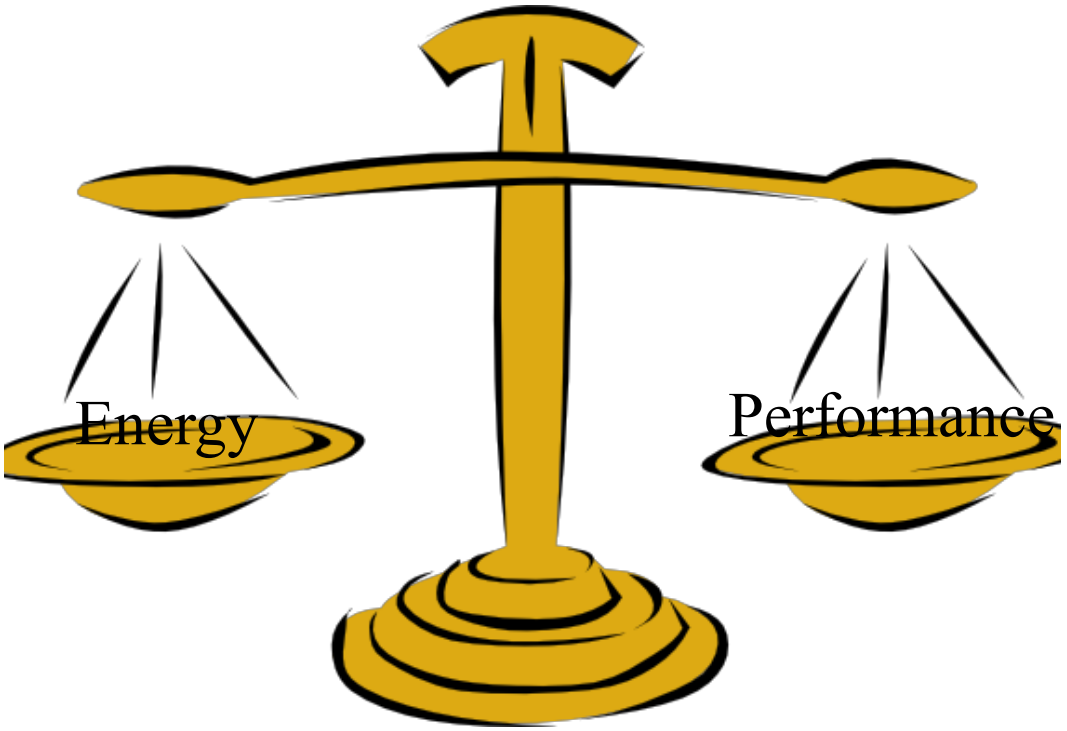
Integrating VM Selection Criteria in Distributed Dynamic VM Consolidation Using Fuzzy Q-learning

Seyed Saeid Masoumzadeh
Helmut Hlavacs

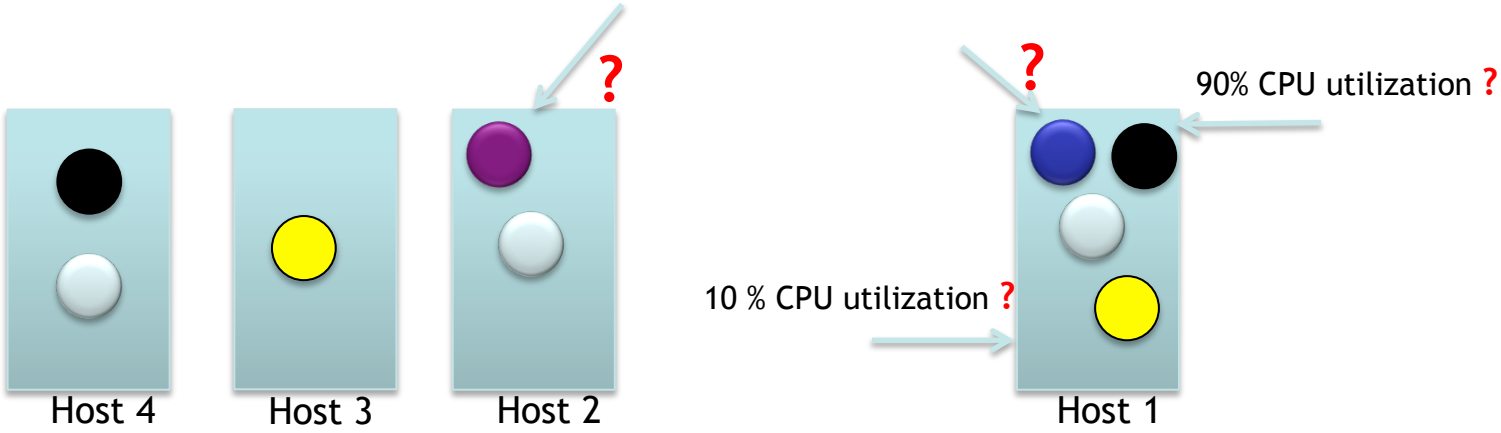
Research Group : Entertainment Computing (Former: Distributed Systems)
University of Vienna

