Optimal Migration Contracts in Virtual Networks: Pay-as-You-Come vs Pay-as-You-Go

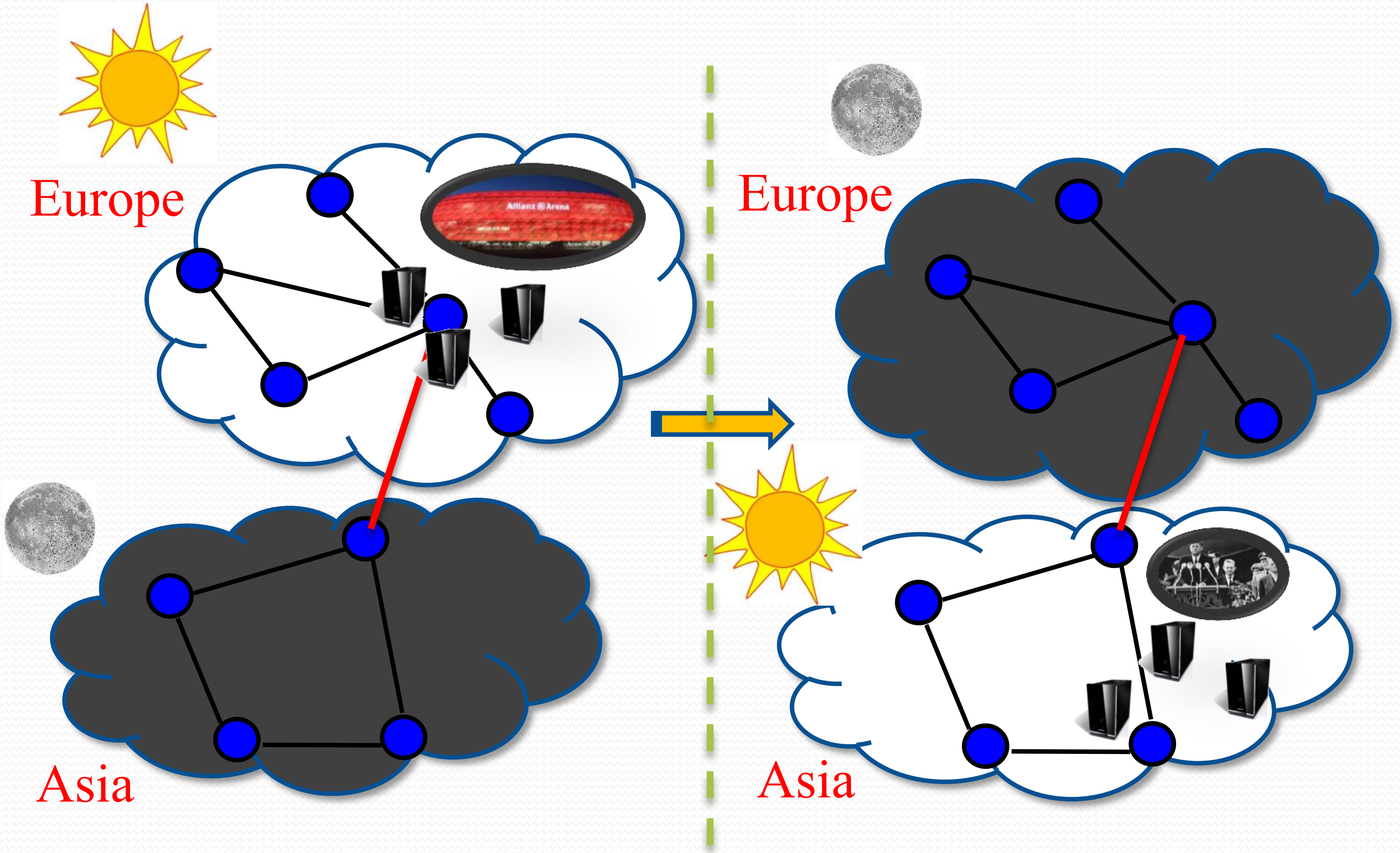
Xinhui Hu, Stefan Schmid’, Andrea Richa* and Anja Feldmann’*

*Arizona State University
’Telekom Innovation Laboratories & TU Berlin

Outline

- *Motivation*
- Related Work
- System Model and Problem Formulation
- Algorithm
- Performance Evaluation
- Conclusion

Motivation



Virtual Networking of Cloud Resources

Research Challenges

- When and where to migrate a service?

