Integrated Resource Management for Cluster-based Internet Services

Kai Shen Hong Tang Tao Yang LingKun Chu

Published on OSDI2002

Presented by Chunling Hu

About the Authors

- Kai Shen: Assistant Professor of DCS at Univ of Rochester. Research interest is in Parallel and distributed systems, computer networks and Web searching

About the Authors

- Hong Tang: PhD student of DCS at UCSB. Research interest is in parallel and distributed systems
- Lingkun Chu: PhD student of DCS at UCSB. Research interest is in parallel and distributed systems, especially cluster-based network services

Tao Yang: Associate Professor of DCS at UCSB. Research interest is in Parallel and distributed systems, high performance scientific computing, cluster-based network services, and Internet search. He graduated from Rutgers in 1993 and is currently the Chief Scientist at Ask Jeeves

Outline

- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level request Distribution and Scheduling
- Performance Evaluations on a Linux Cluster
- Related Work and the Conclusion

Introduction

- Large-scale resource-intensive Internet services hosted on server clusters.
  - Yahoo, MSN, Google, Teoma/Ask Jeeves ...
- Challenges/requirements for resource management:
  - Bursty client requests require scalability and robustness;
  - Online users require interactive responses;
  - Resource (CPU, IO)–hungry service processing and large user traffic require efficient resource utilization;
  - Fluctuating user traffic requires adaptive resourece management;
  - Differentiated services are supported to different classes of user requests.

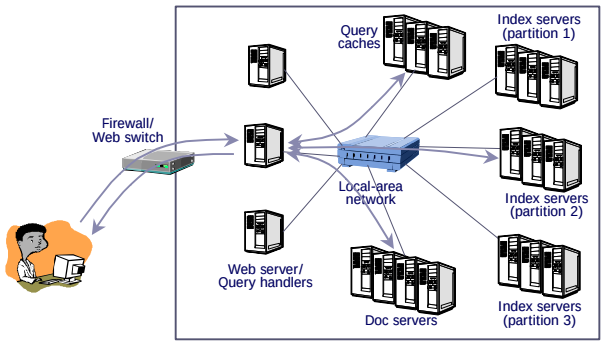
Main Point

- Design and implementation of an integrated resource manegement framework for cluster-based services (Part of Neptune system)
- Advanteges
  - Flexible resource management objectives
    - Maximizing aggregated service yield from all requests
    - Service differentiation
  - Fully decentralized clustering architecture with quality-aware resource management
  - Efficient resource utilization under quality constraints
    - two-level request distribution and scheduling scheme

Next...

- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level Mechanism
- Trace-driven Performance Evaluations on a Linux Cluster
- Neptune deployments, Related Work and the Conclusion

Architecture of Targeted Services: Document Search Engine



Next…

- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level request Distribution and Scheduling
- Performance Evaluations on a Linux Cluster
- Related Work and the Conclusion

Quality-aware Resource Utilization Efficiency

- Throughput: measure resource utilization efficiency.
- Service response time: measure client-perceived service quality.
- Aggregate service yield: measure quality-aware resource utilization efficiency.
  - Fulfillment of each service request generates quality-aware service yield – a function of service response time.
  - Service yield function $Y(r)$ – specified by service providers
  - System goal – maximizing aggregate service yield: $\sum_r Y(r)$

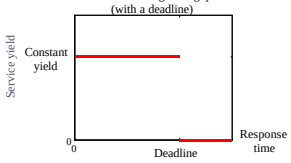
Sample Service Yield Functions

$$Y_{throughput}(r) = \begin{cases} C & \text{if } 0 \le r \le D, \\ 0 & \text{if } r > D. \end{cases}$$

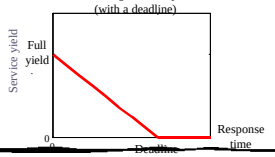
$$Y^{resptime}(r) = \begin{cases} C(1-\frac{r}{D}) & \text{if } 0 \le r \le D, \\ \end{cases}$$

$$Y^{hybrid}(r) = \begin{cases} C & \text{if } 0 \le r \le D', \\ C-(C-C')\frac{r-D'}{D-D'} & \text{if } D' \le r \le D \\ \end{cases}$$

