyond deadline

$$\lambda_{s_n}^{g} = \mu_{s_n} - \max\{\mathcal{K}_{t_k} : s_n \in \Re(t_k)\}$$

  - System requirement to achieve multiple goals with constraints
    - Each server has a request arrival rate lower than λ_{sn}^{g}
    - Consolidate workloads of requests to fewer servers
    - Minimize replications and replicate with proximity-awareness
    - Distributed data allocation scheduling

PDG design

- Tree-based Parallel Process
  - Unsolved servers
    - Underloaded and overloaded servers
  - Each VN (virtual node) runs PDG
    - Serving ratio reassignment
    - Data replication
    - Report unsolved servers to parents



(a) Overlay (b) Parallel structure

PDG design

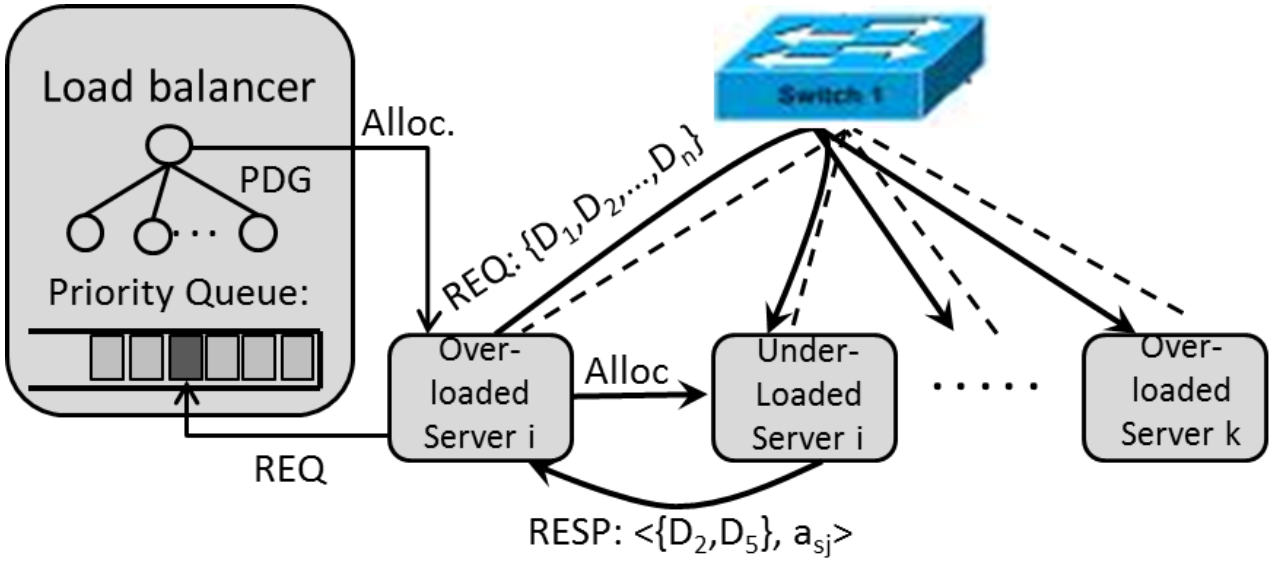
- Serving Ratio Reassignment
  - Loop all replicas in overloaded servers to redirect the serving ratio to replicas in underloaded servers
- Data Replication
  - Create a new replica in the most overloaded server to the most underloaded servers
  - Reassign serving ratio for this replica
  - Loop until no overloaded servers

PDG design

- Workload consolidation
  - Goal
    - Energy consumption minimization
  - Trigger
    - If total available service rate is larger than the minimum λ_{sn}^{g}
  - Procedure
    - Sort servers in an ascending order of λ_{sn}^{g}
    - Deactivate the first server
      - If SLA is guaranteed, deactivate next server
      - Otherwise, termination

PDG design

- Prioritized data reallocation
  - SLA guarantee under request arrival rate variation
    - Select the most heavily requested data items
    - Broadcast within rack for request ratio reassignment
    - Report unsolved servers to load balancer
    - Load balancer conducts PDG to balance requests over racks



Outline

- Introduction
- Related work
- PDG design
- Evaluation
- Conclusion

Evaluation

- Experimental settings
  - 3000 data servers
    - [6TB, 12TB, 24TB] storage capacity
    - [80,100] service capacity
    - Fat-tree with three layers
  - 500 tenants
    - [100ms, 200ms] Deadline
    - 5% maximum allowed percentage of requests beyond deadline
    - [100, 900] data partitions with request arrival rate follows distribution in [2]

[2] CTH Trace. http://www.cs.sandia.gov/Scalable IO/SNL_Trace_Data/, 2009.

