Business Oriented Autonomic Load Balancing for Multitier Websites

Based on:

Business-Oriented Autonomic Load Balancing for Multitiered Web Sites, J. ***Ewing*** and D.A. ***Menasce***, 17th ACM/IEEE Intl. Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2009), London, UK, September 21-23, 2009

Presented By:

Kaveh Razavi

Motivation

- Context
  - Complex environment like large e-commerce websites
  - Workload varies widely and is hard to predict
  - Multitier architecture using load balancers and server clusters
- Q:
  - how to design the system so that it can regulate itself without human interaction?
- A:
  - Employ autonomic techniques for self-optimization and self-configuration
  - To maximize a business-oriented utility function
  - Through a 2-level policy adaptation of the load balancer
    - Dynamically changing ***request redirection*** policy
    - Dynamically changing ***resource allocation*** policy

Environment: Customer Model

- Three categories of customers in an e-commerce web site: **Gold**, **Silver**, and **Bronze**
- Workload generated by each category can be described using a **Customer Behavior Model Graph (CBMG)**
  - Directed arcs
    - represent possible transitions between
    - are labeled with corresponding transition probability
  - Sum of outgoing transition probabilities for each node adds up to one

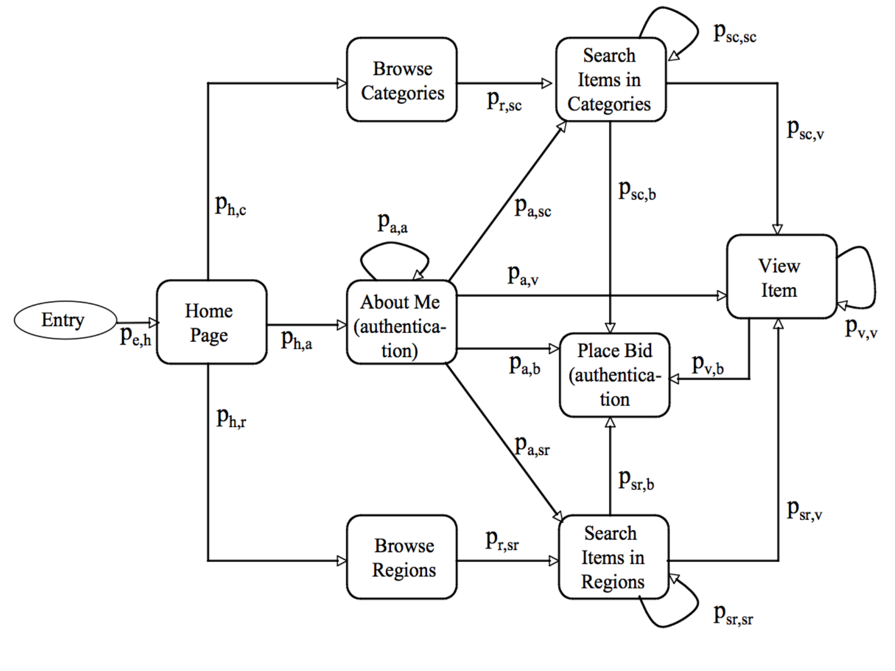


Fig. 2. Customer Behavior Model Graph (CBMG).

Environment: Multi-tiered Architecture

- $\mathbf{n_{ws}}$: number of web servers
- $\mathbf{n_A}$: number of application servers
  - $n_A = n_G + n_S + n_B$
  - n_X: varying number of servers in x = {gold, silver, bronze} customer server cluster
- Each server cluster services requests from
  - Its corresponding customer category
  - Customers from other categories according to redirection policy