- Offline/online algorithm!
  - Offline: every day the same
  - Online: no knowledge of the future requests

- Economical dimension:
  - Migration comes at costs: contracts!

Our Perspective

- Migration contracts
  - Contracts with more bandwidth or longer duration are cheaper! (discounts)
  - Which contract (bandwidth, duration) to buy?
- Objective
  - Find the *optimal migration contracts* in virtual networks for two pricing models:
    - Pay-as-You-Come: “pay in advance even if not needed”
    - Pay-as-You-Go: “pay for what you use only”

Contribution

- We present *two optimal offline algorithms* for migration contracts in virtual networks for Pay-as-You-Come and Pay-as-You-Go pricing models (PAYC and PAYG)

- We present *two online algorithms* for Pay-as-You-Come and Pay-as-You-Go pricing models (ONC and ONG)

Outline

- *Motivation*
- *Related Work*
- System Model and Problem Formulation
- Algorithm
- Performance Evaluation
- Conclusion

Related Work

- Online Model but without Economics:

D. Arora, M. Bienkowski, A. Feldmann, G. Schaffrath, and S. Schmid, "**Online strategies for intra and inter provider service migration in virtual networks**," Proc. IPTComm, 2011

- CloudNet Prototype (with NTT DoCoMo):

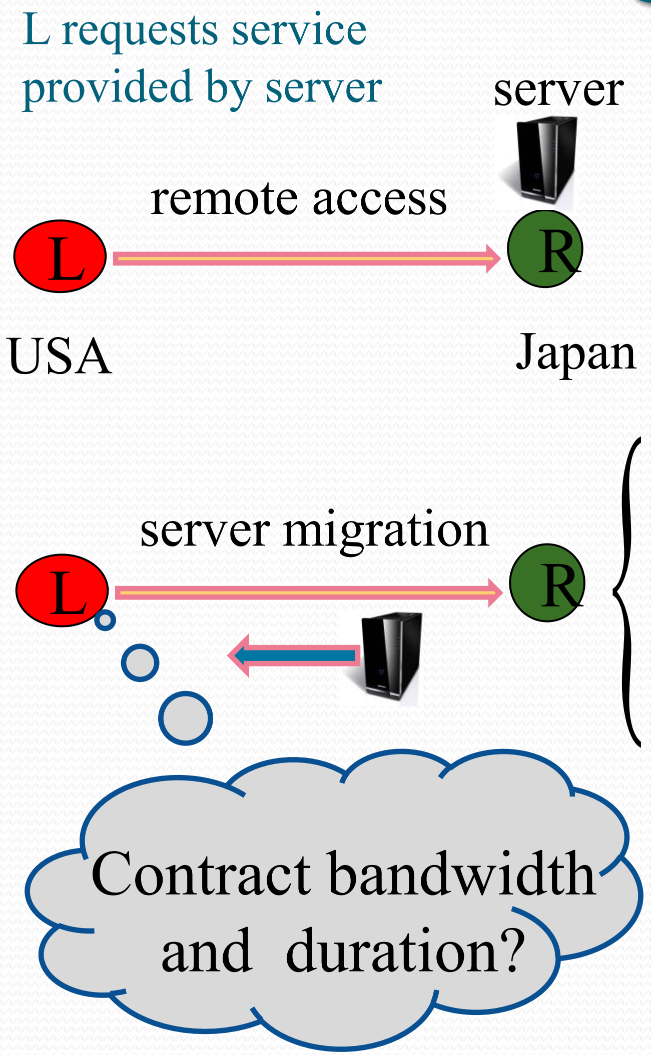
G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, "**Network virtualization architecture: proposal and initial prototype**," Proc. ACM VISA, 2009.

