REINFORCEMENT LEARNING

University of Nebraska Lincoln

CSCE 990: Advanced MAS

January 11, 2013

Human RL

Resource Allocation

- Fundamental problem in CS applications
  - Given a set of resources with limited quantities, how to apply to various tasks?
  - Goal: maximize benefits and/or minimize costs
    - Reward Tradeoff

- Applications:
  - CPU load
  - Memory management
  - Power consumption
  - Access to network connections

Internet Data Centers

- Resource Allocation Problem
  - How to assign available servers to incoming user requests?
  - Goals
    - Meet Service Level Agreements across several applications
    - Tradeoff responsiveness with power consumption savings
      - More servers = faster response time
      - Less servers = less power consumed

Overview

- Background
- Hybrid Reinforcement Learning for SLAs
- Power Savings
- Conclusion

Based on: (Tesauro *et al.*, 2007; Das *et al.*, 2008)

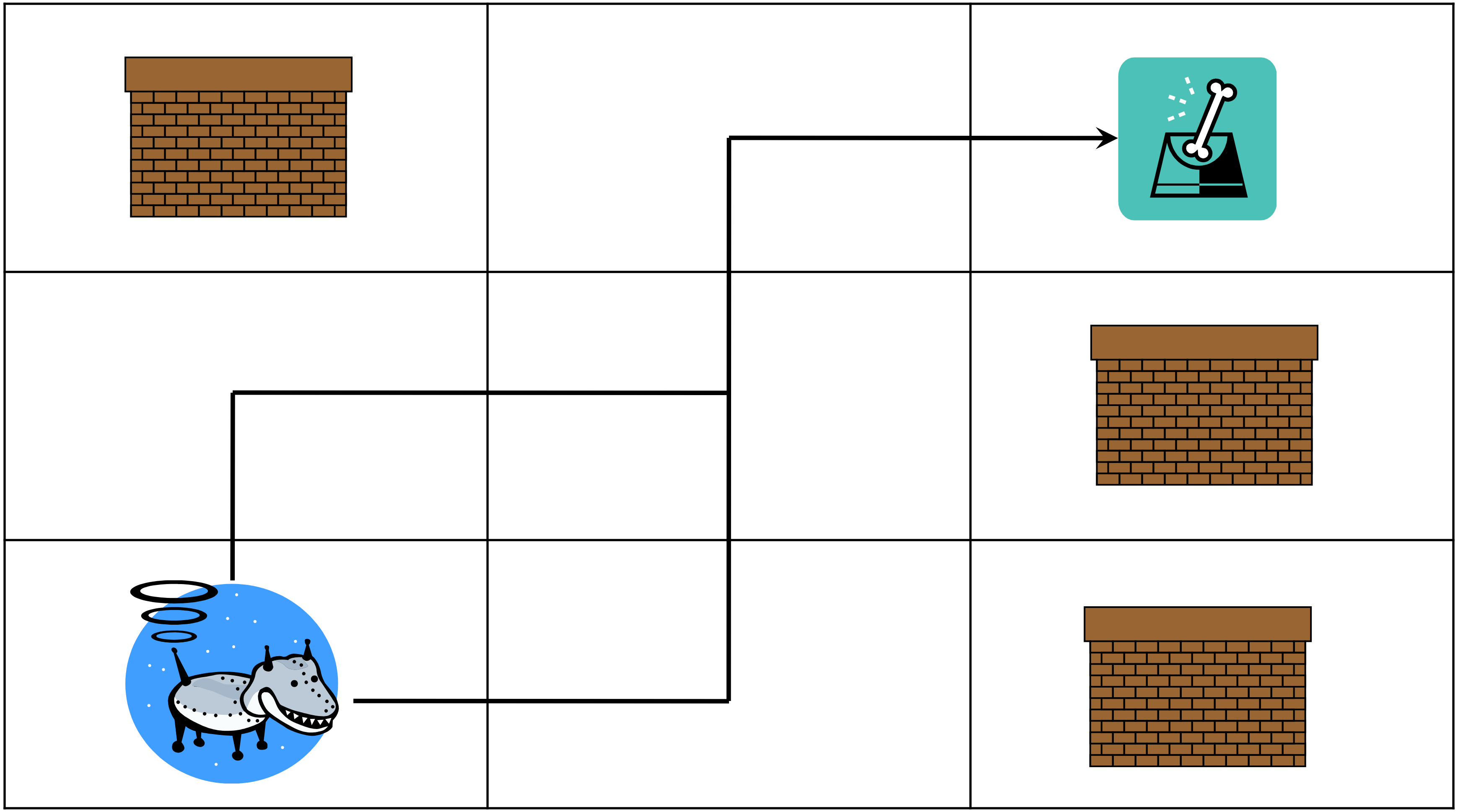
Background | Overview

- Reinforcement Learning
- Neural Networks
- Multiagent Systems

Background | Reinforcement Learning

- Problem
  - Learn a mapping of state/action pairs to utility values
  - Use learned utilities to form policies
    - Plans of actions maximizing utility
    - Underlying MDP model
- Terms
  - States *S*– description of environment
  - Actions *A*– action taken to change environment
  - Reward *R(s,a)* – numeric result of action