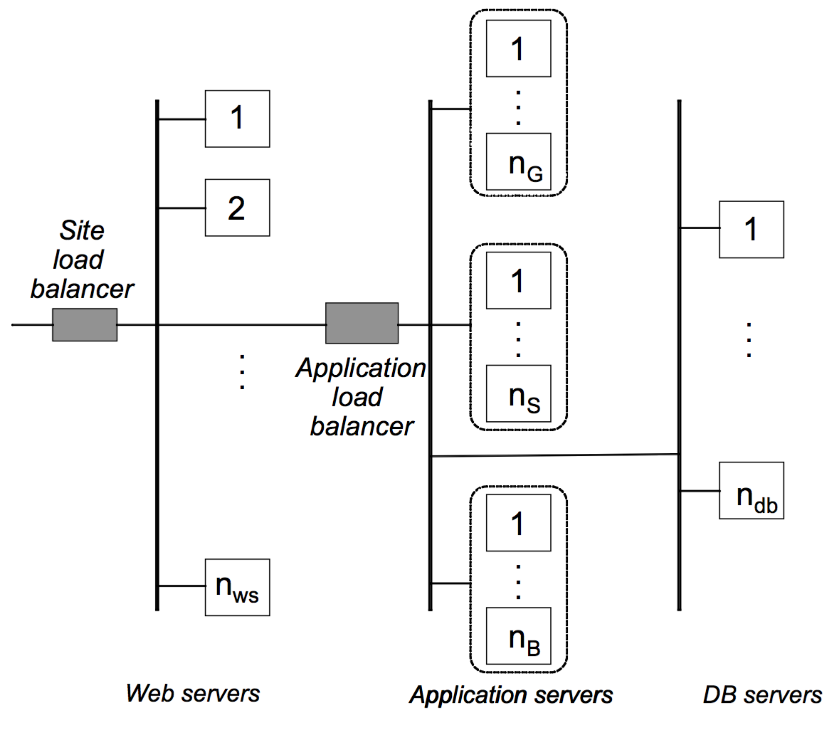


Fig. 1. Multi-tiered Environment

Utility Function: Response Time

- Good response times cause
  - Current customers continue using web site
  - New customers being attracted to web site by favorable impressions
- Represents likelihood of customers returning tomorrow
- Models whether utility generated by complying with Response Time Service Level Objective (SLO)
  - ***s***: priority class (gold, silver, bronze)
  - ***β_s***: average response time SLO of class s
  - ***R_s***: average response time of all class s transactions
  - ***w_s***: weight of class s
  - ***R vector***: vector of average response times for classes

$$U_s^R(R_s) = \frac{e^{-R_s+\beta_s}}{1+e^{-R_s+\beta_s}}$$

$$U_{total}^R(\vec{R}) = \sum_{\forall\ s} w_s \times U_s^R(R_s)$$

Utility Function: Throughput

- Throughput of specific revenue-generating transactions (bids)
- Represents how much money is being made today!
- ***X vector***: vector of average bid throughputs for classes

$$\boxed{X_{bid}(\vec{X}) = \sum_{\forall\ s} X_{bid,s}}$$

Utility Function: Global

- Goal is to maximize both Response Time and Throughput functions in a way that
  - Maximizes revenue today
  - While ensuring most important customers are satisfied and will return in future

$$\boxed{U_g(\vec{R}, \vec{X}) = X_{bid}(\vec{X}) \times U^{R}_{total}(\vec{R})}$$

Policies: f-policy & s-policy

- f-policy
  - A re-direction policy
  - Affects dispatching of requests to server clusters
  - Vector $\vec{f} = (f_{S,G}, f_{B,S}, f_{B,G})$
  - $f_{i,j} \in [-1, 1]$: fraction of requests from priority class i to cluster j
  - The autonomic LB dynamically adjusts the f-policy $\vec{f}$ to maximize the global utility U_g
  - $f_{S,G}$ and $f_{B,G}$ must carry the same sign to avoid cycles
- s-policy
  - A resource allocation policy
  - Determines how many servers should be allocated to each cluster
  - Vector $\vec{n}_A = (n_G, n_S, n_B)$