- Much more literature on economical aspects of cloud pricing

Outline

- *Motivation*
- *Related Work*
- *System Model and Problem Formulation*
- Algorithm
- Performance Evaluation
- Conclusion

Cost Model



L requests service provided by server

server

remote access

L

R

USA

Japan

server migration

L

R

Contract bandwidth and duration?

- **Access cost**: latency associated with satisfying request remotely
- **Migration cost**: cost of migrating server from current location to location of request (cost = service interruption time, depends on bandwidth)
- **Contract cost**: cost of buying/renting resources (different discounts are provided for different contract bandwidths and durations)

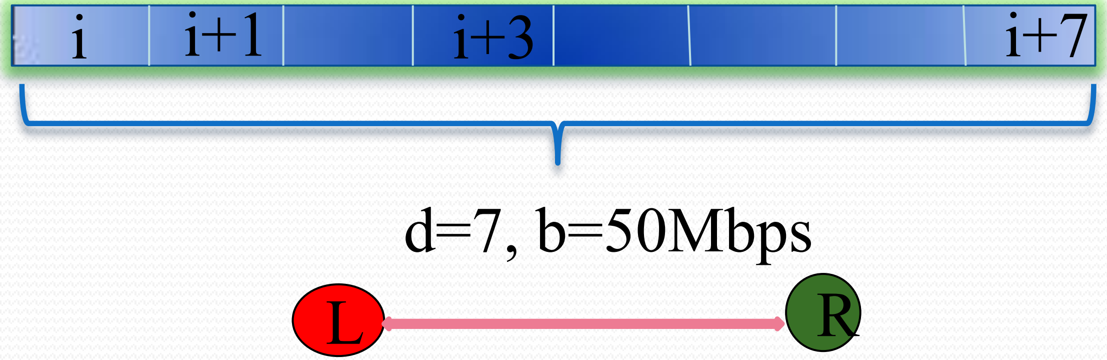
Pricing Model

- **Pay-as-You-Come**: pay for the resource in advance, before resource is utilized
  - if not used, it's your fault! (like MPLS ☺)
- **Pay-as-You-Go**: pay for the resource after resource is utilized
  - only if actually utilized (like EC2)

Data Model

- **Contract durations**: $\tilde{D} = \{d_1, d_2, \ldots, d_k\}, d_1 \leq d_2 \leq \ldots \leq d_k$
- **Contract bandwidths**: $\tilde{B} = \{b_1, b_2, \ldots, b_q\}, b_1 \leq b_2 \leq \ldots \leq b_q$
- **Discount function**: f
  - Linear
  - Non-linear: e.g., sqrt, log, …
  - For example, a twice as long contract may cost only 50% more, and doubling the reserved bandwidth may cost only 30% more.
- **Request sequence**: $<r_1, t_1>, <r_2, t_2>, \ldots$
  - $<r_i, t_i>$ represents the i_{th} request from r_i at time t_i
- **Two sites**: L (e.g., USA) and R (e.g., Japan)

An example



Problem Formulation

Given requests <r_1, r_2, ..., r_n> at respective times <t_1, t_2, ..., t_n>, we aim to

minimize $\text{Cost} = \text{AccCost} + \text{MigCost} + \text{ConCost}$

where

- Access cost: $\text{AccCost} = \sum_i D[r_i, s_i]$
- Migration cost: $\text{MigCost} = \sum_i S \cdot D[s_{i-1}, s_i]/b_i,$
- Contract cost:
  - Pay-as-You-Come: $\text{ConCost} = \sum_i f(d_i, b_i)$
  - Pay-as-You-Go: $\text{ConCost} = f(\mu, b_i)$

Outline

- *Motivation*
- *Related Work*
- *System Model and Problem Formulation*
- *Algorithm*
- Performance Evaluation
- Conclusion