Background | Reinforcement Learning



Background | Reinforcement Learning

- Utility estimation stored as a Q-table

	Actions
States	Utility Values $Q(s,a)$

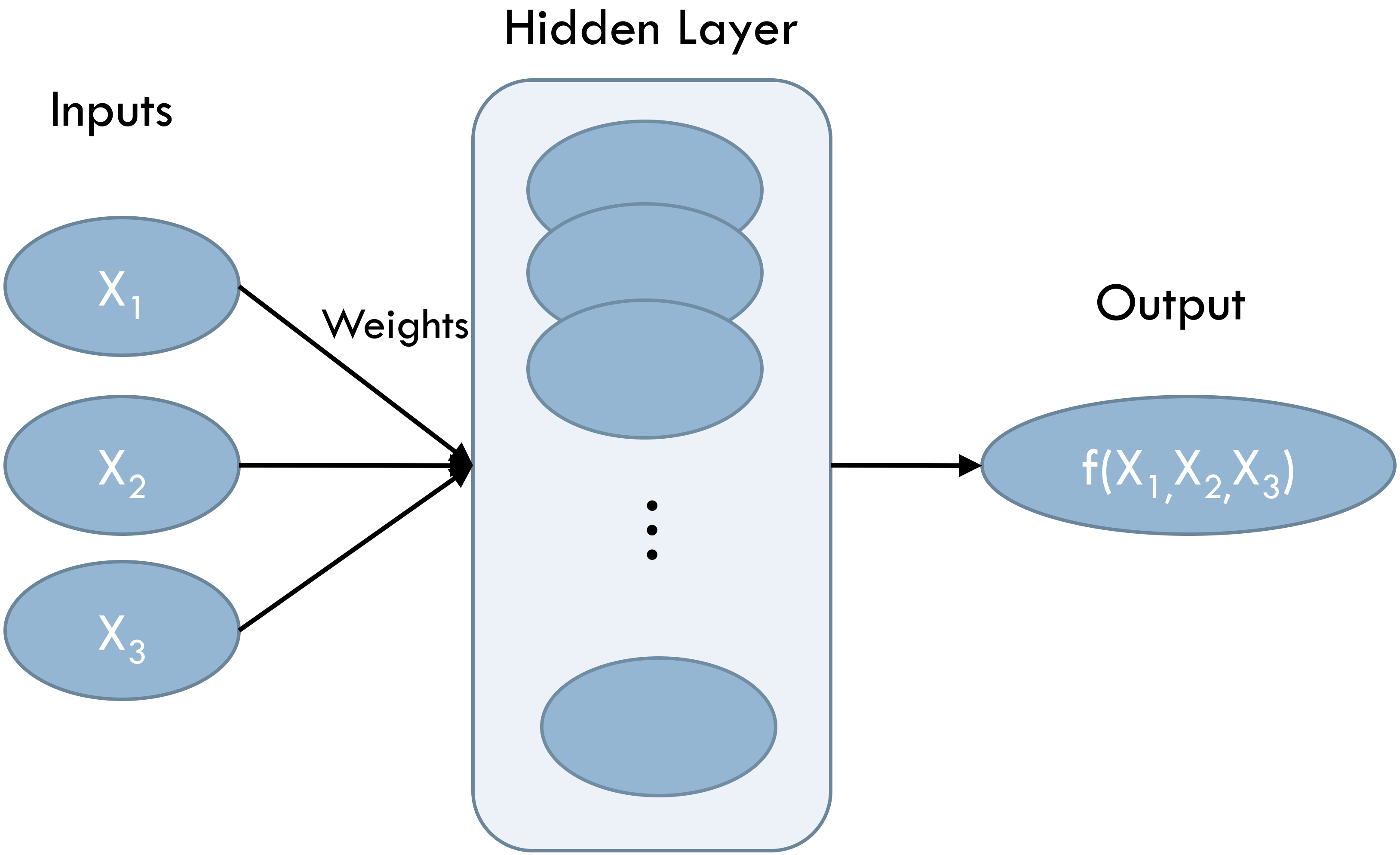
- Utility updates (SARSA):
  - $Q'(s,a) = (1 - \alpha)Q(s,a) + \alpha\,[R(s,a) + \gamma Q(s',a')]$