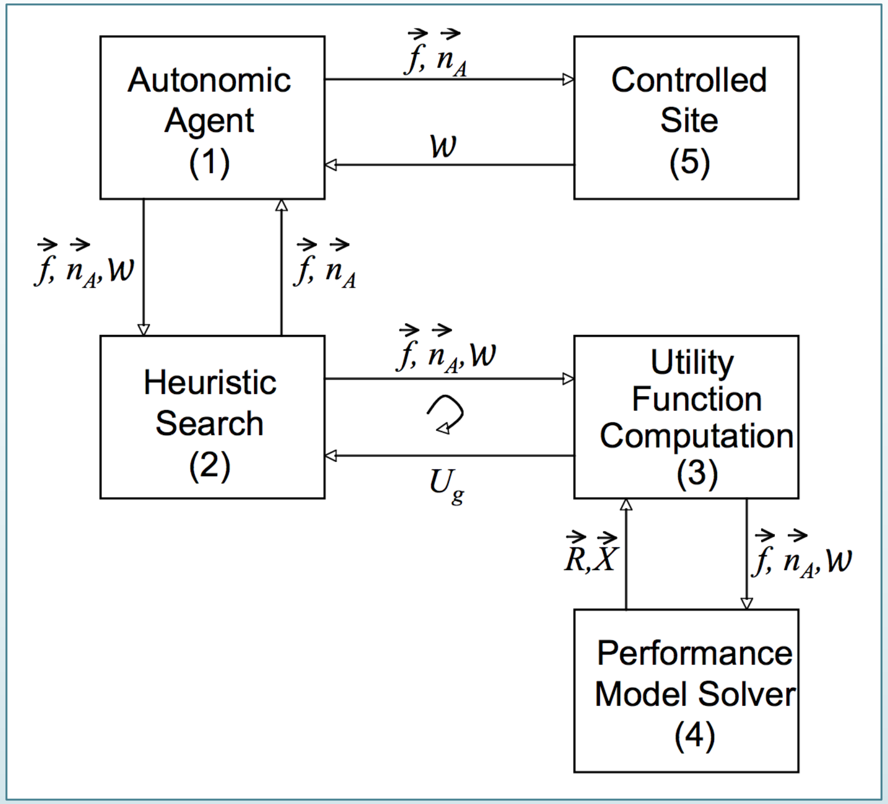
State Space

- The state space S of all possible configurations
- The three last constraints say no more than 100% of requests initially directed to one cluster can be redirected
- Is typically very large

$$\begin{aligned}
\mathcal{S} \quad = \quad & \{\vec{s} = (f_{S,G}, f_{B,S}, f_{B,G}, n_G, n_S, n_B) \mid \\
& n_G, n_S, n_B \in \{1, 2, \cdots, N_A - 2\}, \\
& n_G + n_S + n_B = N_A, \\
& f_{S,G}, f_{B,S}, f_{B,G} \in [-1, 1], \\
& f_{S,G} \times f_{B,G} \geq 0, \\
& \epsilon(f_{S,G})|f_{S,G}| + \epsilon(f_{B,G})|f_{B,G}| \leq 1, \\
& (1 - \epsilon(f_{S,G}))f_{S,G} + \epsilon(f_{B,S})|f_{B,S}| \leq 1, \\
& (1 - \epsilon(f_{B,S}))f_{B,S} + (1 - \epsilon(f_{B,G}))f_{B,G} \leq 1\}.
\end{aligned}$$

