Quality of Service and Service Differentiation

Kai Shen
Dept. of Computer Science, University of Rochester

Service Qualities

- Services can be completed with varying qualities.
- Important qualities for network services
  - service response time (for almost all services)
  - relevance and completeness (for keyword searching)
  - graphics resolution (image composition or games)
  - anything else?
- Tradeoff
  - between resource demand and service qualities
    - for search relevance or completeness
    - for service response time
    - among different service qualities
      - search completeness vs. service response time

Tradeoff between resource demand and service qualities

- What do you do when overall demand exceeds the system capacity?
  - drops requests
  - complete requests with less quality (thus reduce demand)
- Objective: quality-aware resource utilization efficiency
  - how to choose from "drop requests" and "complete requests with less quality"?
    - i.e., "complete 80% requests with 100% quality" or "complete 100% requests with 80% quality"?
    - application-specific.
- Possible:
  - threshold: maximize the number of completed requests that meet a quality threshold
  - value-based: assign some amount of value for each quality level and try to maximize the overall value

An Example

- search completeness
  - threshold?
  - value-based?
- response time
  - threshold?
  - value-based?
**Service Differentiation**

- Scenario #1:
  - service with some paying users and some non-paying users
  - goal: allow priorities
- Scenario #2:
  - service with three groups, each paying equal amount
  - goal: proportional allocation
- Any other scenarios or goals?
- Additional goal:
  - leftover resources should be given to whoever needs it.

**An Example**

- priority?
- proportional allocation?
- remember that "leftover resources should be given to whoever needs it".

**Server Cluster**

- priority?
- proportional allocation?
- remember that "leftover resources should be given to whoever needs it".

Solution 1:
- let each server do it individually
- problem: uneven distribution at the cluster-level

Solution 2:
- partition the servers among groups
- problem:
  - can't deal with priority
  - no way to use leftover resources
  - fix: dynamic adjustment
Server Cluster: Summary

- Cons of the two-level approach
  - difficult to do even distribution at the cluster level - weak guarantee
  - weaker isolation

- Cons of the server partitioning
  - slow in adjusting to runtime dynamics