Background | Neural Networks

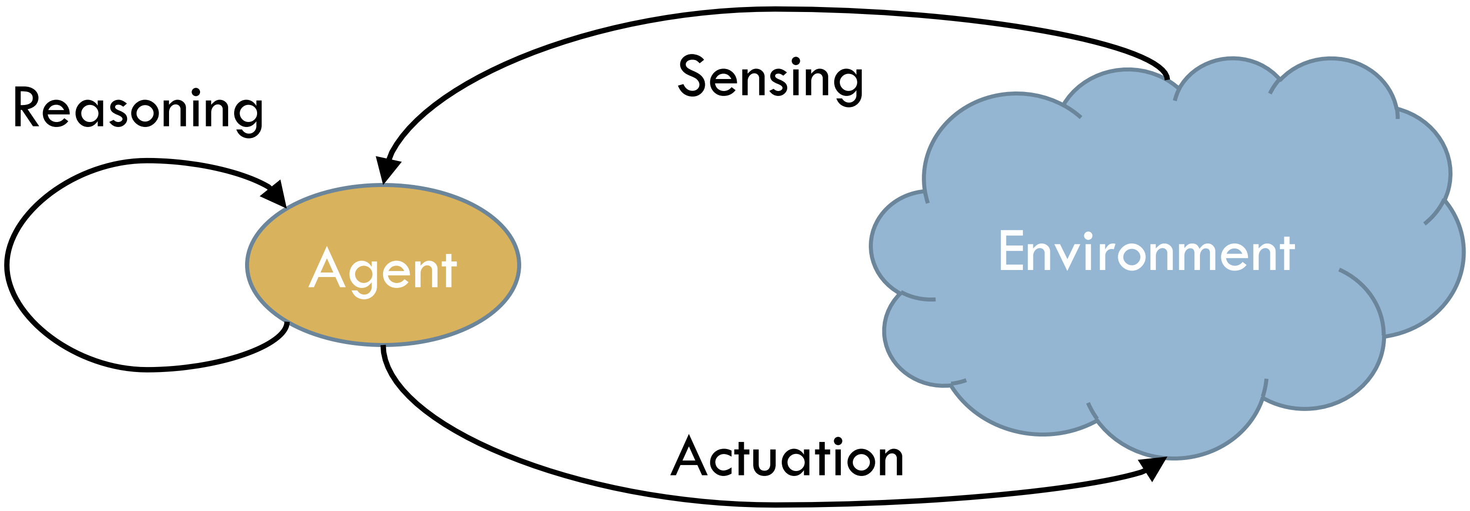
- Problem
  - Function approximation
    - non-linear output based on linear pieces (perceptrons)
  - Trained using examples (supervised learning)

- Often used in classification in Machine Learning
  - Continuous output
  - Discrete output by thresholding

Background | Neural Networks



Background | Multiagent Systems



- Agent Characteristics
  - Intelligent
  - Autonomous

Background | Multiagent Systems

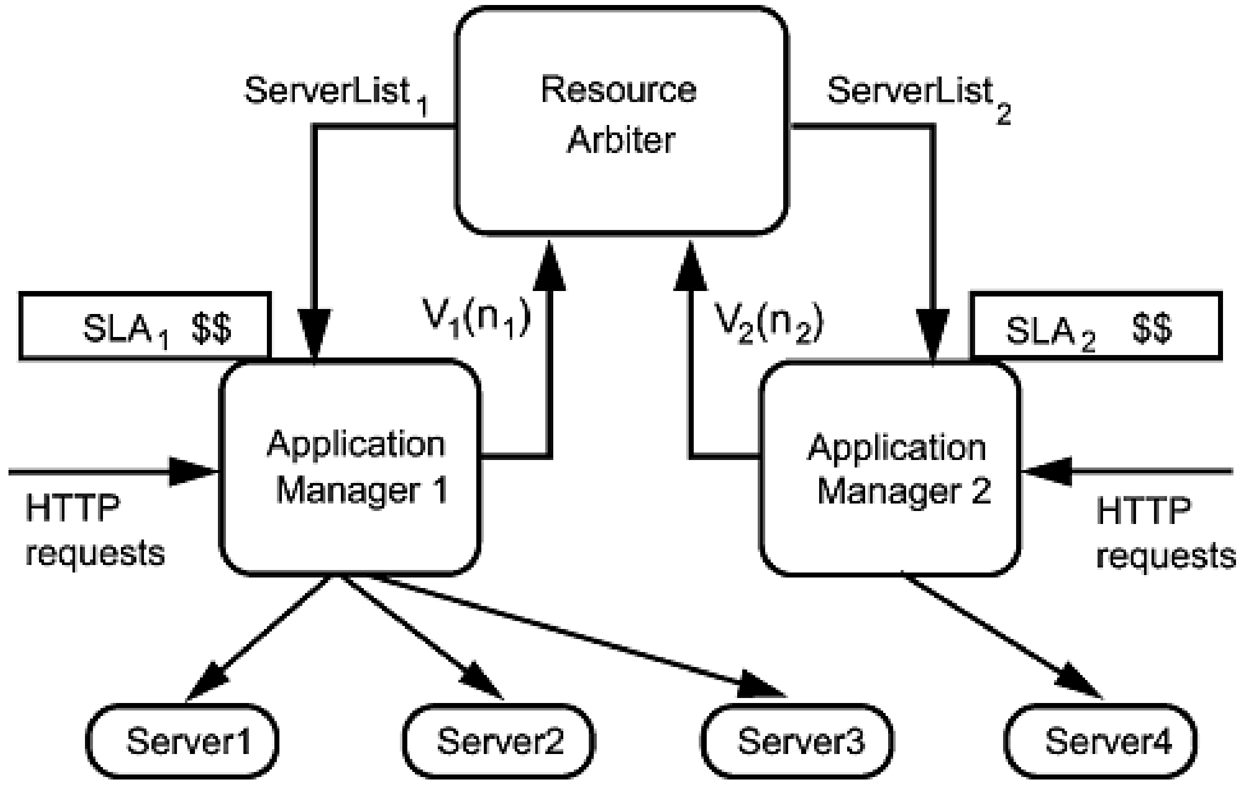
- Multiagent System (MAS)
  - Multiple agents in the same environment
    - Agents are aware of one another
    - Cooperative vs. Competitive
- Environment Characteristics
  - Dynamic
  - Uncertain/Noisy
  - Influenced by each agents