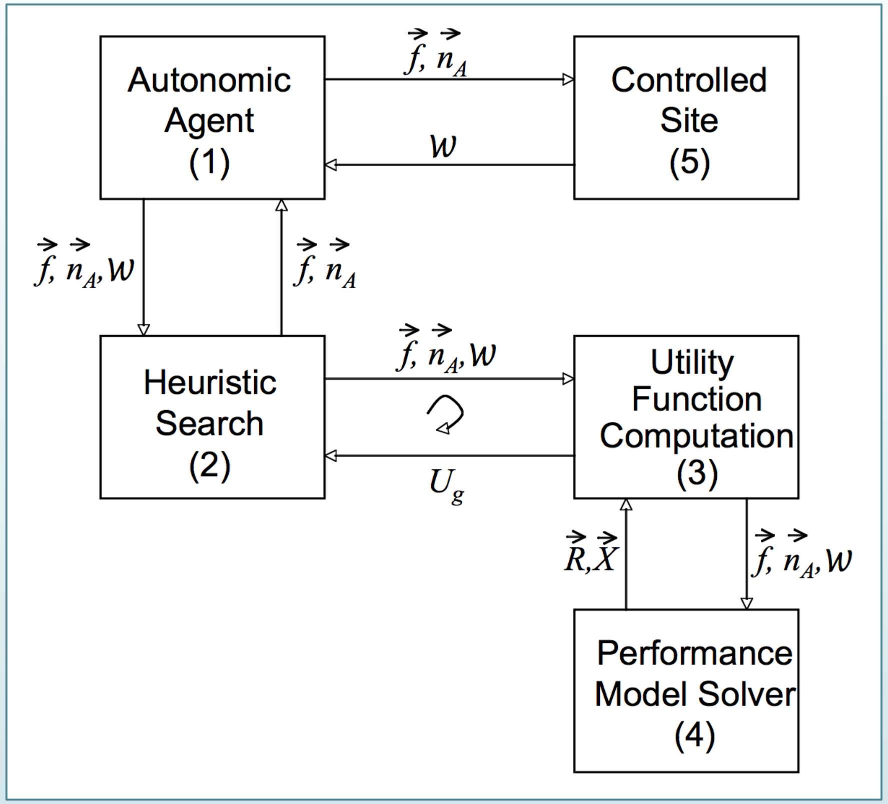
Autonomic LB Controller

- Redirects a fraction of incoming requests from one cluster to another cluster according to the f-policy
- Sends requests to servers within a cluster in a round-robin fashion
- May need to move one or more servers from one cluster to another to comply with a new s-policy
- Consists of 5 elements
- Time is divided into 30-second time intervals called controller intervals (CIs)