Main Results

- Optimal algorithms (based on *novel dynamic programming* approaches)
  - PAYC for Pay-as-You-Come pricing model in $O(n^2(n+kq))$
  - PAYG for Pay-as-You-Go pricing model in $O(qn^3)$

k = # of available durations
q = # of available bandwidths

- Online algorithms
  - ONC for Pay-as-You-Come pricing model
  - ONC for Pay-as-You-Go pricing model
- Experimental evaluation

Data structure for PAYC

Dynamic Progr. Matrix	Description
$C_{n\times n\times 4}$	Total cost matrix in PAYC, an entry *C[i,j,k]*, where *i,j* ∈[*n*] and *k* ∈{*(s, s')*|*s, s'* ∈{L, R}}, denotes the minimum total cost for satisfying all requests from *ri* to *rj* for a scenario where at the beginning of the *i*th request the server is at node *s* and at the end of request *j* the server is at node *s'*
$(AM_m)_{n\times n\times 4}$	Combined access cost and migration cost matrix for bandwidth b_m, AM_m[i, j,(s,s')] stores the combined access and migration costs for the best migration strategy that satisfies the sequence of requests from *ri* to *rj*

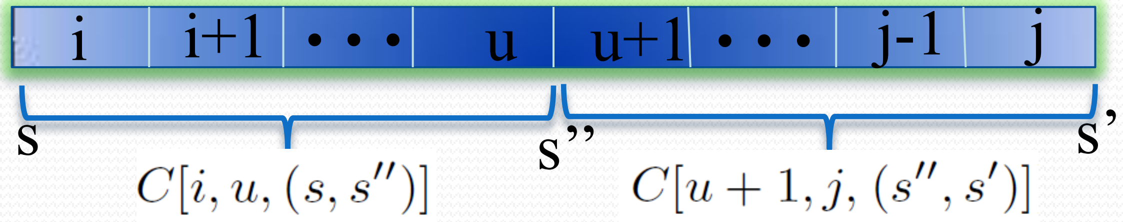
PAYC: Satisfying one request

- service request from node s at time t, and server size S
- If the server is also located at s then no cost
- Else, either pay
  - access cost or
  - pay server migration cost S/bm and d_1-day contract cost $f(d_1,bm)$, for some bandwidth bm and smallest contract duration d_1

PAYC: Satisfying multiple requests

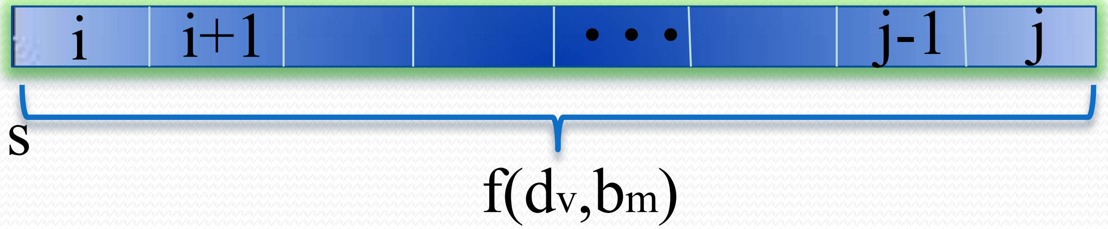
multiple requests from time t_i to time t_j

Split the contract at u_{th} request



$$C[i,j,(s,s')] \leftarrow \min_{i\leq u<j;s''\in\{L,R\}}\{C[i,u,(s,s'')]+C[u+1,j,(s'',s')]\}$$

