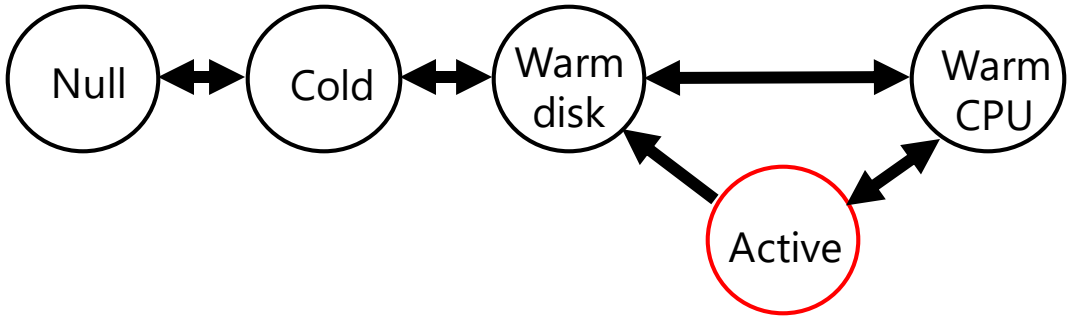
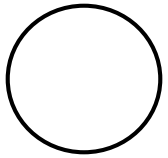
Reinforcement Learning

Serverless container lifecycle



State	Description
Null	Idle, nothing is related to the service
Cold	The abstraction of service has been deployed
Warm disk	Service and Image are available
Warm CPU	Container is available, code in container is running and waiting for incoming connection
Warm Mem	Container is available but in “paused”/sleep state
Active	Code is processing the request

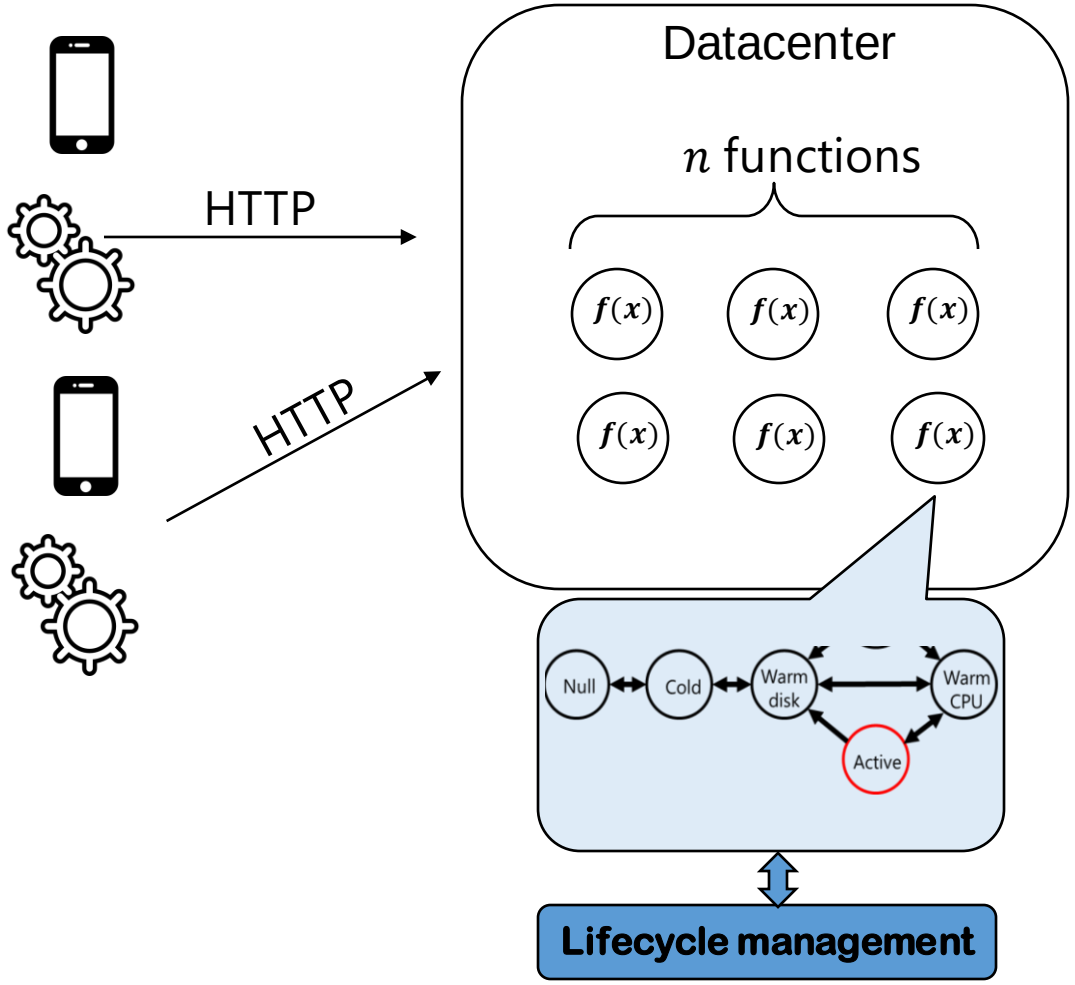
- Moving from one state to another costs:
  - Time
  - Energy