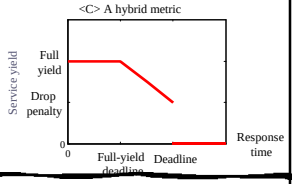
<A> Maximizing throughput (with a deadline)



<B> Minimizing mean response time (with a deadline)



<C> A hybrid metric



Service Differentiation

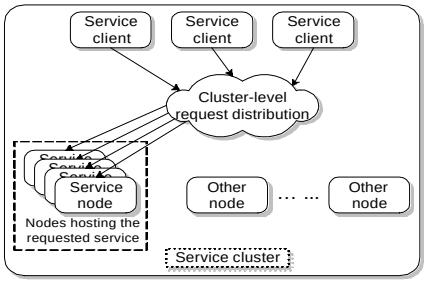
- Service class – a category of service accesses that enjoy the same level of service support.
  - Client identities: paid vs unpaid, consumers vs corporate partners.
  - Service types or data partitions: order placement vs catalog browsing.
- Service differentiation in Neptune
  - Differentiated service yield function for efficient resource utilization
  - Proportional resource allocation guarantee, preventing starvation for low priority service classes

Next…

- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level request Distribution and Scheduling
- Performance Evaluations on a Linux Cluster
- Related Work and the Conclusion

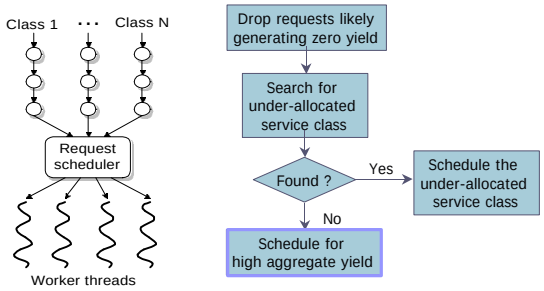
Two-level request Distribution and Scheduling(1)

- Cluster-level : class-aware load balancing scheme, random polling policy



Two-level request Distribution and Scheduling(2)

- Node-level : multi-queue scheduler(1 queue per service class)



Node-level service scheduling(1)

Estimating Resource Consumption

- Resource consumption of class C_k at time t:

$$\alpha_k(t) = \sum_{\{r \mid r \in C_k \text{ and } ct(r) \le t\}} \beta^{t-ct(r)} s(r), \quad 0 < \beta < 1$$

β = 0.95

r – request

ct(r) – completion time

s(r) – measured resource usage, accumulated CPU consumption

$\frac{\alpha_k(t)}{\sum_{i=1}^{N} \alpha_i(t)}$ is compared with the guaranteed share to search for under-allocated service classes

Node-level service scheduling(2)

Achieving High Aggregate Yield

- Offline optimal scheduling is NP-complete.
  - Different priority-based scheduling policies

Policy	**Priority** (the smaller the higher)
EDF	Relative deadline;
YID	Relative deadline divided by expected yield;
Greedy	Expected resource consumption divided by expected yield;
Adaptive	Dynamically switch between YID (in under-load) and Greedy (in overload).

Next…

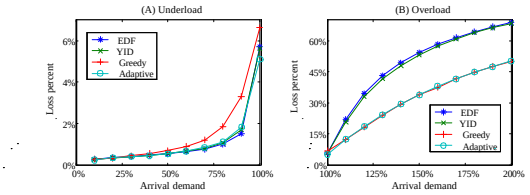
- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level request Distribution and Scheduling
- Performance Evaluations on a Linux Cluster
- Related Work and the Conclusion

Experimental Evaluation Settings

- Evaluation platform
  - A cluster of Linux servers connected by switched Ethernet.
- Objective of evaluation
  - Demonstrate the performance, scalability and service differentiation achieved by the proposed techniques
- Workload I: differentiated search
  - Document search on a 2.5GB search index.
  - Based on 1.5M search queries selected from an one-week access trace at Ask Jeeves search in January 2002.
  - Three service classes are based on same service type
  - "Service yield"-based priority order: Gold : Silver : Bronze = 4:2:1.
  - Only non-cache requests are concerned
- Workload II: CPU-spinning micro-benchmark.
  - Three service classes are based on different service types
  - Poisson process arrival; exponentially-distributed service processing time.