Hybrid RL| Overview

- Problem: how to allocate servers to various web applications?
  - Goal: maximize SLA revenue
    - Appropriate Response Time
  - Desired: self-managing system

- Approaches
  - Queue-based models
  - Reinforcement Learning
  - Hybrid Reinforcement Learning

Hybrid RL| System



Source: (Tesauro *et al.*, 2007)

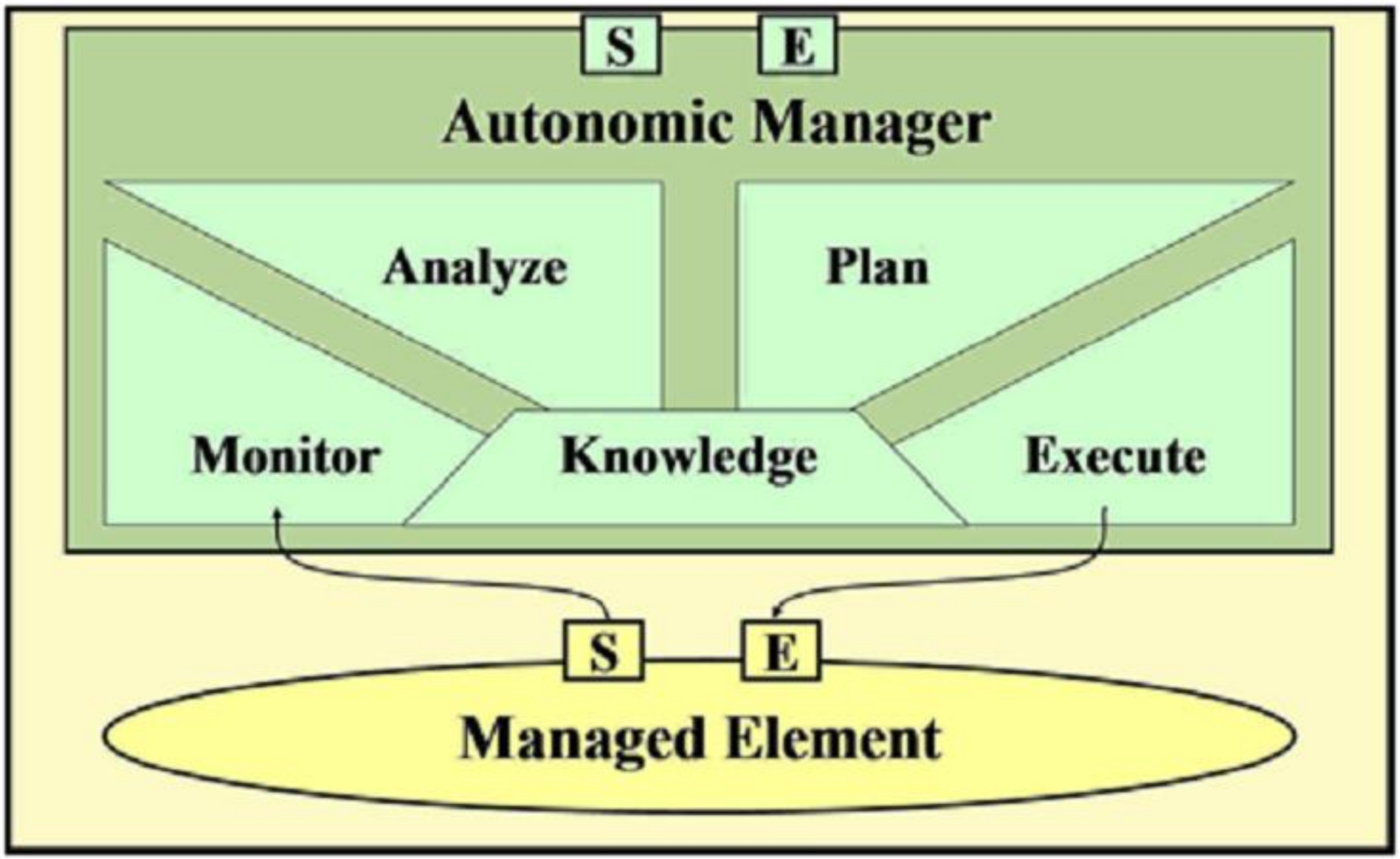
Hybrid RL | Resource Allocation

- Resource Arbiter assigns resources to Application Managers
  - Global decision based on distributed input

- Resource Arbiter's Decision Process
  - Get utility functions from each Application Manager
  - Choose allocation that maximizes total SLA revenue
    - Possibly sub-optimal for individual apps
    - Polynomial time?

