毕业实训进展汇报

1752919-祁好雨

目录

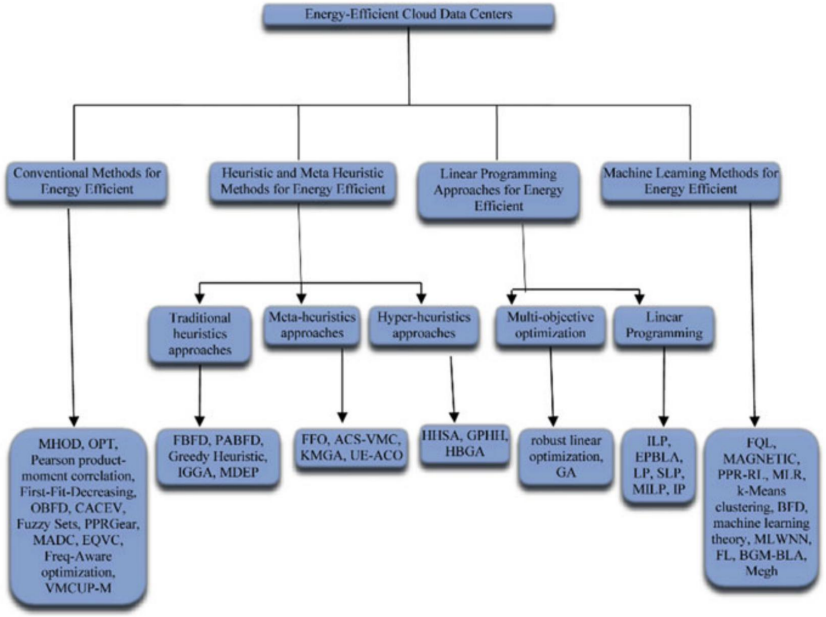
- 理论进展
- 实践进展

理论进展

- 论文阅读
- kubernetes in action

论文阅读：A review on dynamic consolidation of vm ...

- consolidation(vm):Virtual machine (VM) consolidation utilizes live migration of virtual machines (VMs) to transfer a VM among physical servers in order to improve the utilization of resources and energy efficiency in cloud data centers.
- Background study：conventional methods；linear programming approaches；heauristic methods；ML based



论文阅读： A review on dynamic consolidation of vm ...

S. No	Scheme	Method name	Advantages	Disadvantages
1	FQL [44]	Fuzzy Q-learning (FQL)	Energy-performance trade-off in cloud data centers	It is not implementing more VM selection
2	MAGNETIC [45]	Multi-agent machine learning-based approach for energy-efficient dynamic consolidation (MAGNETIC)	Reduces data center energy consumption	Not support for parallel applications at a time
3	ML [46]	Heuristic and machine learning (ML)	Energy and network traffic for reduced scenarios	High error-susceptibility
4	PPR-RL [47]	Performance-to-power ratio-based reinforcement learning (PPR-RL)	Reduces the energy consumption	It does not support real cloud infrastructure
5	Regression [48]	Regression-based approach	Improves energy efficiency	It is assumed that the cause and effect relationship between variables remains unchanged
6	MLR [49]	Multiple linear regression (MLR)	Increased energy consumption of the servers	It assumes that is a dependent and independent variable which is incorrect many times
7	BFD [50]	k-Means clustering, stochastic Wiener filter and modified best fit decreasing (BFD)	Reduces energy consumption	Euclidean distance for all the data points in the cluster and it will become stable
8	Machine learning [51]	Machine learning theory	Save energy consumption, energy efficiency	This system not focused in the heterogeneous environment
9	MLWNN [52]	Multiple linear regression and wavelet neural network (MLWNN)	Suitable for low workload cloud data center	It is required processors with parallel processing power
10	RL and FL [53]	Reinforcement learning (RL) and fuzzy logic (FL)	Energy-efficient resource allocation	It is not always accurate
11	LA and game theory [54]	Learning automata (LA) and game theory	Energy consumption	Not focused on security problems

(continued)

Table 4 (continued)

S. No	Scheme	Method name	Advantages	Disadvantages
12	BGM-BLA [55]	Binary graph matching-based bucket-code learning algorithm (BGM-BLA)	Energy consumption, communication between VMs	Not suitable for a real-time environment
13	Megh [56]	Megh	Energy-efficient resource management	Not support for large-scale cloud data center
14	Speedy cloud [57]	Speedy cloud	Increased resource utilization	Not focused on scheduling jobs among CPUs and FPGAs

论文阅读： Applying reinforcement learning towards automation resource allocation and application scalability

