Ubiquitous Computing

• Anytime/Anyplace Computing
  - help organize and mediate social interactions wherever and whenever these situations might occur.

Key Characteristics

• Task dynamics
  - Applications have to adapt to the dynamics of users’ goals, environments and the resulting uncertainties.

• Device heterogeneity and resource constraints
  - Omnipresence is achieved by either making the devices move with the user or by having applications move seamlessly between devices tracking the user.

• Social issues
  - Introduction of a ubiquitous computing environment implies introduction of sensors, which inevitable have an impact on the social structure, e.g., privacy.

History and Evolution

• Mark Weiser, *Scientific American*, 1991

• Changing Forces
  - Wireless telecommunication capabilities
  - Open networks
  - Continued increases in computing power
  - Improved battery technology
  - Emergence of flexible software infrastructures

• Mobile Computing + Embedded Computing
  - Mobile computing: increasing the capability of moving computing services with us; computing model does not change
  - Embedded computing: increasing the capability of obtaining information from its environment and utilize it to dynamically build computing models, less level of mobility

Research Challenges

• Building the Flexible Software Infrastructure
  - capable of finding, adapting, and delivering the appropriate applications to the user’s computing environment based on user’s context

• Semantic Modeling
  - to describe the preferences of users and the relevant char. of computing components using a high-level semantic model

• Developing and Configuring Applications
  - Developed services should be easily and reliably reused by other developers and composed into larger applications

• Validating the User Experiences
  - development of effective methods for testing and evaluating the usage scenarios enabled by ubiquitous computing
Related Projects

<table>
<thead>
<tr>
<th>Project</th>
<th>Challenges being addressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aura (CMU)</td>
<td>Software infrastructure</td>
</tr>
<tr>
<td></td>
<td>Service construction and composition</td>
</tr>
<tr>
<td></td>
<td>User experience validation</td>
</tr>
<tr>
<td>Oxygen (MIT)</td>
<td>User experience enhance. and validation</td>
</tr>
<tr>
<td>Aware house (GaTech)</td>
<td>User experience enhance. and validation</td>
</tr>
<tr>
<td>Jini (Sun)</td>
<td>Context-based adaptation</td>
</tr>
<tr>
<td>PIMA (IBM)</td>
<td>Application development and deployment</td>
</tr>
<tr>
<td></td>
<td>Context-based adaptation</td>
</tr>
<tr>
<td>Condor (HP)</td>
<td>Service construction and composition</td>
</tr>
<tr>
<td></td>
<td>User experience enhance. and validation</td>
</tr>
</tbody>
</table>

Service Differentiation

- **The General Objective**
  - To provide predictable and controllable differentiated QoS levels to different request classes of clients
  - DiffServ vs. Best-effort
- **Perspectives**
  - On an indiscriminate Web site
    - Control behaviors of aggressive clients for fairness
  - On a Web content hosting site
    - Treat clients of various content providers differently
  - On a streaming site
    - Adapt to various devices, access patterns
  - On an Ecommerce site
    - Give higher priority to heavy buyers than visitors

Models and Properties

- **Models:**
  - Absolute differentiated services: clients receive an absolute share of resource usages; possible low resource utilization
  - Relative differentiated services: higher classes will receive relatively better (or no worse) QoS than lower classes
- **Properties:**
  - Predictability: differentiation schedules must be consistent, independent of variations of the class loads
  - Controllability: a number of controllable parameters adjustable for quality differentiation between classes
  - Fairness: lower classes not be over-compromised, especially when workload is low

State-of-the-Art (Taxonomy)

- **Server-side**
  - Admission control (ON/OFF $\rightarrow$ DiffServ)
    - Measure-based approaches [TC2], [INFOCOM01]
    - Queueing-theoretical approaches [SIGMETRICS99]
    - Control-theoretical approaches [TPDS02], [ICDCS01]
  - Resource allocation & scheduling
    - Processor, memory, and network [WWW99]
    - Priority-based request scheduling: strict priority [Network99], time-dependent priority [SIGMETRICS99]
    - Node partitioning: static vs. dynamic [ACM-PER01], [INFOCOM01]
  - OS support
    - New OS abstractions for resource management [OSDI99], [SIGMETRICS99]
    - Application-oriented mechanisms: QLinux [OSDI99], [SIGCOM99], [MM00]

- **Proxy-side**
  - Content adaptation: static, dynamic, streaming [WWW99], [INFOCOM00]
  - Cache management [ICDCS01], [ICDCS01]

- **Network-side**
  - Proportional delay differentiation
    - Rate-based packet scheduling: BPR [SIGMETRICS99], JFBS [WWW99]
    - Time-dependent priority packet scheduling: WTP [TON02], ATWTP [TON01], PAD & HPD [TON02], MDP [INFOCOM99]
  - Proportional loss differentiation
    - PLR [WWW99], JFBS [WWW99]

I: Diff. Streaming Services

- **Two-dimensional QoS metric**
  - Streaming bit rate and startup latency

An Integrated Approach

- **Static and dynamic content adaptation**
- **Two-level Schedulers**
  - Dispatcher-based scheduling
  - Server-based scheduling
Opportunity 1: Transcoding

- **Transcoding**
  Transform a continuous media stream on-the-fly from its original format to another one; from its original encoding bit rate to degraded streaming ones.