- Application Managers model utility function

Hybrid RL| Application Manager



Source: (Tesauro *et al.*, 2007)

Hybrid RL | Queue-based Models

- Model each application as a Queue
  - Parallel homogeneous servers
  - Parameters
    - Request rate from users
    - Response time
    - # of servers
    - # of users
- Open-loop: infinite users
  - Steady incoming requests
- Closed-loop: fixed pool of users
  - Users "think", then submit request

Hybrid RL | Reinforcement Learning

- Model each application as a MDP
  - States = mean arrival of requests (λ)
  - Actions = number of servers *n* to assign
  - Reward = SLA revenue

- Problems
  - State space grows exponentially with characteristics
  - Exploration during online learning costly

Hybrid RL| Hybrid RL Approach

- State space problem
  - Use function approximation in place of Q-tables
    - **Neural networks**, regression trees, SVMs, etc.
    - Linear state space growth
  - Generalize across unseen states/actions

- Cost of exploration problem
  - Offline learning using server traces
    - Initial policy based on reasonable queuing model
  - Simulated or actual runs

Hybrid RL| Approach Comparison

- Queue-based model
  - Simple model (few parameters, no learning)
  - Standard practice
  - Cannot handle dynamic changes
  - Requires domain expertise

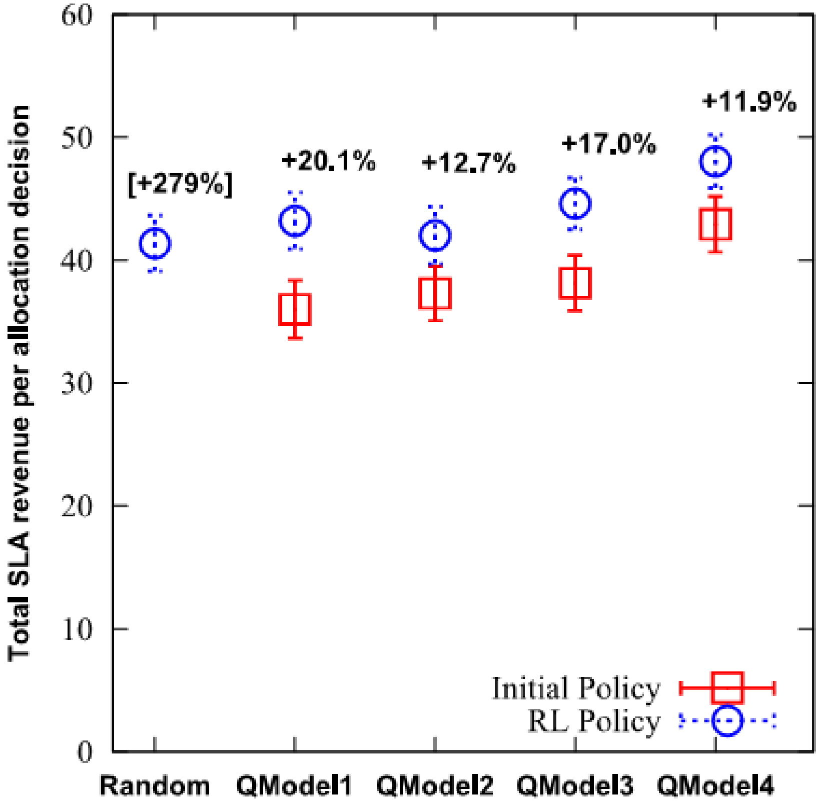
- Hybrid RL
  - Improves model over time through learning
    - No worse if starting with a decent initial policy
  - Can handle dynamic environment
  - Minimal domain knowledge necessary
  - Requires training
    - Expensive if no available traces

Hybrid RL | Experiment Setup

- Environment
  - 8 servers
  - 3 applications
    - 2 based on Trade3, an electronic trading simulation
    - 1 batch processing

- Approaches
  - Random policy
  - Queue-based models
  - Hybrid RL from Queue data

Hybrid RL| Closed Loop Results



- QModels:
  1. MVA with no smoothing
  2. (4) with cumulative reward
  3. Parallel M/M/1
  4. MVA with smoothing