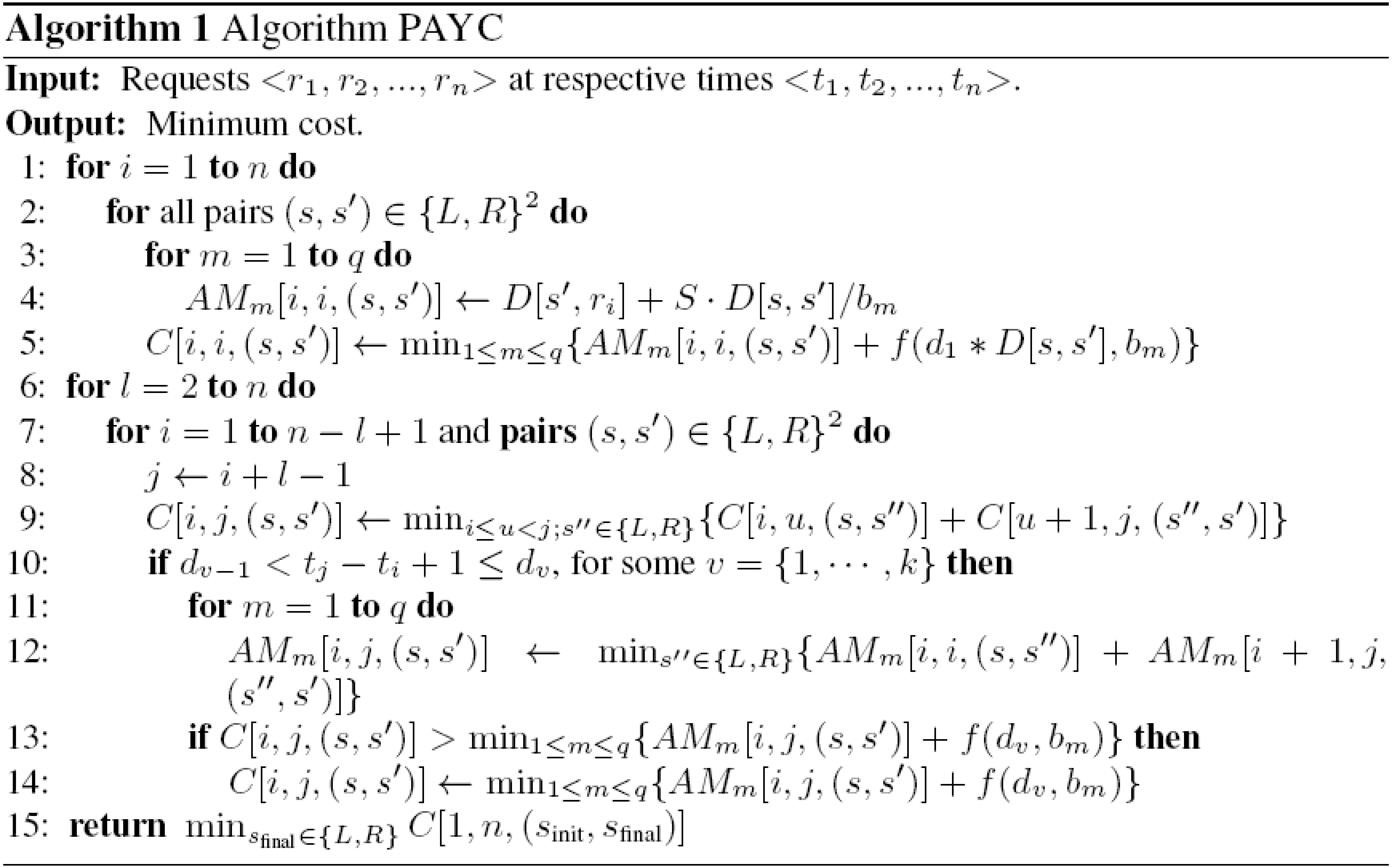
Buy a long contract d_v to cover the interval, where v ∈ [k]



$$AM_m[i,j,(s,s')] \leftarrow \min_{s''\in\{L,R\}}\{AM_m[i,i,(s,s'')]+AM_m[i+1,j,(s'',s')]\}$$

$$C[i,j,(s,s')] \leftarrow \min\{C[i,j,(s,s')],\ \min_{1\leq m\leq q}\{AM_m[i,j,(s,s')]+f(d_v,b_m)\}\ \}$$

Optimal algorithm: PAYC



Algorithm 1 Algorithm PAYC

Input: Requests $<r_1, r_2, ..., r_n>$ at respective times $<t_1, t_2, ..., t_n>$.