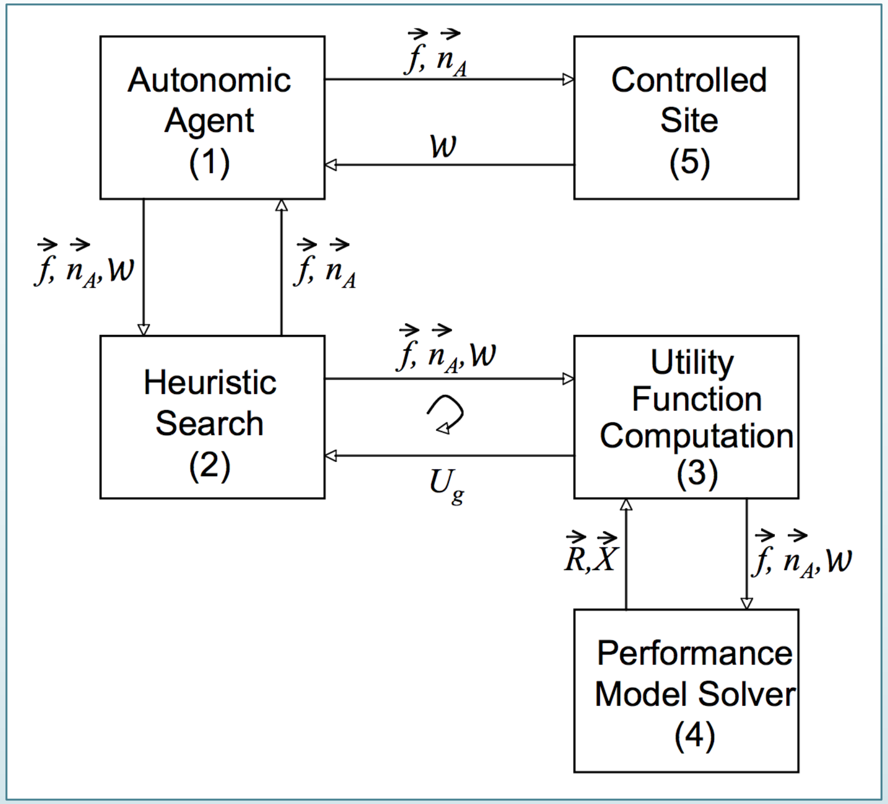
Autonomic Agent Component

1. Receives a description of workload intensity from last CI
2. Invokes heuristic search module
   - Every 1 CI to find optimal f-policy
     - Routing tables are changed
   - Every 10 CIs to find optimal s-policy
     - Because of switching costs
     - Affected clusters are informed



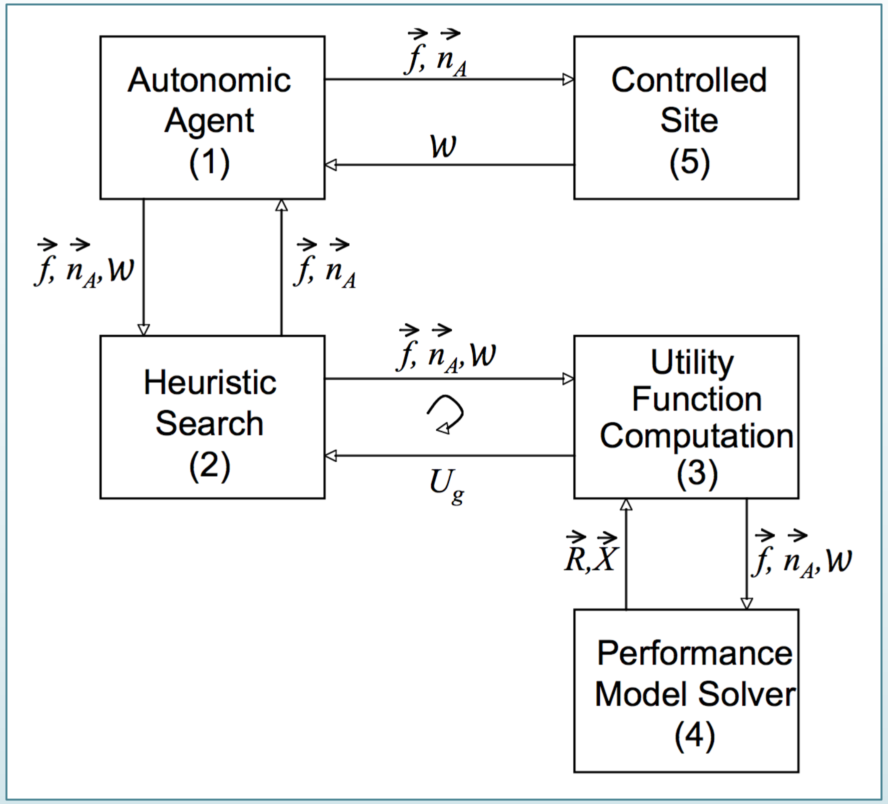
Heuristic Search Component

1. Is given current state and workload intensity
2. Is given an evaluation budget
   - $B_{f,max}$ when asked for optimal f-policy
   - $B_{s,max}$ when asked for optimal s-policy
3. Hill climbing is used as the heuristic to search state space
   - Using standard optimization techniques is not an option for real-time policy change
   - neighboring states of state $(f_{S,G}, f_{B,S}, f_{B,G}, n_G, n_S, n_B)$ are states of the form:
     - Optimal f-policy: $\vec{s}\,' = (f_{S,G} \pm \delta_f, f_{B,S} \pm \delta_f, f_{B,G} \pm \delta_f, n_G, n_S, n_B)$
     - Optimal s-policy: $\vec{s}\,' = (f_{S,G} \pm \delta_f, f_{B,S} \pm \delta_f, f_{B,G} \pm \delta_f, n_G \pm \delta_s, n_S \pm \delta_s, n_B \pm \delta_s)$
   - Uses Utility Function Computation unit to evaluate utility of a solution