- Introduction

- Iaas的优势：动态调度
- 动态调度是困难的：1. resource performance始终在变，2. 调度会影响同host的其他vm
- MDP(马尔可夫决策过程)和Q-learning等增强学习手段适应这种变化的情况下做决策
- 创造了parallel Q-learning减少做决策的时间，解决新增state variable对维度的影响

论文阅读

- VMM控制VM对host resources的access，并不隔离performance(?)
- performance unpredictability是一个主要困难
- 传统方法是rule-based，根据超参upper bound和lower bound来scale up/down
- 再进一步使用MDP
- 但是MDP需要时间来得出计算结果

- 对这个drawback的一个解决方法是使用一个在外部sample data上面已经train好的外部决策方法（有点像迁移学习）

- 该论文提出了一个和外部决策互为补充的内部决策算法(根据当前数据train)

论文阅读

- 提出的parallel q-learning有两个优势：1. 利用并发，决策时间减少 2.适应resource number的变化

- 实验环境：
  - Xen Hypervisor：无法选择实际的接口；不能隔离同hostVM的performance
  - Microbenchmark EC2

- TODO：Q-learning

论文阅读

Algorithm 1 Reinforcement Learning Algorithm (Q-learning)

Initialize $Q(s, a)$ arbitrarily

Repeat (for each episode)

Initialize s

repeat

Choose a from s using policy derived from Q (ϵ-greedy)

Take action a and observe r, s'

$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$

$s \leftarrow s'$;

until s is terminal

论文阅读

Algorithm 2 Parallel Q-Learning

Initialise $Q(s,a)_l = 0$, $Q(s,a)_g = 0$, $Q(s,a) = 0$
$comms_{out}$, $comms_{in} \leftarrow \emptyset$
$\pi \leftarrow$ an arbitrary ϵ-greedy policy w.r.t to Q

repeat
  for all $s \in S$ **do**
    Choose a from s using policy π
    Take action a, observe r, s'
    $Q(s,a)_l \leftarrow Q(s,a)_l + \alpha[r + \gamma \max_{a'} Q(s',a')_l - Q(s,a)_l]$
    $s \leftarrow s'$;
    if $\| (Q(s,a)_l - Q(s,a)_g) \| > \theta$ **then**
      Add $Q(s,a)_l$ to $comms_{out}$
    end if
    Transmit $comms_{out}$ to all agents
    Receive $comms_{in}$ from other agents
    if $comms_{in} \neq \emptyset$ **then**
      for all $Q(s,a)_g \in comms_{in}$ **do**
        $Q(s,a) = \frac{Q(s,a) \times Expc_l + Q(s,a)_g \times Expc_g}{Expc_l + Expc_g}$
      end for
    end if
  end for
until s is terminal

- 多个agents在同个task上learning
- 访问不同的states和action，计算不同rewards，有不同的learning experience
- 每一个agent有一个local Q_l和一个global Q_g
- Q_g是所有agents q value的权重和
- agents之间的信息通过communication arrays in 和 out来传递

论文阅读reference： MDP

MDP：https://towardsdatascience.com/understanding-the-markov-decision-process-mdp-8f838510f150

- agent
- environment
- state
- action
- policy

什么是马尔可夫状态?

A state S_t is *Markov* if and only if

$$\mathbb{P}[S_{t+1} \mid S_t] = \mathbb{P}[S_{t+1} \mid S_1, ..., S_t]$$

论文阅读reference： MDP

什么是马尔可夫状态?