- **Two challenging issues**
  - When, and how to schedule streaming requests so as to provide predictable and controllable differentiated services in terms of streaming bit rate and startup latency at the same time.

Opportunity 2: Cluster Arch.

- **Cluster Architectures**
  - Inktomi (mid 2000)
    - 2 40 workstations
  - Hotmail
    - 60 web servers
    - 30 data servers

- **Two key issues**
  - How to statically and dynamically produce multiple versions of video objects with different encoding bit rates and where the replicas should be placed on the cluster?

Project Objectives

- **Objective I**
  - Develop an integrated framework for providing different QoS levels of streaming services to different clients
    - Streaming bit rate and Startup latency

- **Objective II**
  - Develop context-aware bandwidth allocation and distributed request scheduling algorithms
    - Dispatcher-based and server-based schedulers coordinate to determine when, where and how to process a request

- **Objective III**
  - Develop efficient and scalable static content adaptation and layout strategies
    - QoS abstractions, resource capacities, transcoding cost model

A General Optimization Model

- **In a cluster of** \( M \) **servers**
  - Given the \( N \) classified contending request classes with diff. weights \( w_i \), scheduling process is divided into a sequence of bandwidth allocation and scheduling intervals

- **Optimization Model**
  \[
  \text{Minimize} \quad F(w_i, \vartheta(b_1, s_i)) \\
  \text{subject to} \quad L_i \leq b_1 = G(\mu_i, \lambda_i) \leq U_i \\
  \sum_{i=1}^{N} \mu_i = \sum_{i=1}^{N} B_i \\
  \sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N} B_i < \sum_{i=1}^{N} U_i \cdot \lambda_i \\
  \vartheta(b_1, s_i) \geq \vartheta(b_2, s_i) \text{ iff } w_i \geq w_{i2}
  \]

A Client-ware Scheduling Strategy

- **In a single-server model**
  - One-dimensional QoS metric, two schedulers (FCFS, HoQ)
    - Objective: high bit rate streams to high priority clients

- **A Harmonic Proportional Model**
  \[
  \text{Minimize} \quad \sum_{i=1}^{N} w_i \cdot \frac{1}{b_i - 1} \\
  \text{subject to} \quad 1 \leq b_i = \mu_i / \lambda_i \leq U \\
  \sum_{i=1}^{N} \mu_i \leq B \\
  \sum_{i=1}^{N} \lambda_i \leq B < U \cdot \sum_{i=1}^{N} \lambda_i \\
  b_{i1} \geq b_{i2} \text{ iff } w_i \geq w_{i2}
  \]

Properties of the Results

- **A resource allocation problem**
  \[
  b_i = \frac{1}{\sum_{i=1}^{N} \sqrt{w_i / \lambda_i}} - \frac{(B - A) \sqrt{w_i / \lambda_i}}{\sum_{i=1}^{N} \sqrt{w_i / \lambda_i}} \quad (1 \leq i \leq N) \\
  \delta_i = \frac{\sum_{i=1}^{N} \sqrt{w_i / \lambda_i}}{\sqrt{w_i / \lambda_i}} - \frac{(B - A) \sqrt{w_i / \lambda_i}}{\sum_{i=1}^{N} \sqrt{w_i / \lambda_i}}
  \]

- **Properties**
  - Availability
  - QoS guarantees
  - Controllability
  - Predictability
  - Fairness
Proportional-share Allocation

- **Application level:**
  \[ \frac{b_i}{\lambda_i} = \frac{w_i}{\sum_{j=1}^{N} w_j} \quad (1 \leq i \leq N) \]

- **Allocation results**
  \[ \lambda_i = \lambda_i \left( 1 + \frac{(B - \lambda_i) w_i / \lambda_i}{\sum_{j=1}^{N} w_j} \right) \]

- **References:**

Simulation Model

**Allocation & Scheduling Schemes**

- **Allocation schemes**
  - Relative differentiation
    - Optimization-based (Optimal S.D.)
    - Proportional-share (Pro. S.D.)
  - Absolute differentiation (Static S.D.)
  - No service differentiation
    - Fixed 2 Mbps
    - Fixed 3 Mbps
    - Fixed 4 Mbps

- **Request scheduling**
  - First-Come-First-Served with First-fit backfill (FCFS/FF)
  - Head-of-Queue priority scheduling

Result I-1.1

- **Impact of Service Diff. on System Perf.**
  - On average rejection rate

Result I-1.2

- **Impact of Service Diff. on System Perf.**
  - On average queuing delay

Result I-1.3

- **Impact of Service Diff. on System Perf.**
  - On adaptive bandwidth utilization
### Result I-2.1

- Impact of S.D. Bw. Allocation Schemes
  - On minimization objective function

![Graph showing the impact of S.D. Bw. Allocation Schemes on minimization objective function.](image)

- Static S.D.
- Pro. S.D.
- Opt. S.D.