Energy-Performance Trade-off



Distributed Dynamic VM Consolidation

- Host Over-loading Detection
- Host Under-loading Detection
- VM Selection
- VM Placement



VM Selection

1- Minimum Utilization

The VMs with the lowest CPU utilization are selected

2- Maximum Utilization

The VMs with the highest CPU utilization are selected

3- Minimum Migration time

The VMs with the lowest migration time are selected . Based on memory and bandwidth

VM Selection Task as a Dynamic Decision-Making (DDM) Task

In our proposed strategy , each host based on its own current state decides which criterion must be chosen to select VMs towards energy-performance trade-off optimization.

Methodology

Q-Learning

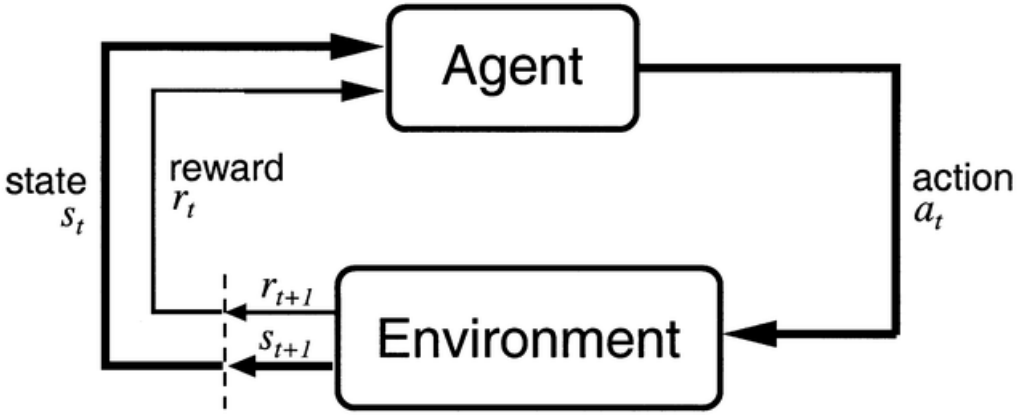
- ✓ Is one of the most popular reinforcement learning (RL) method
- ✓ Is an effective approach to design dynamic system management
- ✓ Is an effective approach to produce dynamic decision-making tasks
- ✓ Can automatically learn high quality decision-making task without an explicit model and with little or no built-in system specific knowledge

Fuzzy Q-learning

- ✓ Is the fuzzy extension of Q-learning
- ✓ Is capable of handling continuity in the state space
- ✓ Is powerful enough to tackle the curse of dimensionality

each physical host is associated with an FQL agent and each FQL learns how to find an optimal strategy to applying multiple criteria for selecting Vms in different states towards improving energy – performance trade-off.