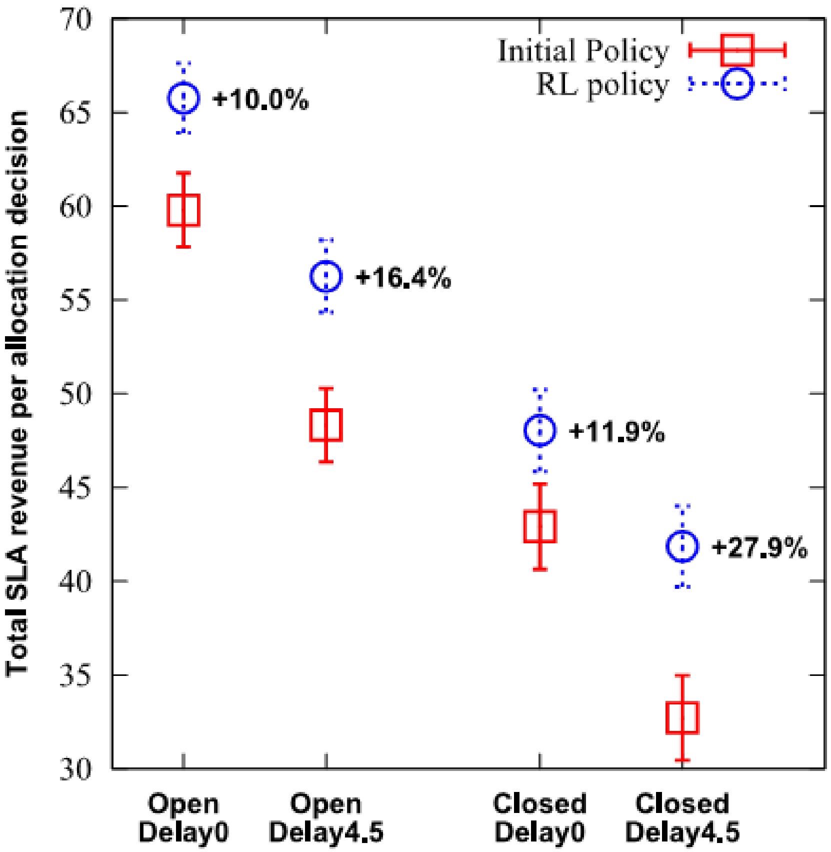
Source: (Tesauro *et al.*, 2007)

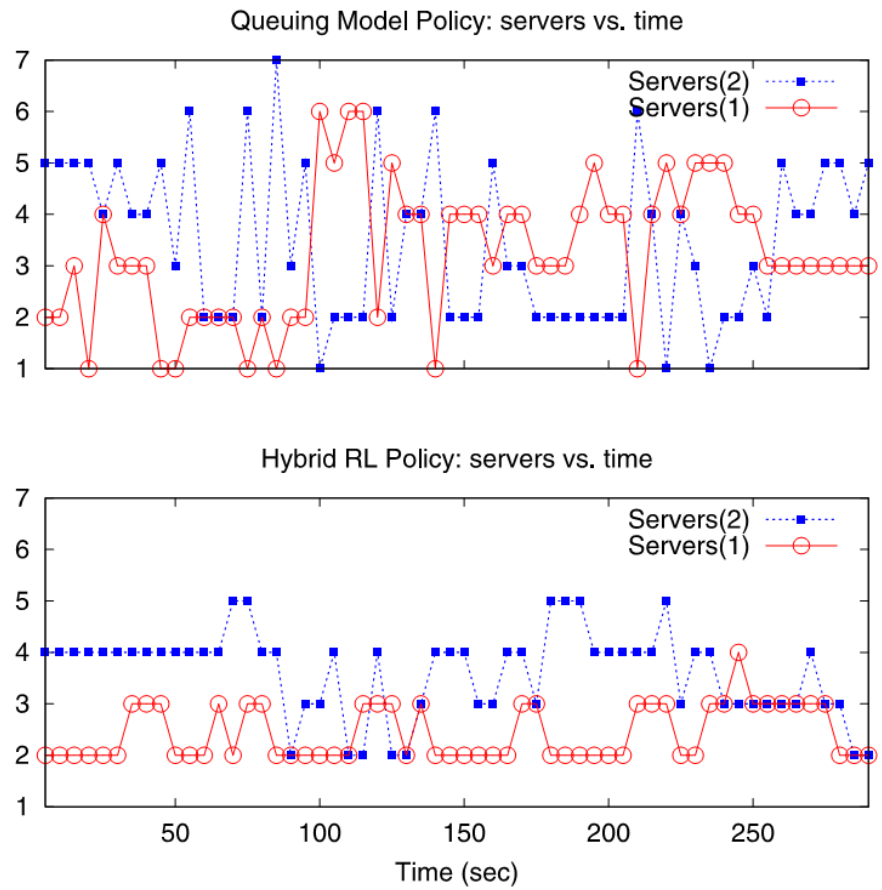
Hybrid RL | Hysteresis

- Dynamic Environment (not in Queuing model)
  - Impact of reassigning servers

- 4 Causes
  - Switching delays
  - Transient period of suboptimal performance
    - Starting processes
    - Load balancing
  - Temporary increased demand
  - Thrashing