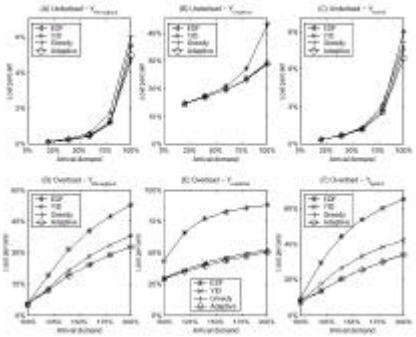
Evaluation on Scheduling Policies (16 nodes aggregate)—on Micro-benchmark

Performance Metric: $\text{LossPercent} = \frac{\text{OfferedYield} - \text{RealizedYield}}{\text{OfferedYield}} \times 100\%$

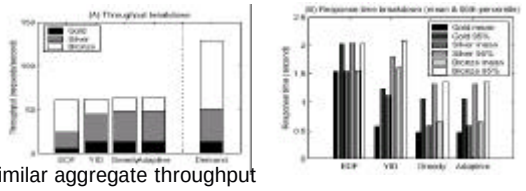


- EDF and YID perform better than Greedy during system under-load; Greedy performs better during system overload.
- Adaptive dynamically switches between YID and Greedy to achieve good performance under both situations.

Evaluation on Scheduling Policies (16 nodes aggregate) — on Differentiated Search

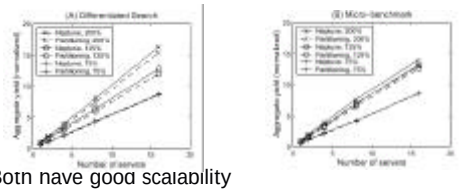


Per-class Performance Breakdown



- Similar aggregate throughput
- Greedy and Adaptive policies have more efficient resource utilization
- Greedy and Adaptive policies have better quality for competed requests

Performance and Scalability of Request Distribution Schemes

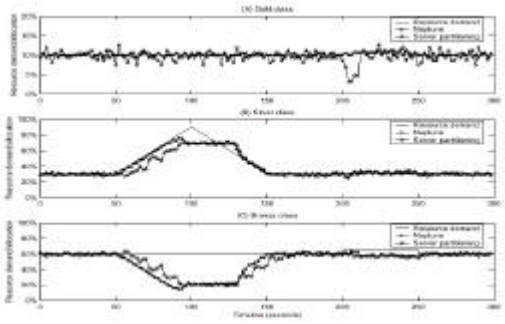


- Both have good scalability
  - underlying load balancing strategy
  - Random polling policy
- Under high overload, Neptune is better
  - Whole cluster-wide load balancing

Service Differentiation during a Demand Spike and a Node Failure (8 nodes)

- Experiment Conditions:
  - "Service yield"-based priority order: Gold > Silver > Bronze.
  - 20% proportional resource guarantee for low-priority Bronze class.
  - Demand spike for the Silver class between time 50 and 150.
  - One node fails at time 200 and recovers at 250.

Service Differentiation during a Demand Spike and a Node Failure Comparison



Next...

- Introduction (background and mainpoint)
- Architecture of Targeted Services
- Multiple Resource Management Objectives
- Two-level request Distribution and Scheduling
- Performance Evaluations on a Linux Cluster
- Related Work and the Conclusion

Related Work

- Software infrastructure for cluster-based Internet services – Scallability and availability issues, replicaton support
- QoS support and service differentiation– Network bandwidth allocation and packet delay.
- Resource management for clustered services
- Locality-aware request distribution
- Service scheduling – many schedulings have been studied in real-time system and general-purpose OS

Conclusion

- Design and implementation of an integrated resource manegement framework for cluster-based services with:
  - Multiple resource management objectives:
    - quality-aware resource utilization efficiency
    - service differentiation
  - Two-level resource management mechanism:
    - non-partitioning at the cluster level
    - adaptive scheduling at the node level
- Evaluations show that the proposed techniques can efficiently utilize system resources under quality constrains and provide service differentiation
- Only evaluated with read-only workload.