Standard Reinforcement Learning and Q-learning



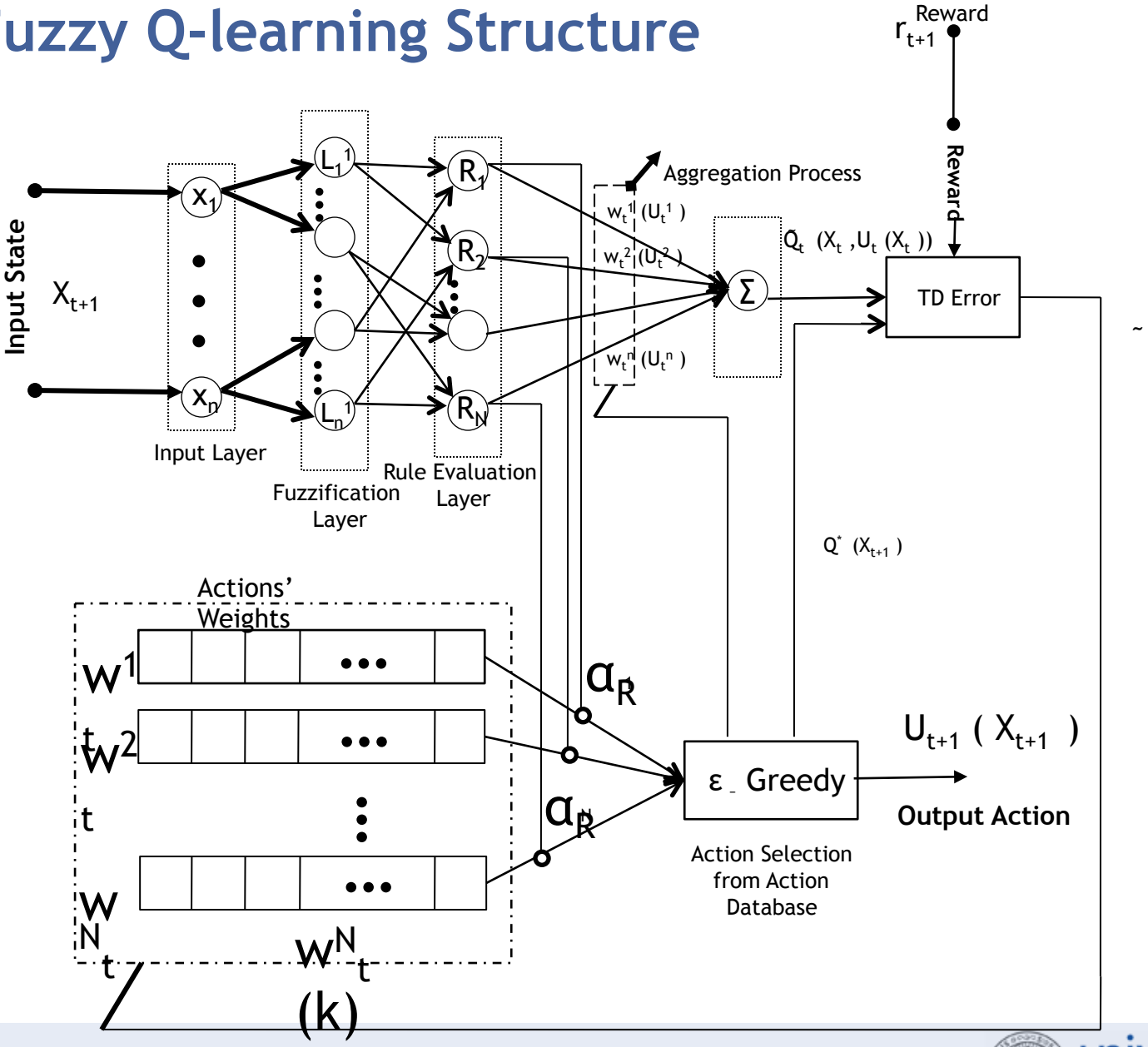
- ✓ The problem can be modeled by Markov Decision Process (MDP)
- ✓ Solved using Dynamic Programming (DP)

- DP provides algorithms to find an optimal policy π^* with corresponding optimal state value function V^*