- Staying at one state costs:
  - Resource (RAM)

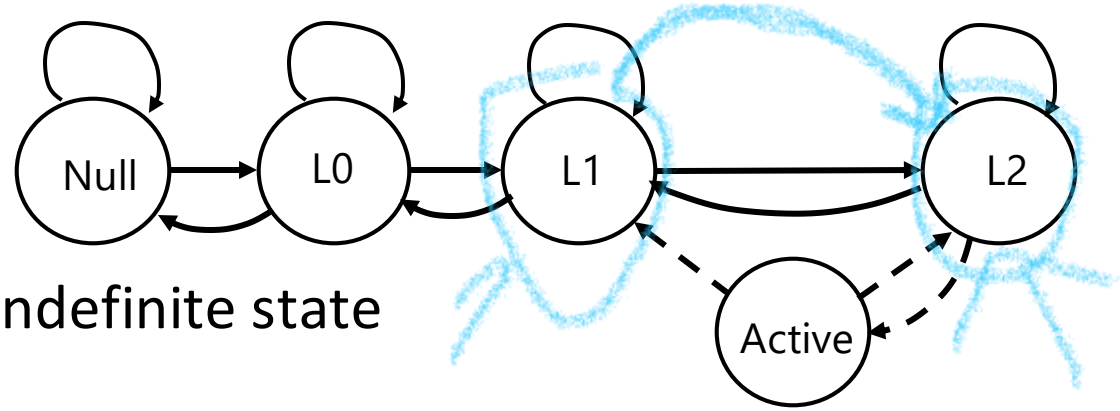
Problem and Methodology

- Scenario
  - DC with n functions
  - Functions are serverless (flexible lifecycle)
- Challenges
  - ~~How to minimize latency caused by moving from intermediate states to Active?~~
  - ~~How to minimize energy consumption caused by moving back and forth between states?~~
  - How to maximize revenue for operators?
- Methodology
  - Applying RL to prepare serverless functions in advance



Simplified state machine

- Simplified state-machine
  - L0, L1, L2 indicate cache level
  - Active may mean nothing because it's not an indefinite state



- Profiling results
  - Costs of staying at one state and move to another states

Transition	RAM required (GB)	Time (ms)	Energy (J)
Null -> L0	0	5000	50
L0 -> L1	0	50000	2000
L1 -> L2	0.9	7000	100
L2 -> Act	1.2	50	0
Act -> L2	0.9	0	0
L2-> L1	0	30000	400
L1 -> L0	0	2000	50
L0 -> Null	0	1400	50

State	RAM required (GB)	CPU	Power (W)
Base	0	0	75
Null	0	0	0
L0	0	0	0
L1	0	0	0
L2	0.9	0	0
Act	1.2		50