Hybrid RL | Hysteresis Results





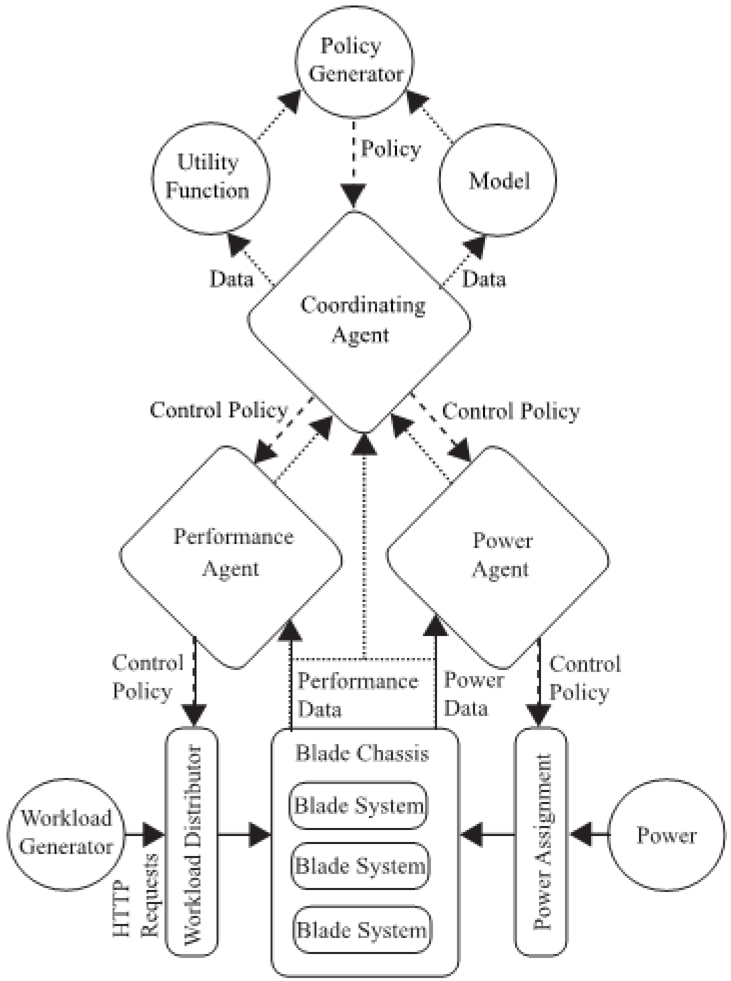
Source: (Tesauro *et al.*, 2007)

Power Savings | Overview

- Problem: can meet SLA requirements, but what about costs?
  - Power consumption second leading cost in 70% of IDCs
  - Power wasted by unused servers
  - Performance/power tradeoff

- Approach
  - Turn off unused servers, turn on when necessary
  - Modeled as a MAS for intelligent, distributed decisions

Power Savings | System



Source: (Das *et al.*, 2008)

Power Savings | Agents

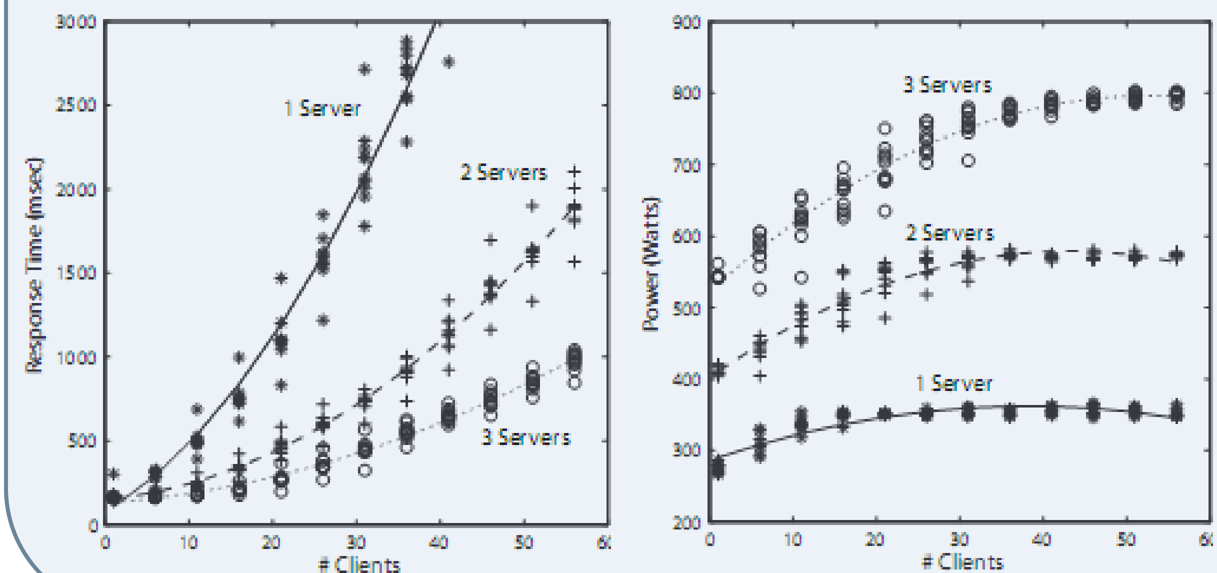
- Performance Agent
  - Load balancing on servers (Apache)
  - Monitor demand from requests
- Coordinating Agent
  - Get info from performance/power agent
  - Choose allocations based on master policy
- Power Agent
  - Monitor power consumption (EMT)
  - Turn machines on/off
  - Can override Coordinating Agent