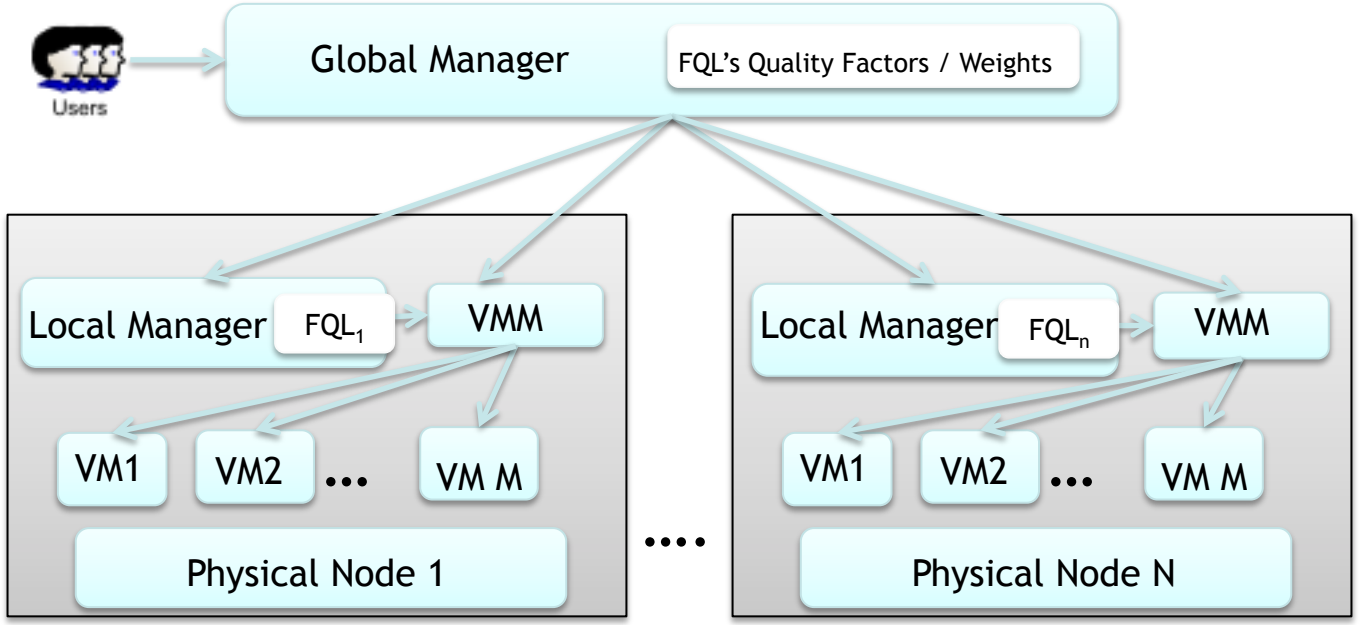
Q-learning :

$$\tilde{Q}_{t+1}(s,a) = \tilde{Q}_t(s,a) + \alpha\left[r_{t+1} + \gamma \max_{a'} \tilde{Q}_t(s',a') - \tilde{Q}_t(s,a) \right], \qquad 0 < \alpha < 1.$$

Fuzzy Q-learning Structure



Our System Model



Energy-Performance Trade-off Model

- Energy Consumption is defined as a linear function of the CPU utilization
- Cloud performance is defined as a function that evaluates the Service Level Agreement (SLA) delivered to any VM deployed in an IaaS.

$$SLAVO = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{s_i}}{T_{a_i}},$$

$$SLAVM = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{d_j}}{C_{r_j}}$$

$$SLAV = SLAVO \times SLAVM$$

$$ESV = SLAV \times EC$$

A. Beloglazov and R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic
B. Consolidation of Virtual Machines in Cloud Data Centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1–24, 2011.

Energy-Performance Trade-off Model

$$SLAVO_i = \frac{T_{s_i}}{T_{a_i}}, \qquad 1 \leq i \leq N$$

$$SLAVM_i = \frac{1}{K_i} \sum_{j=1}^{K_i} \frac{C_{d_{ji}}}{C_{r_{ji}}}, \qquad 1 \leq i \leq N$$

$$SLAV_i = SLAVO_i \times SLAVM_i, \qquad 1 \leq i \leq N$$

$$ESV_i = SLAV_i \times EC_i, \qquad 1 \leq i \leq N$$

Formulating the FQL Task

The Input State:

$$X_{t_i} = \{CU_{t_i}, NumVM_{t_i}\}, \qquad 1 \leq i \leq N$$

The Action Set:

$$A(X_t) = \{Criterion_1, Criterion_2, ..., Criterion_n\}$$

The reward Function :

The reward is a performance feedback on the resulted new VM selection criterion chosen from the action set.

$$Reward_{i_{t+1}} = \frac{1}{ESV_{i_{t+1}}}, \qquad 1 \leq i \leq N$$

FQL Procedure

When the VM selection task is called for a host

1- The FQL agent associated with this host observes the current state $\mathbf{S}_{ti}$ and gathers information about the input state $\mathbf{X}_{ti}$

2- Immediately choses a VM selection criterion as an action from the action set $A(X_t)$ in order to make a decision.