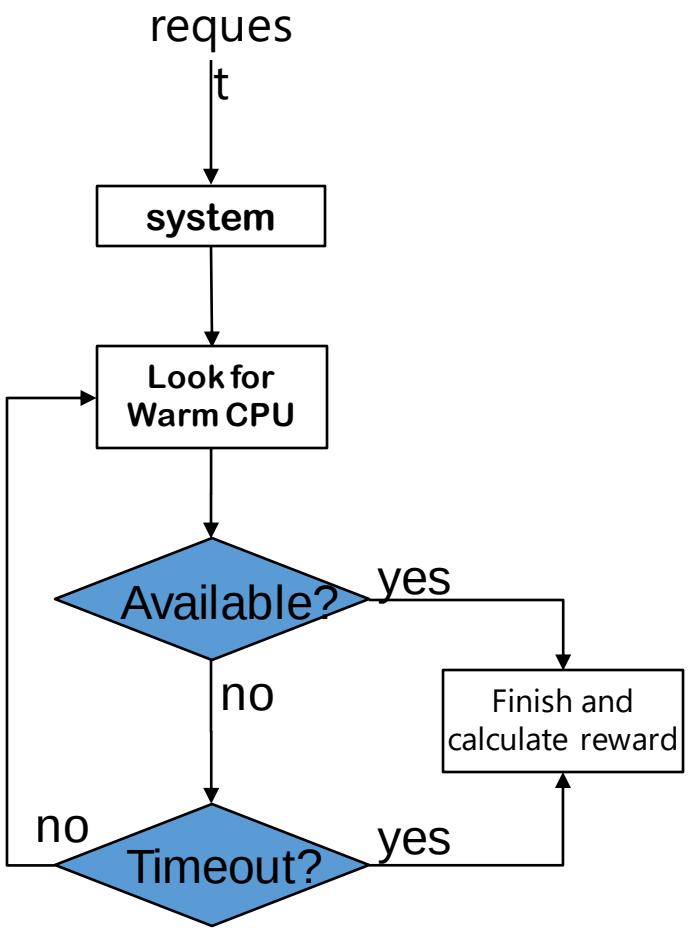
Traffic model

- System has N types of service*, I functions per service.
  - Each instance i can serve only one request of type n
  - Total number of instances is: $N * I$
  - *We must have multiple services, otherwise the system will keep everything at Warm CPU

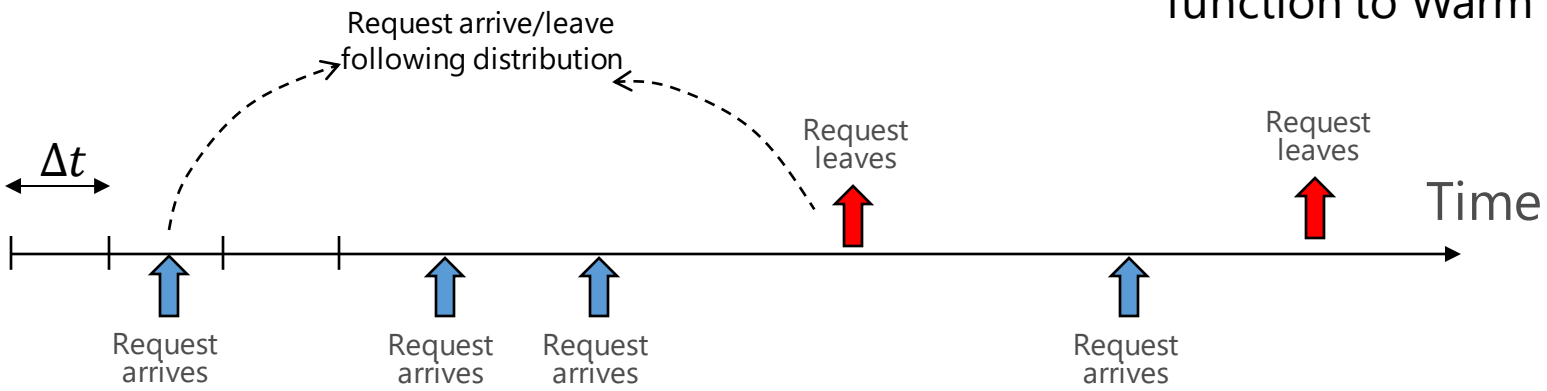
- Online problem
  - Arrival rate following Poisson distribution, for ex: $\lambda = 100 \sim 100 req/hour$
  - One request of type n arrives, one instance i of n must be turned to Active
  - Assume the each request of type N requires $t_{processing}$ seconds