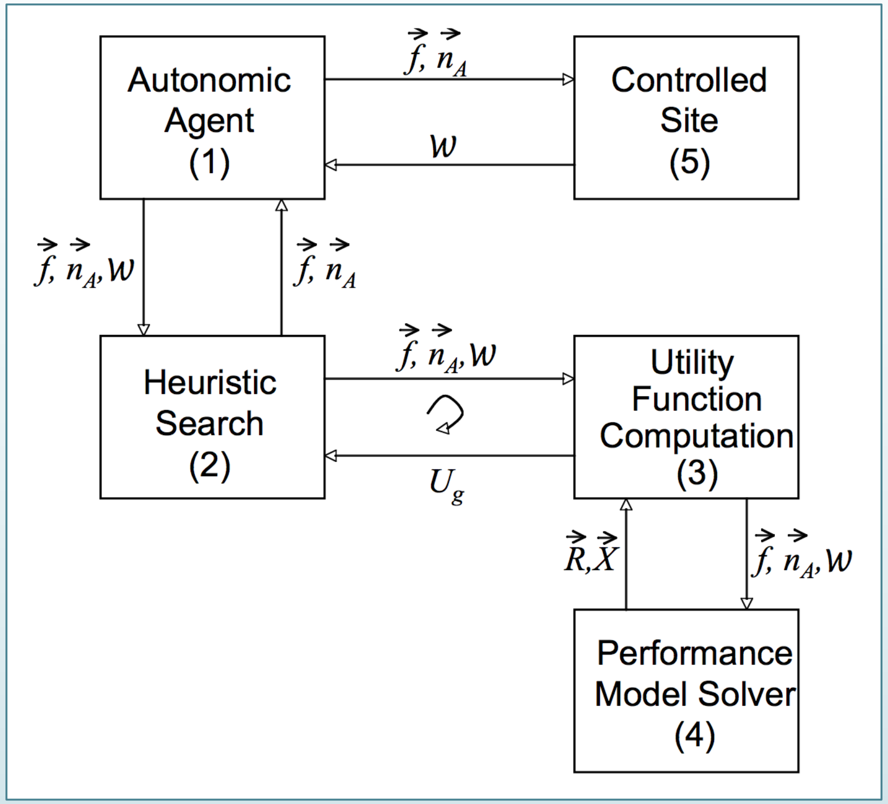
Utility Function Computation

- Invokes the performance model solver component to determine the expected response time and bid throughput for each priority class and then uses the global utility equation.



Performance Model Solver Component

- Uses analytic multiclass closed queuing network (QN) model
  - Solved using Approximate Mean Value Analysis (AMVA) technique

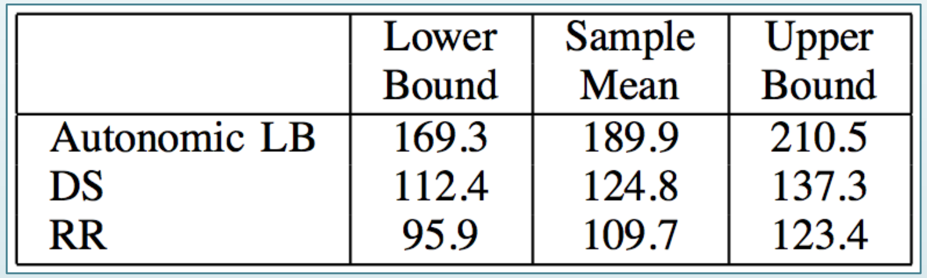


Evaluation

- Autonomic LB is compared with 2 other approaches: round robin (RR) and dedicated server (DS)
  - In all approaches round-robin load dispatching is used at web and DB tier and within each application tier cluster
  - Approaches differ in dispatch of requests to application tier clusters
    - RR: dispatched without regard for customer category
    - DS: always dispatched to cluster corresponding to customer category
  - Each approach was tested against 50 randomly generated loads, each with a duration of 480 minutes
    - Included extreme phenomena like flash crowds