3- One time-step later, the FQL agent receives a numerical reward, $\mathbf{Reward}_{it+1}$ and finds itself in a new state $\mathbf{S}_{it+1}$.

In a trial and error interaction, finally the FQL agent learns how to map the states to the actions in order to maximize the discounted sum of rewards

the FQL agent learns the best sequence of VM selection criteria, corresponding to the states observed during its lifetime.

Experimental Setup

1- The CloudSim toolkit has been chosen as simulation platform

2- Real life data on workload are provided by the CoMon project

3- We simulated a data center comprising 800 heterogeneous physical nodes

- half of which are HP ProLiant ML110 G4
- other half consisting of HPProLiant ML110 G5 servers

4- The power consumption of the selected servers is different at each load level

5- The frequencies of the servers' CPUs are mapped onto MIPS rating

6- Each server is modeled to have 1 GB/s network bandwidth

7- The characteristics of the VMs types corresponding to Amazon's EC2 instance types including

- High-CPU Medium Instance
- Extra Large Instance
- Small Instance
- Micro Instance

8- We used three different workloads, which were collected in a timespan of three different days.

Experimental Setup

In our dynamic VM consolidation procedure, for all of the experiments we used :

1- A static threshold based algorithm to detect overloading host

2- A Best fit decreasing algorithm to place VM

3- A simple strategy to detect under-loading host

selects the host with the minimum utilization compared to the other hosts

FQL Setup:

1- We used three fuzzy sets in the FIS configuration with Gaussian membership functions for each input set element

2- The elements of the action set are :

$$A(X_t) = \{MINU, MAXU\}$$

Experimental Setup

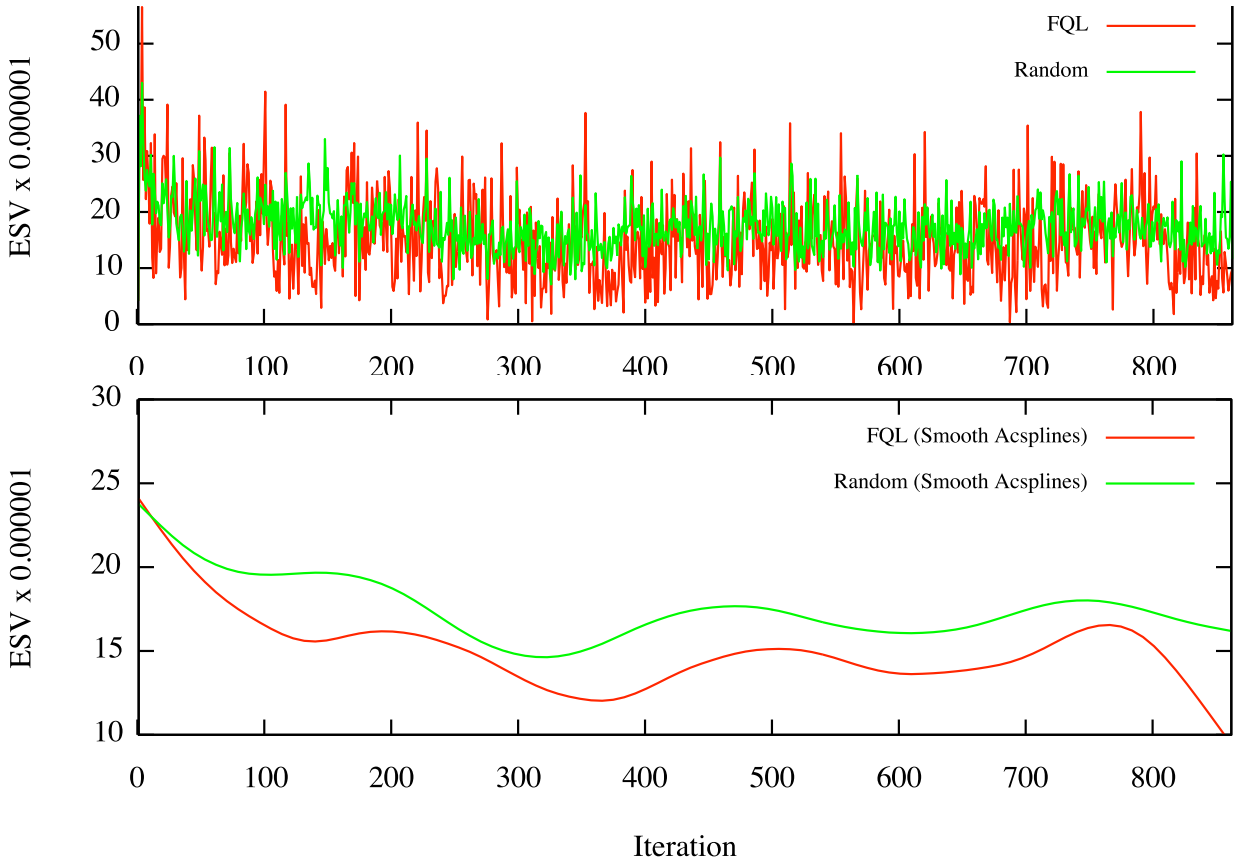
FQL Setup :