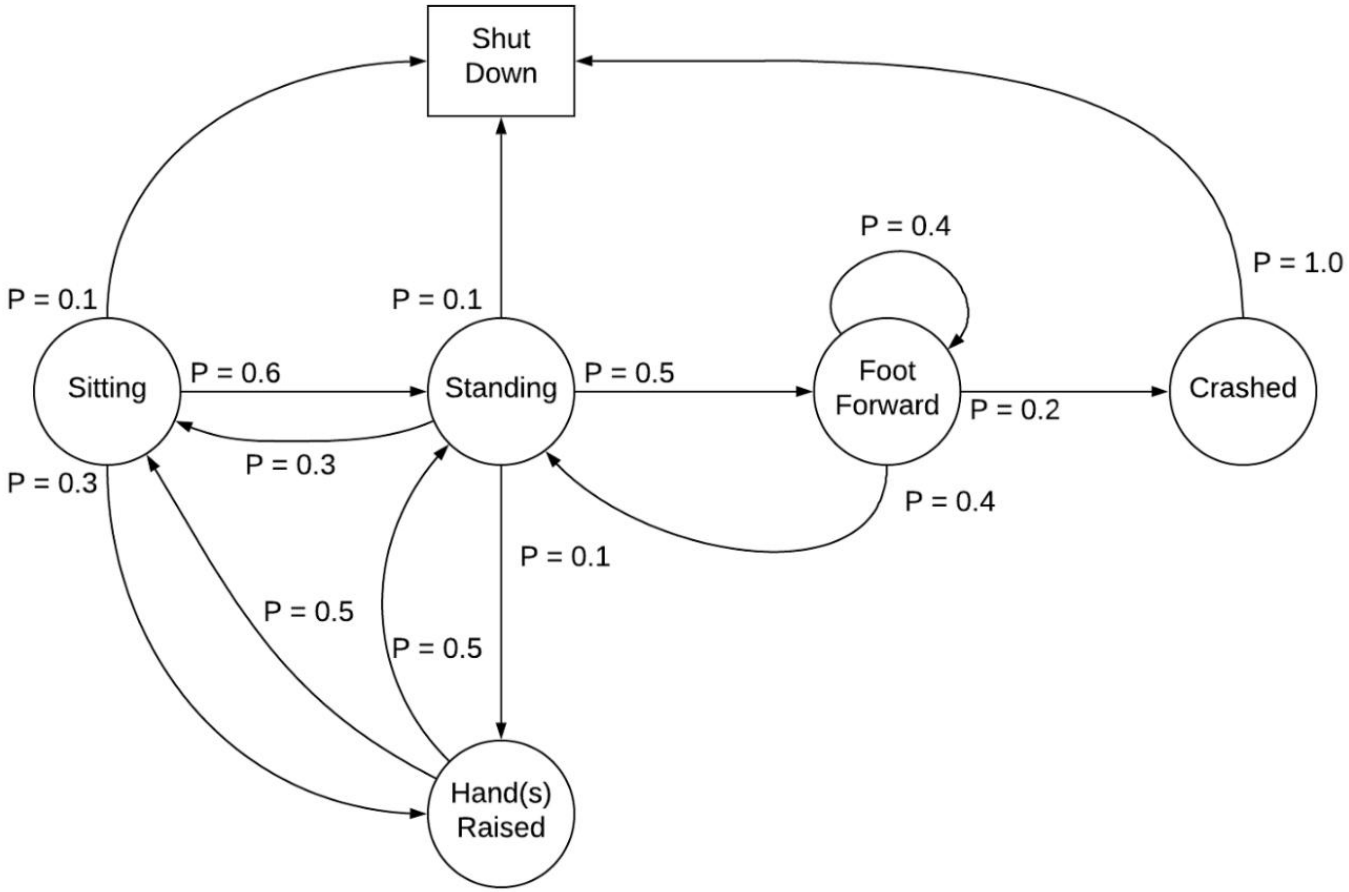
- 这一刻的状态只取决于上一刻的状态

马尔可夫链或者说马尔可夫process:

- $$\mathcal{P}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s\right]$$

- p是状态转移概率（其中从s_t到s_t+1的每一个状态都是马尔可夫的）

论文阅读reference： MDP



论文阅读reference： MDP

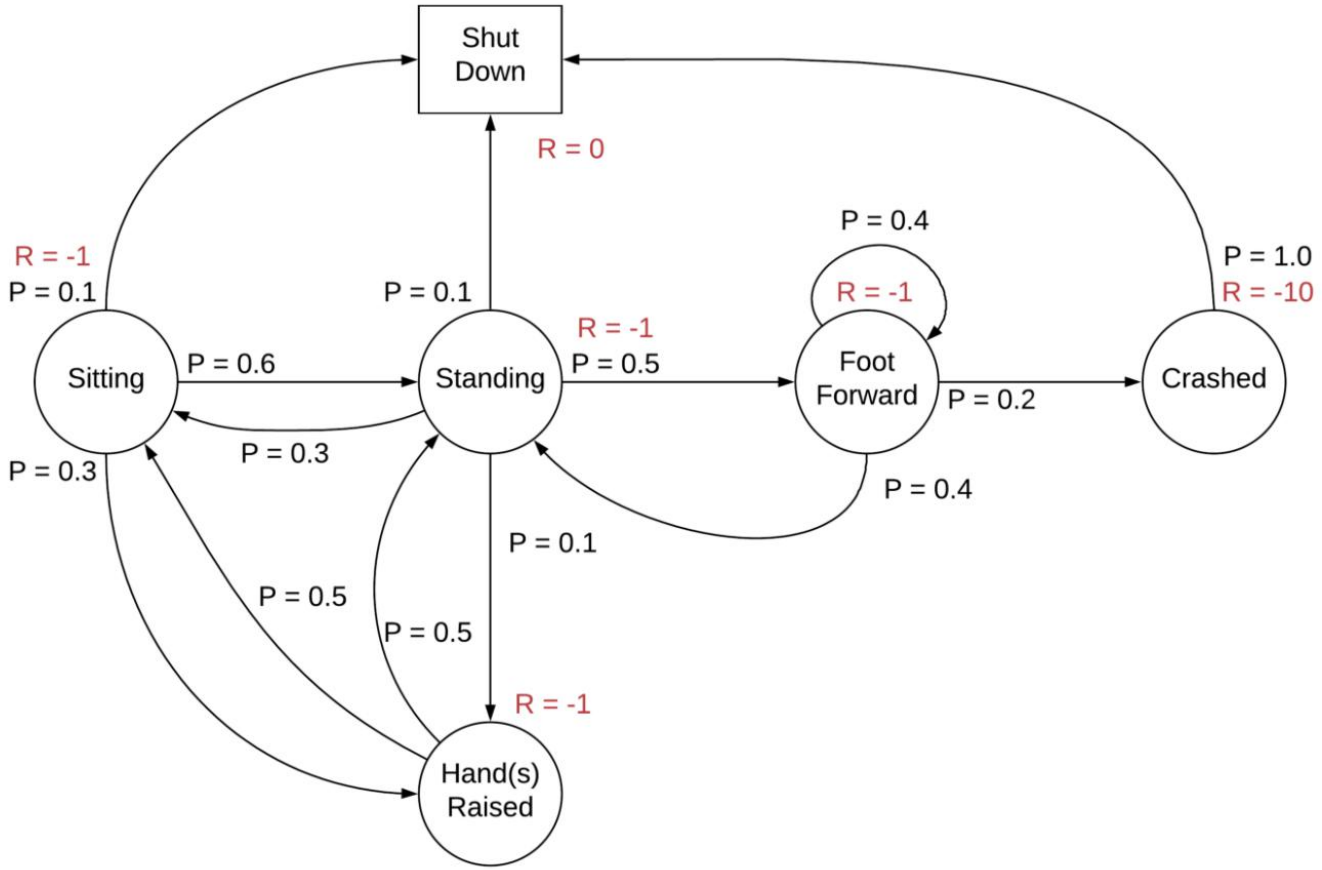
- 马尔可夫 reward
-

$$\mathcal{P}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s\right]$$

$$\mathcal{R}_s = \mathbb{E}\left[R_{t+1} \mid S_t = s\right]$$

- R是一个期望reward，是根据所有从s状态能够转移到的地方及概率计算得出，一旦agent到达状态s，则获得R_s reward

论文阅读reference： MDP



论文阅读reference： MDP

$$\mathcal{P}^{a}_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$$

$$\mathcal{R}^{a}_{s} = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

- A_t是agent在t时刻能做出的action

The *return* G_t is the total discounted reward from time-step t.

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- reward是暂时的，我们关注总共的return

论文阅读reference： MDP

- 上述公式中的discount代表future reward是不确定的；另一方面也避免future reward对return有太大影响(discount属于[0,1])

A *policy* π is a distribution over actions given states,

$$\pi(a|s) = \mathbb{P}\left[A_t = a \mid S_t = s\right]$$

- policy给了一个在当前state做action的概率分布

$$v(s) = \mathbb{E}\left[G_t \mid S_t = s\right]$$

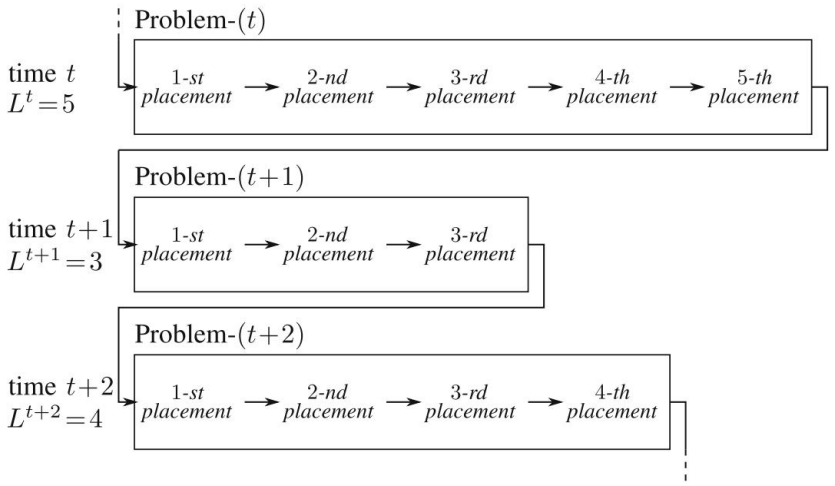
- state value function是我们比较关注的，代表了长期的reward（在某个状态）

$$v(s) = \mathcal{R}_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'} v(s')$$