Workflow

- The number of instances is fixed and equals $N * I$.
- DRL makes decision every Δt
  - Δt must be > 1min, because longest action is 50s
- When request arrives, the following operates:
  - System looks for Warm CPU functions to host the request
  - Request can wait for a fixed time_out before marked as failure
  - Reward will be calculated, and model will be updated



Ideally: during this timeout, RL must realize it needs to turn on a function to Warm CPU



Variable

Notation	Description
$S = \{s_i\}, i \in \Omega_S$	Set of servers on cloud
$S_{run} = \{s_{run_i}\}, S_{run} \subset \Omega_S$	Set of servers are running which have containers are WarmCPU or Active state
$T = \{t_i\} = \{Null, L_0, L_1, L_2, Active\}, i \in \Omega_T$	Set of container's state: Null, Cold, WarmCPU, WarmDisk, Active
$V = \{v_i\}, i \in \Omega_V$	Set of services
$C = \{c^i_{j\,k}\}, j \in V, k \in T$	Set of container in current state of environment. $c^i_{j,k}$ is i^{th} container belong j^{th} service and in k^{th} state.
$A = \{a_i\}, i \in \Omega_a$	Set of avaible action.
$C_{p,q} = \{c^i_{j,k} \in C \mid j = p, k = q\}$	Set of container which belong p^{th} service and in q^{th} state in current state of environment.

Variable

Notation	Description
$M_S = [e_{i,j}],\ 1 \le i \le \lvert V \rvert,\ 1 \le j \le \lvert T \rvert$ $e_{i,j} = \lvert C_{i,j} \rvert$	Matrix state of S state
$M_c^a = [e_{i,j}],\ 1 \le i, j \le \lvert V \rvert$	Coefficient matrix belong action a. $e_{i,i}$ is number container belong service i^{th} which will be change in a.
$M_u^a = [e_{i,j}], 1 \le i \le \lvert V \rvert,\ 1 \le j \le \lvert T \rvert$	Unit matrix.
$M^a = [e_{i,j}], 1 \le i \le \lvert V \rvert,\ 1 \le j \le \lvert T \rvert$ $M^a = M_c^a \times M_u^a$	Action matrix.
RAM_i	Ram usage of a container in i state
RAM_{server}	Ram usage of a server.
CPU_i	CPU usage of a container in i state
CPU_{server}	CPU usage of a server.

Power

$$P_{s_i} = P_{s_i}^{base} + \sum_{j=1}^{V'} P_{v'^{j}_{s_i}}^{active}$$

$$P = \sum_{i=1}^{S} P_{s_i}$$

- P_{s_i}: Power of i^{th} $server$
- $P_{s_i}^{base}$: $Power\ of\ i^{th}\ server\ at\ idle\ state$
- $P_{v'^{j}_{s_i}}^{active}$: $Power\ of\ j^{th}\ service\ in\ i^{th}\ server\ at\ active\ state$

Energy consumption

$$E_{\Delta t} = \left[\sum_{i=1}^{S_{run}} P_{s_{run_i}}^{base} + \sum_{i=1}^{V} P_{v_i}^{active} \right] * \Delta t + \sum_{i=1, j=1}^{T} E_{r_{t_i}^{t_j}}$$

- $E_{\Delta t}$: $Energy\ during\ \Delta_t$
- $P_{s_{run_i}}^{base}$: $Power\ of\ i^{th}\ running\ server\ at\ idle\ state$
- $P_{v_i}^{active}$: $Power\ of\ j^{th}\ service\ at\ active\ state$
- $E_{r_{t_i}^{t_j}}$: $Energy\ of\ process\ change\ state\ from\ t_i\ to\ t_j$
- $|S_{run}| = [\sum_{i=1}^{V} M_{v_i}^{Wcpu} + \sum_{i=1}^{V} M_{v_i}^{active}] / \sum_{i=1}^{S} M_{s_i}$
  - $M_{v_i}^{Wcpu}$: $RAM\ consumption\ of\ i^{th}\ service\ at\ WarmCPU\ state$
  - $M_{v_i}^{active}$: $RAM\ consumption\ of\ i^{th}\ service\ at\ Active\ state$
  - M_{s_i}: $RAM\ consumption\ of\ i^{th}\ server$