1- The learning rate of Q-leaning has been experimentally set to 0.1

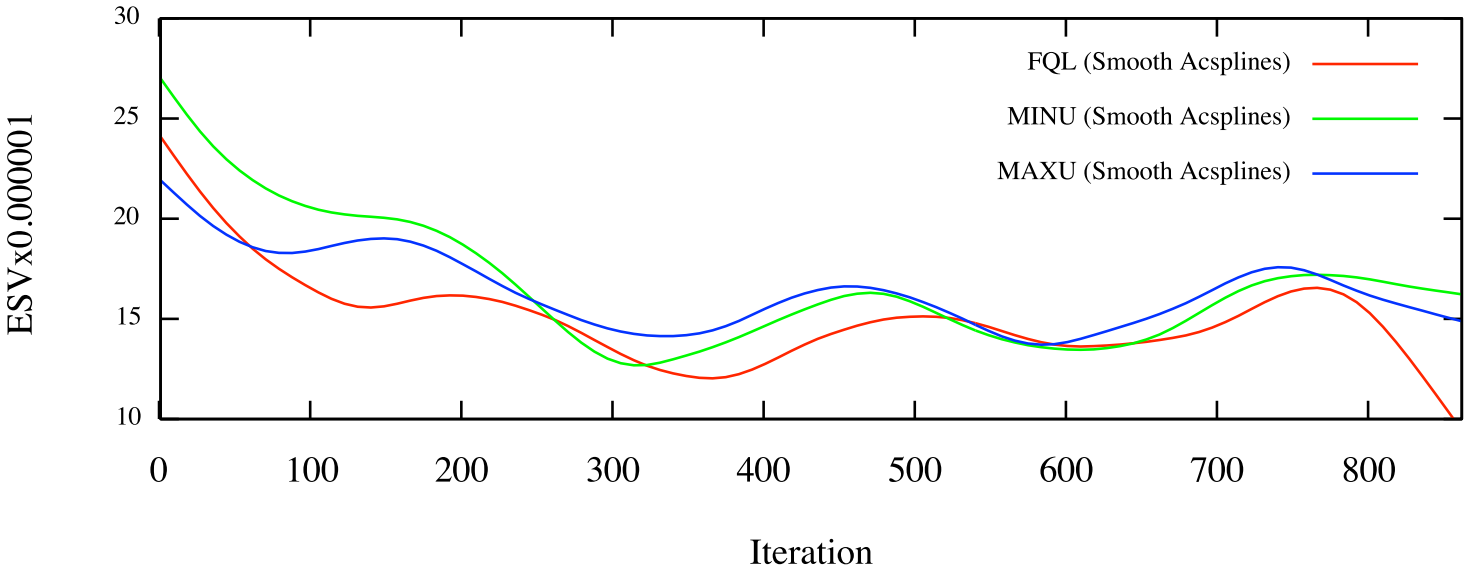
2- The quality factors of actions (weights) were initialized randomly