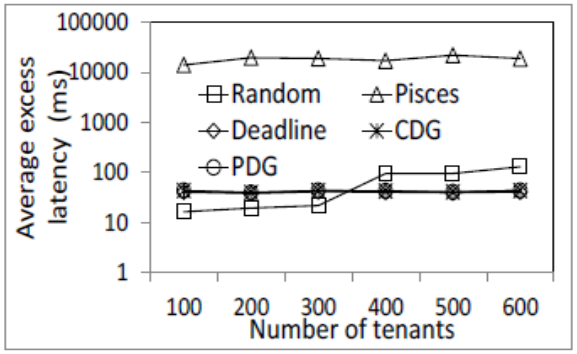
Evaluation

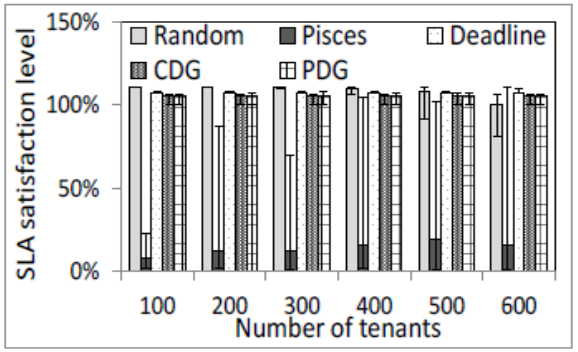
- Comparison methods
  - Deadline guarantee periodically
    - Random: randomly place data among servers
    - Pisces[3]: storage capacity aware data first fit
    - Deadline: deadline aware first fit
    - CDG: centralized load balancing of PDG
  - Deadline guarantee dynamically
    - PDG_H: PDG using highest arrival rates for all data
    - PDG_NR: PDG without prioritized data reallocation
    - PDG_R: PDG with prioritized data reallocation

[3] D. Shue and M. J. Freedman. Performance Isolation and Fairness for Multi-Tenant Cloud Storage. In Proc. of OSDI, 2012.

Evaluation

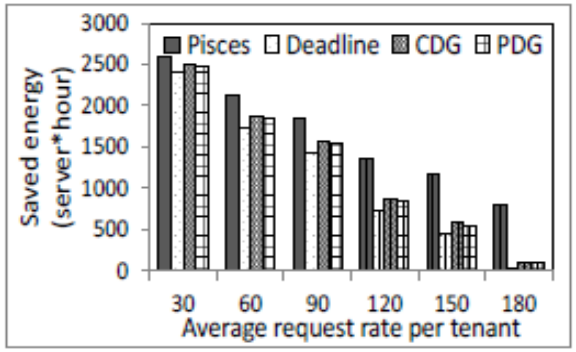
- Important metrics
  - Excess latency: avg. extra service latency time beyond the deadline for a request
  - SLA satisfaction level: actual percentage of requests within deadline/required percentage
  - QoS of SLA: the minimum SLA satisfaction level among all tenants
- SLA guarantee
  - Average excess latency: shortest, best performance in deadline violation case
  - SLA ensured: slightly larger than 100%

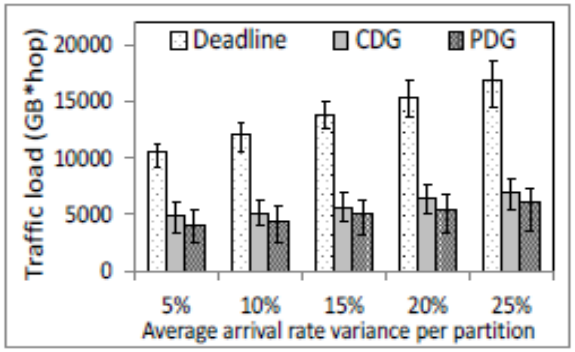




Evaluation

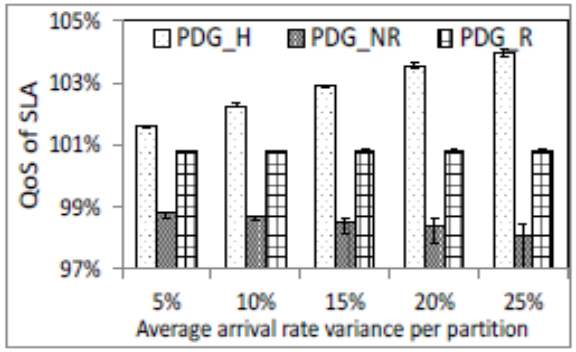
- Objective achievement
  - Effectiveness of workload consolidation
    - Energy: maximized energy saving
  - Effectiveness of tree-based parallel process
    - Traffic load: minimized network for data reallocation
      - Bottom up process introduces a proximity-aware replication

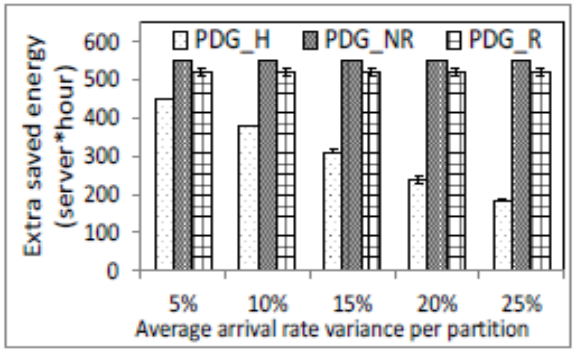




Evaluation

- Dynamic SLA guarantee and energy savings
  - Performance of SLA guarantee
    - QoS of SLA: PDG_H and PDG_R both guarantee SLA
      - SLA-aware dynamical request ratio and data reallocation
  - Performance of energy saving
    - Energy savings: PDG_R saves more energy than PDG_H
      - Use more servers when needed





Outline

- Introduction
- Related work
- PDG design
- Evaluation
- Conclusion

Conclusion

- PDG: parallel deadline guaranteed scheme, which dynamically moves data request load from overloaded servers to underloaded servers to ensure the SLAs
  - Mathematical model to give an upper bound on the request arrival rate of each server to meet the SLAs
  - A load balancing schedule to quickly resolve the overloaded servers based on a tree structure
  - A server deactivation method to minimize energy consumption
  - A prioritized data reallocation to dynamically strengthen SLA
- Future work
  - Real deployment to examine its real-world performance

Thank you!

Questions & Comments?

Haiying Shen

shenh@clemson.edu

Electrical and Computer Engineering

Clemson University