Resource (consider about base state)

- $M_{s_i} = \sum_{i=1}^{V} M_{v_i}^{Wcpu} + \sum_{i=1}^{V} M_{v_i}^{active} + \sum_{i=1}^{S_{run}} M_{s_{run_i}}^{base}$

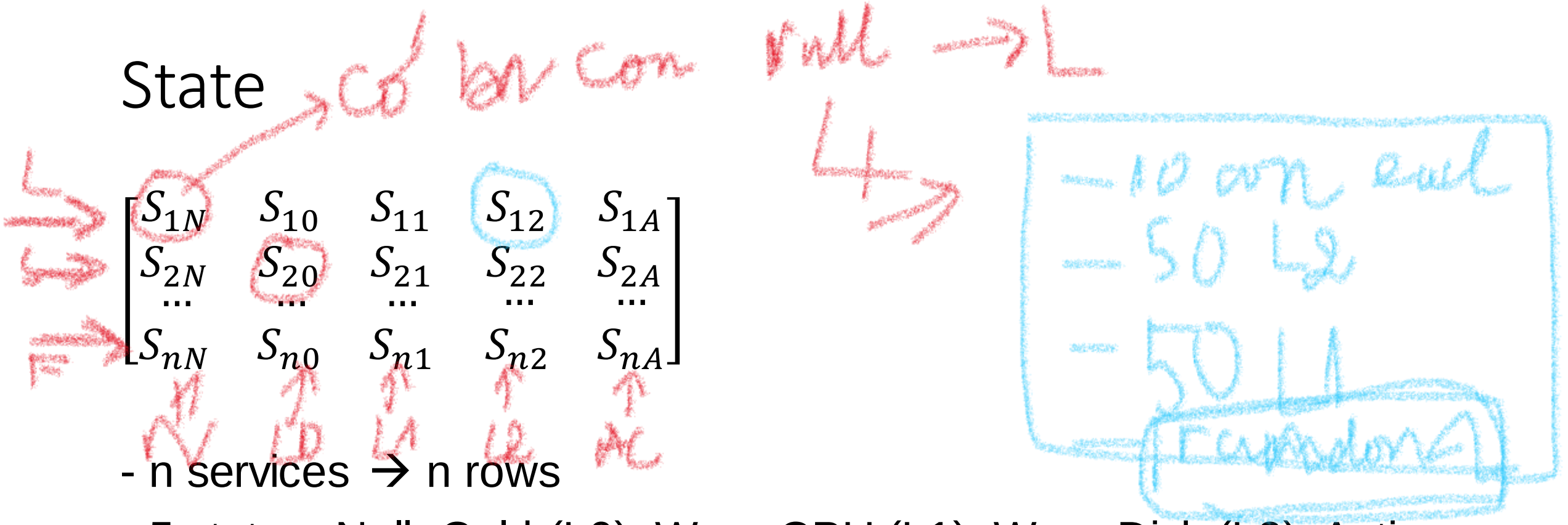
$=> M = \sum_{i=1}^{S} M_{s_i}$

- $C_{s_i} = \sum_{i=1}^{V} C_{v_i}^{active} + \sum_{i=1}^{S_{run}} C_{s_{run_i}}^{base}$

$=> \mathrm{C} = \sum_{i=1}^{S} C_{s_i}$

State

$$\begin{bmatrix} S_{1N} & S_{10} & S_{11} & S_{12} & S_{1A} \\ S_{2N} & S_{20} & S_{21} & S_{22} & S_{2A} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ S_{nN} & S_{n0} & S_{n1} & S_{n2} & S_{nA} \end{bmatrix}$$



- n services → n rows
- 5 states: Null, Cold (L0), WarmCPU (L1), WarmDisk (L2), Active → 5 column
- cell is number of container are this state

Action

$$\begin{bmatrix} S_{1N} & S_{10} & S_{11} & S_{12} & S_{1A} \\ S_{2N} & S_{20} & S_{21} & S_{22} & S_{2A} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ S_{nN} & S_{n0} & S_{n1} & S_{n2} & S_{nA} \end{bmatrix}$$