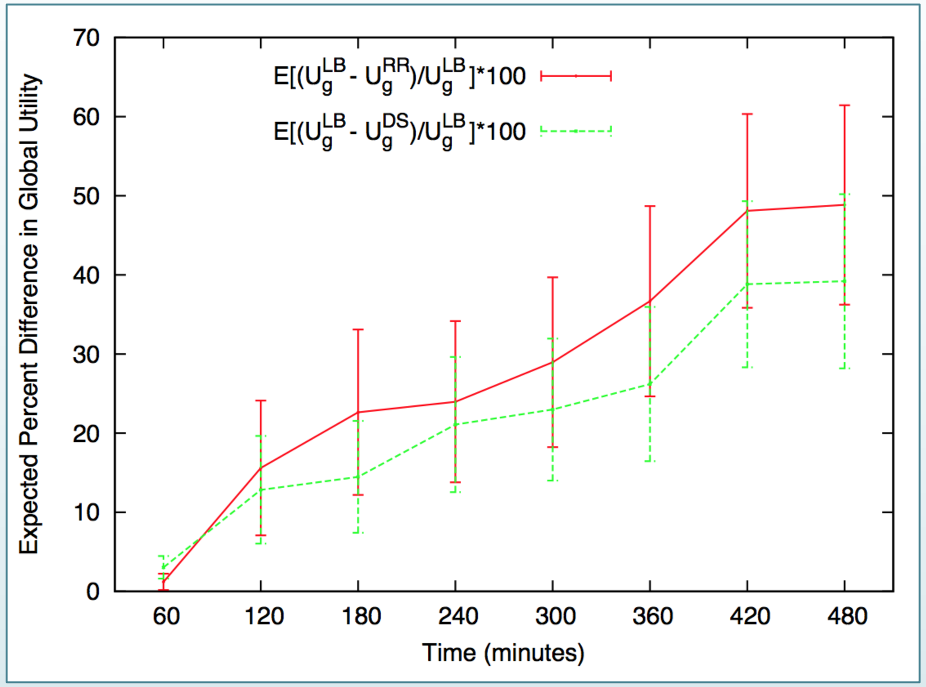
Evaluation

- All three confidence intervals are clearly separated
- The autonomic LB generated significantly higher global utility than either the DS or RR approaches



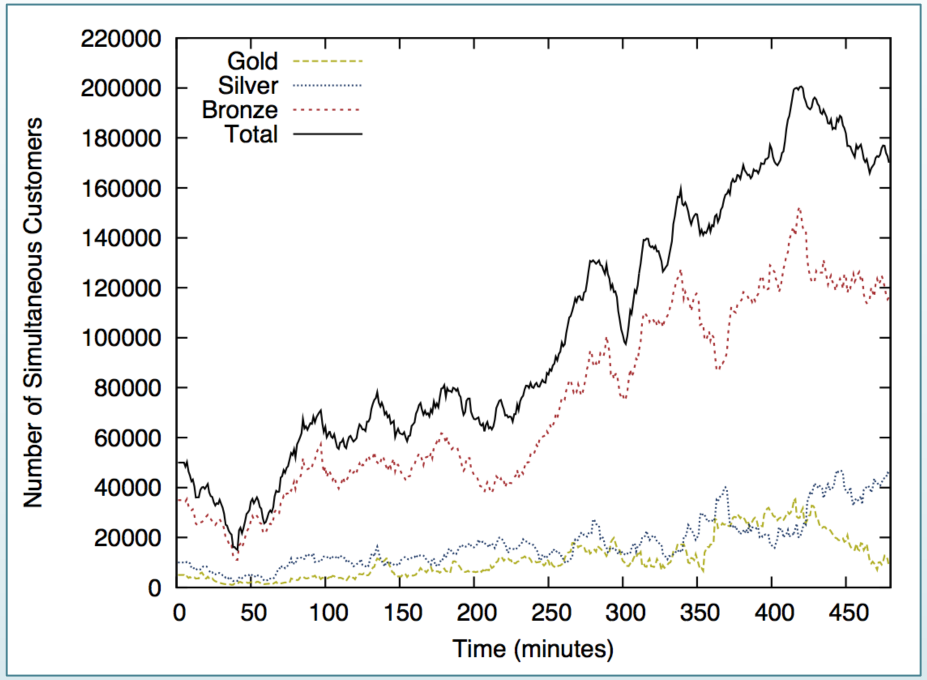
	Lower Bound	Sample Mean	Upper Bound
Autonomic LB	169.3	189.9	210.5
DS	112.4	124.8	137.3
RR	95.9	109.7	123.4