Arrivals per minute vs Minimization objective.

### Result I-2.2

- Impact of S.D. Bw. Allocation Schemes
  - A microscopic view of diff. streaming bit rate

![Graph showing the impact of S.D. Bw. Allocation Schemes on streaming bit rate.](image)

- Opt. S.D., class A
- Pro. S.D., class A
- Opt. S.D., class B
- Pro. S.D., class B
- Opt. S.D., class C
- Pro. S.D., class C

Recording time (minute) vs Streaming bit rate (Mbps).

### Result I-2.3

- Impact of S.D. Bw. Allocation Schemes
  - A long-term view of diff. streaming bit rate

![Graph showing the impact of S.D. Bw. Allocation Schemes on streaming bit rate.](image)

- Opt. S.D., class A
- Pro. S.D., class A
- Opt. S.D., class B
- Pro. S.D., class B
- Opt. S.D., class C
- Pro. S.D., class C

Arrivals per minute vs Streaming bit rate (Mbps).

### Result I-3

- FCFS vs HOQ Priority request scheduling

![Graph showing the comparison of FCFS vs HOQ Priority request scheduling.](image)

- FCFS: Opt. S.D., class A, class B, class C
- HOQ: Opt. S.D., class A, class B, class C

Arrivals per minute vs Queueing delay (minutes).

### Result I-4

- Impact of the Feedback Queue

![Graph showing the impact of the Feedback Queue.](image)

- Class A
- Class B
- Class C

Arrivals per minute vs Queueing delay (minutes).

### Conclusions on This Topic

- Parameterized bandwidth allocation with transcoding enables the streaming servers to efficiently and adaptively manage their resources.
- Service differentiation provisioning can help the streaming servers to achieve high service availability and maintain low queueing-delay.
- The proposed network/I/O bandwidth allocation schemes can achieve short-term & long-term objectives of service differentiation provisioning in terms of streaming bit rate.
- The feedback queue technique can reduce the queueing-delay significantly at a marginal cost of streaming bit rate.
- Priority scheduling can differentiate queueing-delay, but cannot guarantee the degree of differentiation.
Cluster Architectures

- Shared storage vs. Distributed storage
  - Flexibility, scalability, and availability

![Diagram showing shared storage and distributed storage]

Content Layout on Servers

- Dynamic transcoding vs. static adaptation
- Concerns
  - Computation cost of transcoding
  - Storage capacity
  - Disk-I/O bandwidth
  - Quality loss vs. reduction of resource consumption
- Content adaptation, replication and distribution
  - How many versions? And what encoding bit rates?
  - How many replicas for each version?
  - Where to place these replicas?
  - When and how to update the placement?

![Diagram showing content layout on servers]

BW-Constrained Placement

- Data striping vs. Data replication
  - Utilization, LB ability (access skew), scalability
  - Storage capacity

![Diagram showing data striping and replication]

Problem Formulation

- A scalable C-media distribution scheme with partial replication
  - Quality, availability, scalability

\[
\text{Maximize } \sum_{(i,j)} \frac{h_i}{M} + \alpha \sum_{(i,j)} \frac{r_i}{M - \beta} - I_i
\]

Subject to
\[
\sum_{(i,j)} h_i d_i \leq C
\]
\[
I_i = \sum_{j \neq i} \frac{\sum d_j h_j p_j}{r_i} \leq B
\]
\[
\pi(x_i) \neq \pi(x_j), \quad 1 \leq j, k \leq r_i
\]
\[
1 \leq i \leq n
\]

R&P Algorithms

- Fixed bit rate
  - Replication: a minimax optimization close to the classic apportionment problem
    - Splitting and Joining replication \(O(m + \log m)\)
    - Zipf’s distribution based replication \(O(m + \log m)\)
  - Placement: close to load balancing
    - Greatest load first placement – low bounded
  - References

![Diagram showing R&P algorithms]

R&P Algorithms (Cont.)