-1	Previous state
0	Do nothing
1	Next state

$$\begin{bmatrix} c_1 & 0 & 0 & 0 \\ 0 & c_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & c_n \end{bmatrix} x \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- Unit matrix M_u, Initiate all $M_u = 0$
- Coefficient, diagonal matrix M_c, Initiate all $c_n = 1???$

Reward model

- Accepted request λ_{acc} leads to profit:

$$Profit = \lambda_{acc} \times RAMused \times \mathrm{Exec_{time}} \times \$perUnit \ (1)$$

$$Reward = Profit - Energy_{cost} - Penalty_{cost}^{delay} - Penalty_{cost}^{abandone}$$

- $Profit$ follows AWS equation.
  - $RAMused$ is GB, 500MB = 0.5
  - $\$perUnit = 0.0000166667\$$
  - $\mathrm{Exec_{time}}$ is fixed and can serve as the coefficient value, ex. 2 means 2s
- $Energy_{cost}$ may follow state energy price 30 cents/kWh
- $Penalty_{cost}^{abandone}$ is - $Profit$, abandonment rate follows: $5.8\% \times t \ (t > 2)$ (2) [1]
- $Penalty_{cost}^{delay}$ is regarded to "non-Active" states, regardless of execution time
  - $Penalty_{cost}^{delay} = \alpha + \beta \times t$ [2] , α is used to make sure $Penalty_{cost}^{delay}$ always greater than 0, t is counted when counter > 2s. Counter starts when a request arrives.
  - $\alpha = 5\% \ of \ Profit, \beta = 0.34 \Rightarrow maxPenalty_{cost}^{delay} = 10\% \ of \ profit$ [3] since (2) causes max $t_{abadone} \sim 15s$

Constraint

- Constraints for $M_u^a = [e_{i,j}], 1 \leq i \leq |V|, 1 \leq j \leq |T|$
  - Mỗi phần tử có 3 giá trị: -1, 0, 1

$$|e_{i,j}| \leq 1$$

  - Bảo toàn tổng số lượng container sau khi chuyển trạng thái

$$\sum_{j=1}^{|T|} e_{i,j} = 0, \forall\ 1 \leq i \leq |V|$$

- Constraints for $M^a = [e_{i,j}], 1 \leq i \leq |V|, 1 \leq j \leq |T|$
  - Đảm bảo số lượng container được chuyển trạng thái không vượt quá số container hiện tại.

$$|e_{i,j}| \leq |C_{i,j}|$$

Constraint

- Constraints for M^a (continue)
  - Đảm bảo hành động không gây tràn RAM, CPU

$$\sum_{i=1}^{|V|}\sum_{j=1}^{|T|}\left(|C_{i,j}| + e_{ij}\right).RAM_j \leq RAM_{\text{server}} \times |\text{S}_{\text{run}}|$$

$$\sum_{i=1}^{|V|}\sum_{j=1}^{|T|}\left(|C_{i,j}| + e_{ij}\right).CPU_j \leq CPU_{\text{server}} \times |\text{S}_{\text{run}}|$$

  - Không thay đổi trạng thái của container đang active

$$e_{ij} = 0, \forall j = active$$

Optimization

Notation	Description
Request	
$\mathrm{Q} = \{\mathrm{Q_i}\}, i\epsilon\{1, \dots, n\}$	Set of requests
$\mathrm{Q_i} = \{L_i^Q, U_i^Q, P_i^Q\}, i\epsilon\{1, \dots, n\}$	$\mathrm{i^{th}}$ request, include: time to live L_i^Q, resources usage (RAM, CPU) U_i^Q and time to process P_i^Q
Edge	
$E = \{E_i\}$	Set of edge computers
$E_i = \{R_i^e, C_i^e\}$	$\mathrm{i^{th}}$ edge computer, include: RAM capacity R_i^e, CPU capacity C_i^e
Core	
$I = \{R^c, \mathrm{L^c}\}$	Set of physical devices at core network, include routers, links between routers
$\mathrm{R^c} = \{\mathrm{R_i^c}\}$	Set of routers at core network
$\mathrm{L^c} = \{\mathrm{L_{ij}^c}\}$	Link between $\mathrm{i^{th}}$ router and $\mathrm{j^{th}}$ router
Server	
$S = \{S_i\}$	Set of server at cloud
$S_i = \{R_i^s, C_i^s\}$	$\mathrm{i^{th}}$ server, include: RAM capacity R_i^s, CPU capacity C_i^s