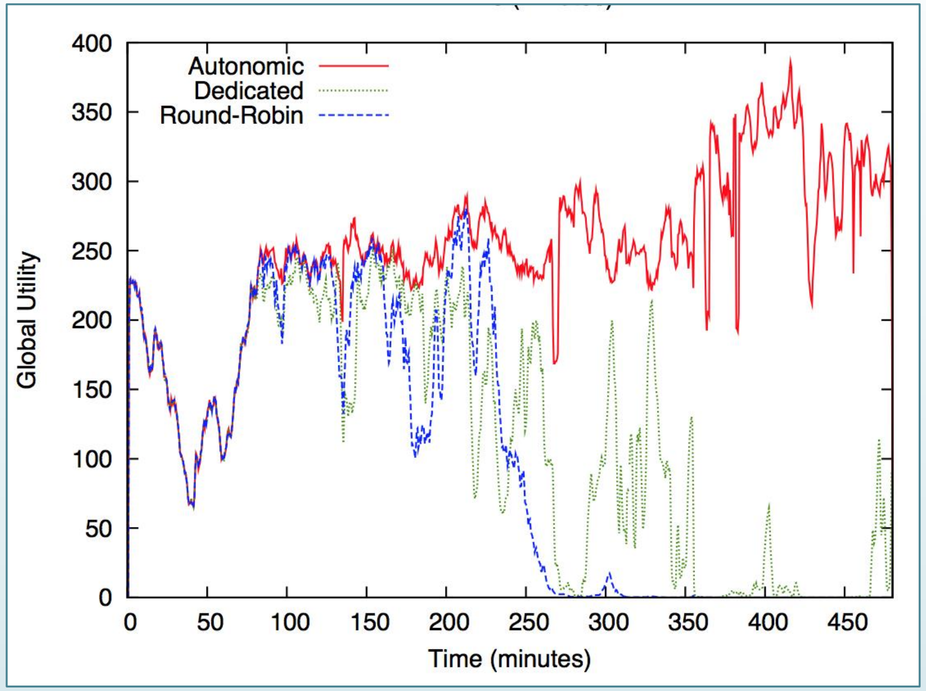
99% confidence interval for mean global utility



Expected percent difference in U_g between autonomic LB and RR and DS with 95% confidence interval

Evaluation





Load over time results from test 46

Global utility over time results from test 46

Conclusion

1) Autonomic LB matches the performance of the DS and RR approaches at low load levels,
2) Autonomic LB generates substantially more utility than DS or RR at higher load levels,
3) Autonomic LB shows a significant utility benefit at the 99% confidence level over the DS and RR approaches, and
4) Detailed examination of results shows that the autonomic controller redirects requests and allocates resources in a manner that maximizes bid throughput while minimizing response times.

Future Work

- Autonomic LB could be extended:
  - To handle server failures by allowing the total number of application servers to vary.
  - To improve energy efficiency in the data center by suspending or shutting down servers:
    1) relaxing the policy space constraint $n_G + n_S + n_B = N_A$ to $n_G + n_S + n_B \leq N_A$,
    2) adding an expression for the utility savings of a shutdown server,
    3) adding a utility expression for the switching cost and
    4) adding a utility expression for the resource shortage risk involved in shutting down a server.