3- The FQL-iteration or time-step for perceiving next state and receiving the reward of the previous state has been set to 300 seconds

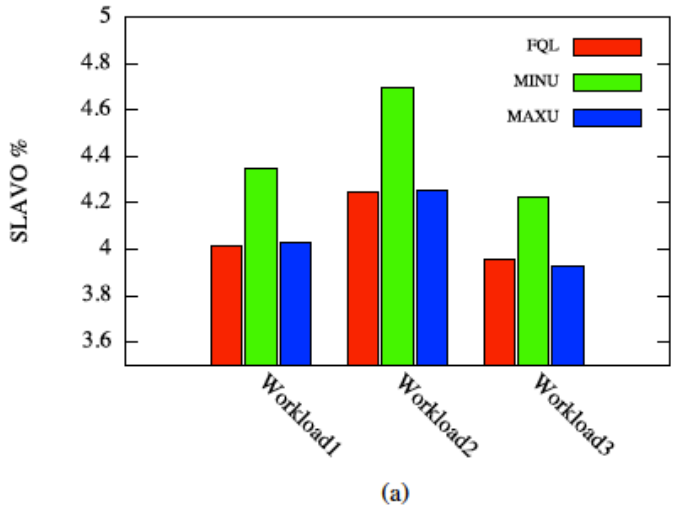
Experimental Results



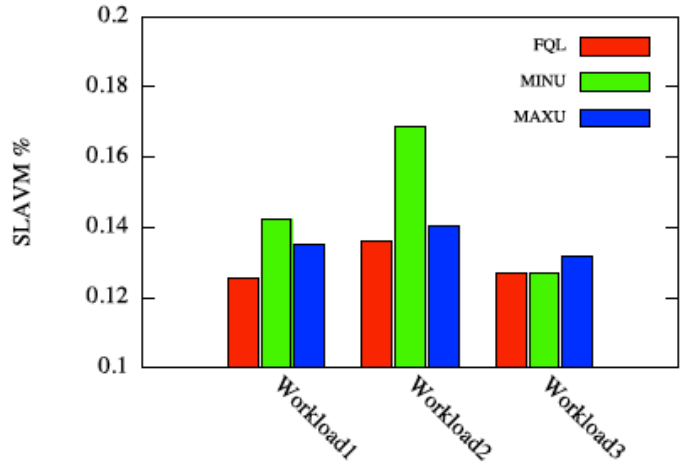
Experimental Results

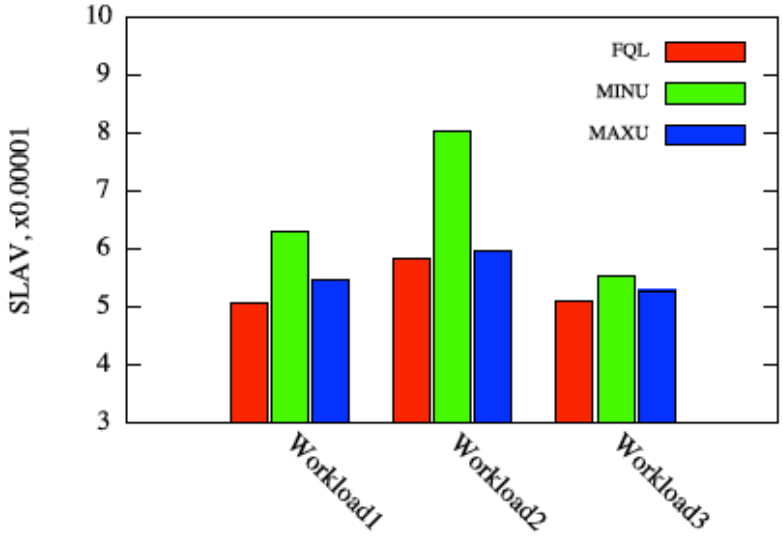


Experimental Results



(a)





Experimental Results