Notation	Description
Container	
$P = \{P_{ij}^{s}\}, s \in \{N, L_0, L_1, L_2, A\}$	Set of pods are deployed $at\ j^{\text{th}}\ server$ and are s state, $s \in \{Null, Cold, WarmDisk, WarmCPU, Active\}$
$C = \{C_i^j\}$	Set of processings change from i state to j state
$C_i^j = \{D_i^j, R_i^j, E_i^j\}$	Processing change from i state to j state, include delay D_i^j, RAM available is required to change state R_i^j, and energy E_i^j
Variable	
$\alpha_i(P_{jk}) \in \{0, 1\}$	$If\ i^{th}$ request is mapped to j^{th} pod at k^{th} server
$\beta(P_{ij}^{s}) \in \{0, 1\}$	$If\ i^{th}$ pod at k^{th} server are s state
$\delta(C_i^j) \in \{0, 1\}$	If processing change from i state to j state

Resource consumption

		Edge	Cloud
RAM	Base		
	Null	0	0
	Cold	0	0
	Wcpu	0	0
	Wdisk		
	Active		
CPU	Base		
	Null	0	0
	Cold	0	0
	Wcpu	0	0
	Wdisk	0	0
	Active		

Resource consumption

- CPU utilization (cloud) => number of requests are Active * CPU + base

$$C_j^{us} = \sum_{i=1}^{P} \beta(P_{ij}^A) * C_{P_{ij}^A} + C_j^{bs}$$

  - C_j^{us}: CPU usage of j^{th} server
  - $\beta(P_{ij}^A)$: If i^{th} pod at j^{th} server is Active state
  - $C_{P_{ij}^A}$: CPU usage of i^{th} pod at j^{th} server that is Active state
  - C_j^{bs}: CPU base of j^{th} server

Resource consumption

- RAM utilization (cloud)

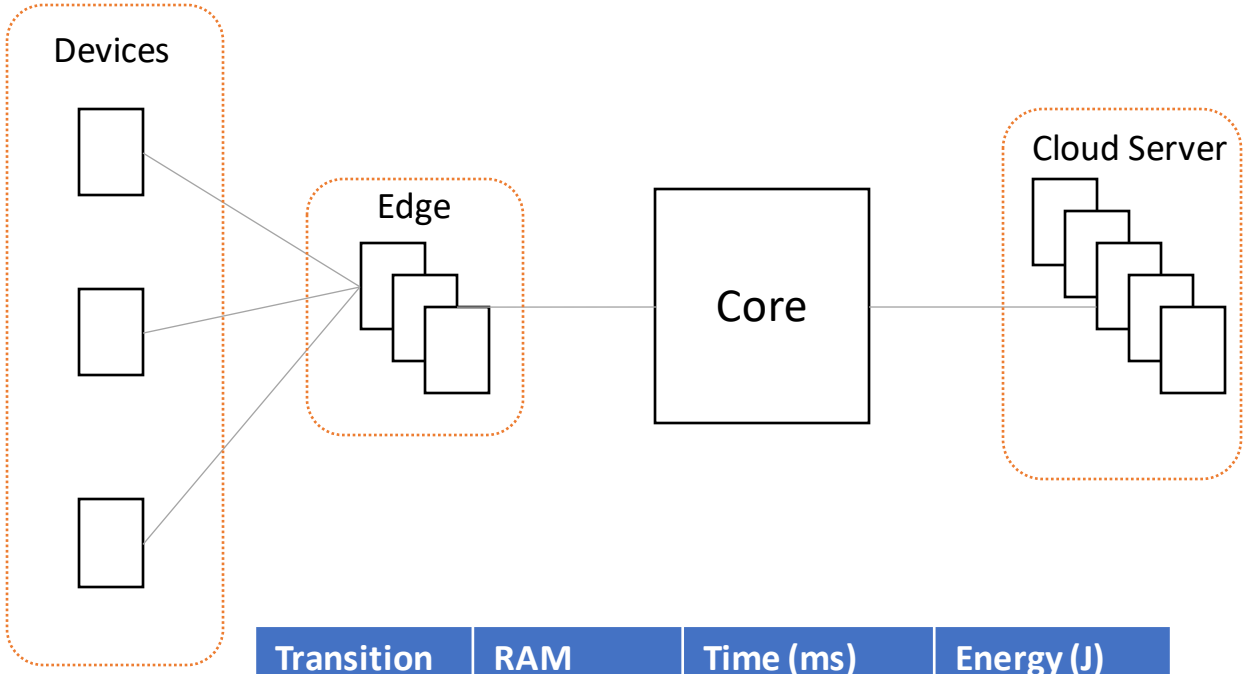
=> number of requests are Wcpu or Active * RAM + base