Power Savings | Policy

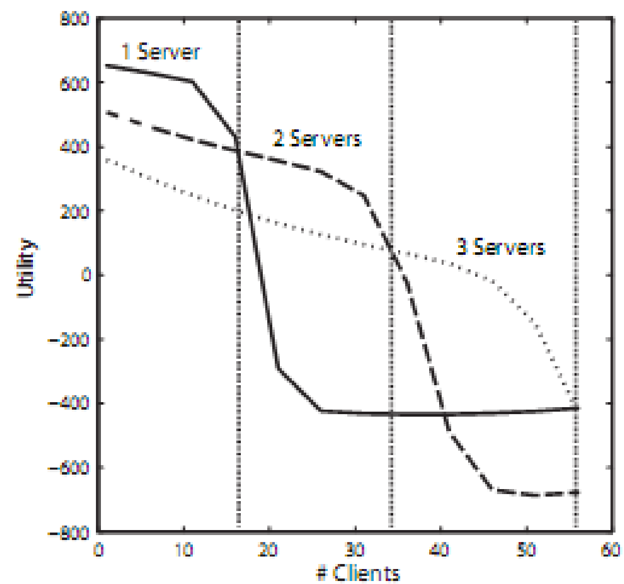
- Goal: maximize utility
  - Positive utility from satisfying requests
  - Negative utility from power consumption
  - Function of control parameters, observations
- Policy
  - Mapping of observations to actions
  - Similar to POMDP solution from previous presentation

Power Savings | Policy

Utility Functions



Joint Utility

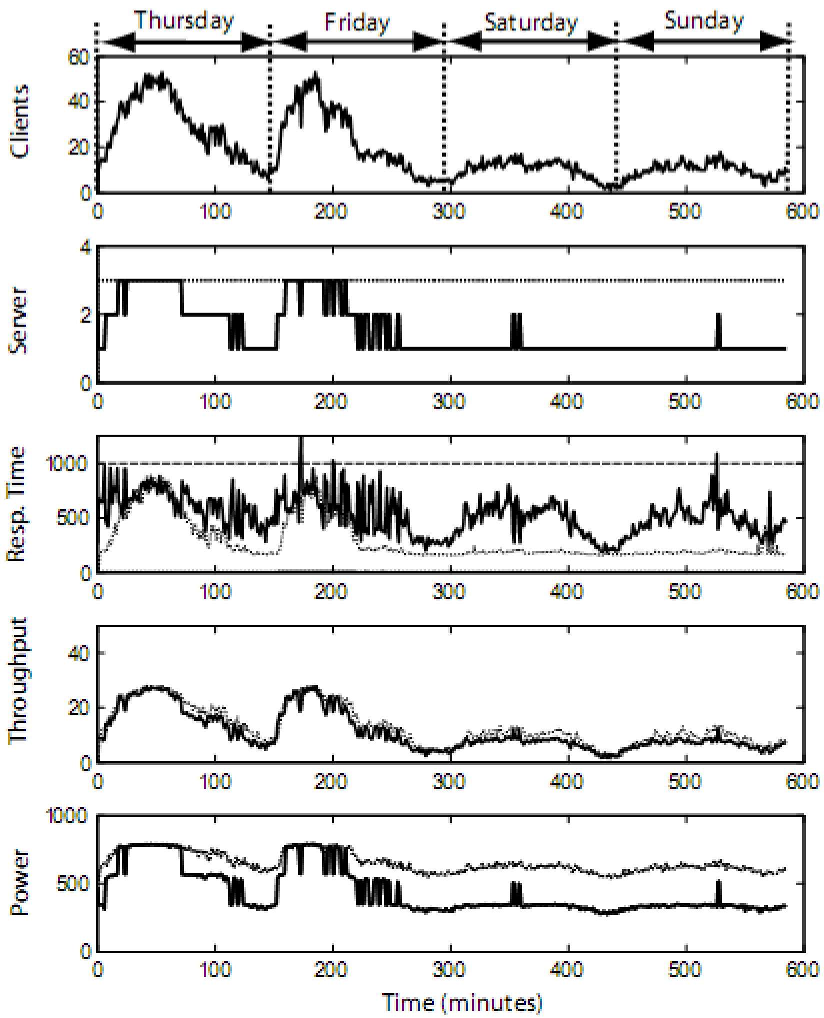


Source: (Das *et al.*, 2008)

Power Savings | Experiment Setup

- Single application
  - Workload: LINPACK linear equations solver
  - Requests: distribution based on NASA web logs
- Servers
  - 3 to run workload
  - 1 for performance agent (Apache)
  - Additional for power/coordination agents, workload generator

Power Savings | Results



Period	Energy (kWh) without Policy	Energy (kWh) with Policy	Energy Saved
Weekdays	16.9	13.5	20.1%
Weekends	14.6	8.5	41.8%
Week	113.7	84.5	25.7%

Source: (Das *et al.*, 2008)

Conclusion | Summary

- Problem: resource allocation in IDCs
  - Servers across multiple applications
  - Performance/power tradeoff
- Hybrid RL improves state-of-the-art in RA across tasks
  - Better than queue-based models
  - Better than "pure" RL approach
- Initial investigation in MAS-based management fruitful
  - Extend to multiple applications?

Questions?

References

- Tesauro et. al., "On the user of hybrid reinforcement learning for autonomic resource allocation", *Cluster Computing*, vol. 10, pp. 287-299, 2007.
- Das et. al., "Autonomic multi-agent management of power and performance in data centers", *Proc. of AAMAS'08*, pp. 107-114, 2008.