论文阅读：Deep reinforcement learning

- efficient cloud datacenters实现策略：
  - consolidation：在host间挪vm/trade off between energy and quality
    - 缺点：
      - 不可忽视的网络带宽要求和计算资源占用
      - 技术难度
  - placement：类似bin packing problem，host是bin，vm是object
    - 所有的VM分成S个class，不同class的VM分配不同资源
    - 用户请求固定数量的一系列VM
    - ML可以用来作为replacement的算法；也可以用来选择不同的heuristics
    - 常用技术：Q-learning、DRL

论文阅读： Deep reinforcement learning



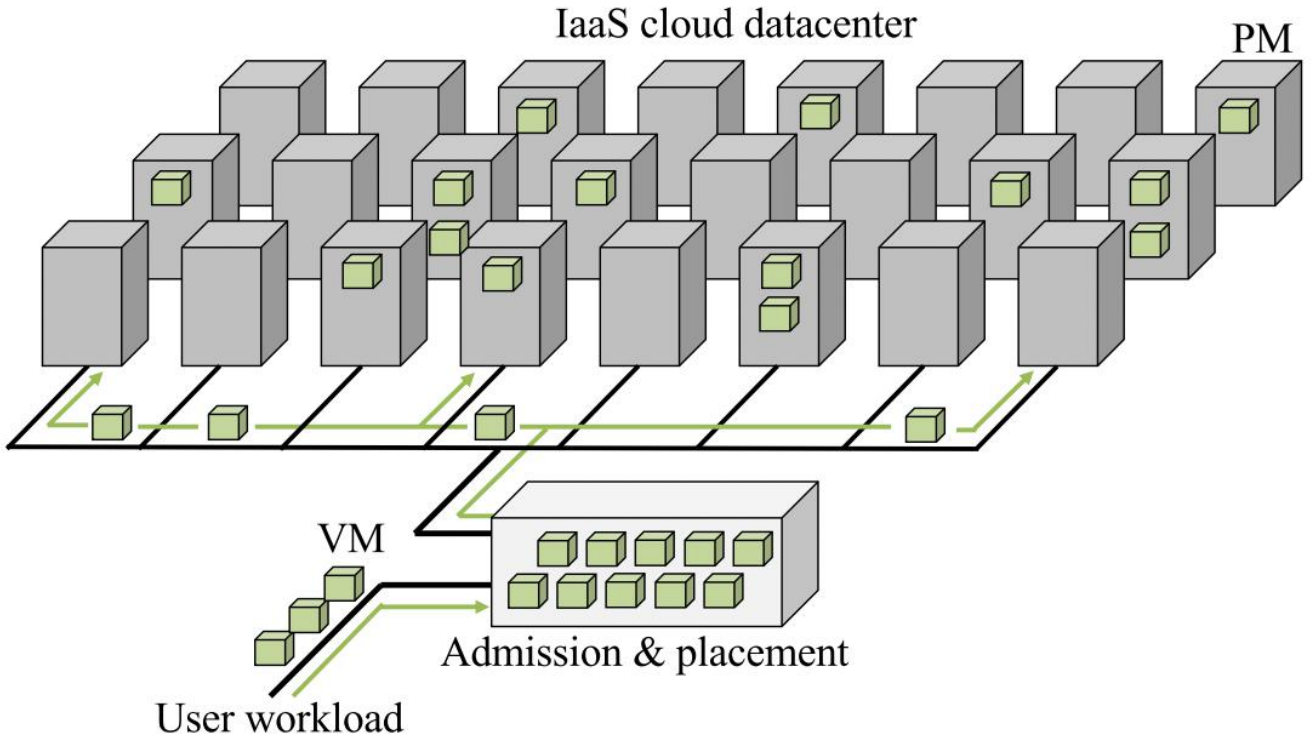


Fig. 1 Reference scenario for the VM placement problem

实践进展（演示代码）

- 多选pods监控模块
- 可视化模块（前端echarts绘图）
- 可视化模块（前端websocket协议实现）
- 可视化模块（后端websocket协议实现）
- 预测模块