$$R_j^{us} = \sum_{i=1}^{P} \beta(P_{ij}^{A}) * R_{P_{ij}^{A}} + \sum_{i=1}^{P} \beta(P_{ij}^{L_2}) * R_{P_{ij}^{L_2}} + R_j^{bs}$$

- R_j^{us}: RAM usage of j^{th} server
- $\beta(P_{ij}^{A})$: If i^{th} pod at j^{th} server is Active state
- $\beta(P_{ij}^{L_2})$: If i^{th} pod at j^{th} server is WarmCPU state
- $R_{P_{ij}^{A}}$: RAM usage of i^{th} pod at j^{th} server that is Active state
- $R_{P_{ij}^{L_2}}$: RAM usage of i^{th} pod at j^{th} server that is WarmCPU state
- R_j^{bs}: RAM base of j^{th} server

Delay

- Delay network = {Delay D-E, Delay E-C, Delay routing, Delay C-S}
- Delay at edge (queue)
- Delay change state
- TTP (Time to processing)



Transition	RAM required (GB)	Time (ms)	Energy (J)
Null -> L0	0	5000	50
L0 -> L1	0	50000	2000
L1 -> L2	0.9	7000	100
L2 -> Act	1.2	50	0
Act -> L2	0.9	0	0
L2-> L1	0	30000	400
L1 -> L0	0	2000	50
L0 -> Null	0	1400	50

Delay

- Delay network = {Delay D-E, Delay E-C, Delay routing, Delay C-S}

$$D_{ij}^{n} = \cdots$$

  - Delay that i^{th} request mapping to j^{th} pod at k^{th} server

- Delay change state to Active

$$D_{ij}^{c} = \beta(P_{ij}^{s}) * \sum_{k=s+1}^{k=A} \left[\delta\left(C_{s}^{k}\right) * D_{s}^{k}\right], k\epsilon\Omega(s)$$

  - Delay that i^{th} pod at j^{th} server change from s state to active

- TTP (Time to processing): P_{i}^{Q}

=> $D_{ij}^{k} = \alpha_{i}\left(P_{jk}\right) * [D_{ij}^{n} + D_{j}^{c} + P_{i}^{Q}]$

  - D_{ij}^{k} : Delay that i^{th} request mapping to j^{th} pod at k^{th} server and change state to handle request
  - $\alpha_{i}\left(P_{jk}\right)$: If i^{th} request mapped to j^{th} pod at k^{th} server

Objective

- Max request accepted when n requests arrive

$$Maximize\ Z = \sum_{i\ \epsilon\ \Omega_Q, j\ \epsilon\ \Omega_P, k\ \epsilon\ \Omega_S} \alpha_i\left(P_{jk}\right)$$

- Min delay (sub objective)

$$Minimize\ D_{ij}^{k}$$

Constraints

- CPU usage $\leq$ Capacity

$$C_i^{us} \leq C_i^s$$

- RAM usage $\leq$ Capacity

$$R_i^{us} \leq R_i^s$$

- Time to change state + TTP + Delay Network $\leq$ TTL

$$L_i^Q \geq D_{ij}^k$$