- Scalable bit rate
  - parSA: a parallel simulated annealing library

![Diagram showing scalable bit rate]

- References:
Result II-1.1

- Impact of rep. degree on rejection rate ($\theta=1.0$)

Result II-1.2

- Impact of rep. degree on rejection rate ($\theta=0.5$)

Result II-2.1

- Impact of layout algorithms on rejection rate

Result II-2.2

- Impact of layout algorithms on rejection rate

Result II-3

- Impact of data layout algorithms on LB

Request Redirection

- Motivation
  - Utilize internal bandwidth of the cluster backbone to balance outgoing network traffic

Reference:
II: Session-based DiffServ

- **Motivation**
  - A session-based workload gives a new interesting angle to revisit the definition and properties of DiffServ and its strategies
- **Inter-session Service Differentiation**
  - Different customers exhibit various navigation patterns
  - Heavy buyer class: 5% and 56% of them completed
  - Occasional buyer & visitor class: 95%
- **Intra-session Service Differentiation**
  - A session has a number of states
  - States have different probabilities to transit to the final buy state, and cost different resource usage on the average

A 2-Dimensional DiffServ Model

- Each dimension meets the basic SD properties
  - Sessions of class 1
  - Sessions of class 2
  - Sessions of class m

CBMG: Customer Behavior Graph

- **Response Time vs. Slowdown**
  - compare requests with different resource demands
  - reflect system load by themselves
  - clients expectations
- **A definition of slowdown**
  - Slowdown \( r_W = \text{waiting time in queue} / \text{service time} \)
  - Given a M/G/1 PS queue, Poisson arrival rate \( \lambda \), exponential service time \( \mu \). Let \( W \) be a request’s waiting time in queue, \( S \) be its slowdown, when \( \rho < 1 \) (\( \lambda < \mu \)), we have
  \[
  E[W] = \frac{\rho}{\mu} (1 - \rho) \\
  E[S] = E[W]/E[X^\mu] = \frac{\rho}{(1 - \rho)} = \frac{\rho}{\mu - \lambda}
  \]

Workload: \( m \times n \) M/G/1 PS Queues

- **Assumption**
  - Session arrivals of each \( m \) customer class: Poisson process
- **Observation**
  - At a state, multiple requests from a customer are dependent
- **Weak dependency deg, for each of \( m \times n \) queues**

Formulation of the 2D DiffServ

- **A 2D resource allocation problem**

  \[
  \text{Minimize } \sum_{i} \sum_{j} c_{i,j} \cdot s_{i,j} \\
  \text{subject to } \sum_{j} \sum_{i} c_{i,j} \cdot s_{i,j} \leq C_i \\
  0 < s_{i,j} < \frac{\lambda_i \cdot d_{i,j}}{c_{i,j}}
  \]

  - **Optimal allocation scheme**

  \[
  s_{i,j} = \frac{\lambda_i \cdot d_{i,j}}{C_i - \sum_{k} \sum_{i} \lambda_k \cdot d_{k,j}} \\
  \]

- **Reference**
  - X. Zhou, J. Wei, C.-Z. Xu, “Modelling and analysis of 2D service differentiation on an e-Commerce server”, submitted to ICS04
Simulation Model

- Implementation Issues
  - Session arrival rate
  - Number of visits at a state in a session
  - Resource demand of a state on the average
  - Resource allocation (lottery scheduling)

Result III-1

- System slowdown

Result III-2.1

- 2D Service Diff. in slowdown

Result III-2.2

- A Microscopic view of 2D S.D. (Opt.)

Result III-2.3

- A Microscopic view of 2D S.D. (Opt.)

Result III-3.1

- A long-term view of S.D. (Opt.)
Result III-3.2

A long-term view of S.D. (Pro.)

III: Network DiffServ

Network-side
- Proportional delay differentiation
  - Rate-based packet scheduling: BPR[SIGCOM99], JoBS[IWQoS01]
  - Time-dependent priority packet scheduling: WTP[TON02], AWTP[TON01], PAD&HP[TON02], MDP/INFOCOM00
- Proportional loss differentiation
  - PLR[IWQoS00], JoBS[IWQoS01]

Reference
- X. Zhou, J. Wei, C.-Z.XU, “Processing rate allocation for proportional slowdown differentiation on single-server systems”, to be submitted to IEEE/IFIP04

Research Plan

DDoS Overview

Goal and Opportunities

Goal
- design effective admission control mechanisms with adaptive resource management mechanisms at the server side to defend emerging degrading DDoS attacks
  - control the access of each client to server resources based on its behavior
  - guarantee fair QoS to legitimate well-behaving clients

Opportunities
- Service Differentiation techniques
  - Measurement-based admission control
  - Adaptive resource management for QoS adaptation
Time sheet

- 9/24/2003 – 10/21/2003: Develop a measurement-based admission control mechanism based on clients’ behavior & profile
- 11/12/2003 – 12/24/2003: Evaluate performance of the proposed techniques; analyses results and generate research reports & papers

Questions?