Output: Minimum cost.

```
for $i = 1$ to $n$ do
    for all pairs $(s, s') \in \{L, R\}^2$ do
        for $m = 1$ to $q$ do
            $AM_m[i, i, (s, s')] \leftarrow D[s', r_i] + S \cdot D[s, s']/b_m$
        $C[i, i, (s, s')] \leftarrow \min_{1 \le m \le q}\{AM_m[i, i, (s, s')] + f(d_1 * D[s, s'], b_m)\}$
for $l = 2$ to $n$ do
    for $i = 1$ to $n - l + 1$ and pairs $(s, s') \in \{L, R\}^2$ do
        $j \leftarrow i + l - 1$
        $C[i, j, (s, s')] \leftarrow \min_{i \le u < j; s'' \in \{L,R\}}\{C[i, u, (s, s'')] + C[u + 1, j, (s'', s')]\}$
        if $d_{v-1} < t_j - t_i + 1 \le d_v$, for some $v = \{1, \cdots, k\}$ then
            for $m = 1$ to $q$ do
                $AM_m[i, j, (s, s')] \leftarrow \min_{s'' \in \{L,R\}}\{AM_m[i, i, (s, s'')] + AM_m[i + 1, j, (s'', s')]\}$
            if $C[i, j, (s, s')] > \min_{1 \le m \le q}\{AM_m[i, j, (s, s')] + f(d_v, b_m)\}$ then
                $C[i, j, (s, s')] \leftarrow \min_{1 \le m \le q}\{AM_m[i, j, (s, s')] + f(d_v, b_m)\}$
return $\min_{s_{\text{final}} \in \{L,R\}} C[1, n, (s_{\text{init}}, s_{\text{final}})]$
```

Data structure for PAYG

Dyn. Progr. Matrix	Description
$C_{n\times n\times 4}$	Total cost matrices in PAYC, an entry *C[i,j,k],* where *i,j* ∈[*n*] and *k* ∈{(*s, s'*)|*s, s'* ∈{L, R}}, denotes the minimum total cost for satisfying all requests from *ri* to *rj* for a scenario where at the beginning of the *i*th request the server is at node *s* and at the end of request *j* the server is at node *s'*
$(A_m)_{n\times n\times 4}$	Access cost matrix for bandwidth *bm*
$(N_m)_{n\times n\times 4}$	Number of migrations matrix for bandwidth *bm*

PAYG: One request

Similar dynamic programming approach as PAYC is used in PAYG. The main difference is how to update the cost for multiple requests.

Initialization:

Access cost: $A_m[i, i, (s, s')] \leftarrow D[s', r_i]$

$m \in [q]$

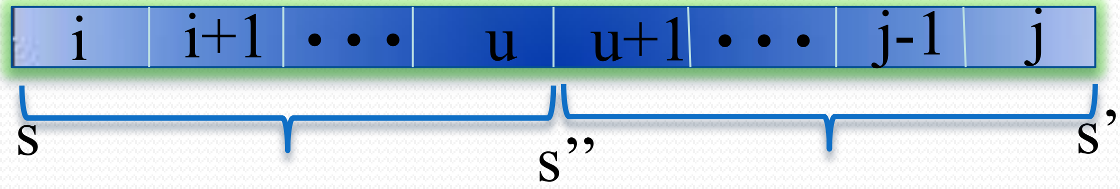
Migration cost: $N_m[i, i, (s, s')] \leftarrow D[s, s']$

Total cost: $C_m[i, i, (s, s')] \leftarrow A_m[i, i, (s, s')] + S \cdot N_m[i, i, (s, s')]/b_m + f(D[s, s'], b_m)$

PAYG: Multiple requests

multiple requests from time t_i to time t_j

Split the contract at uth request

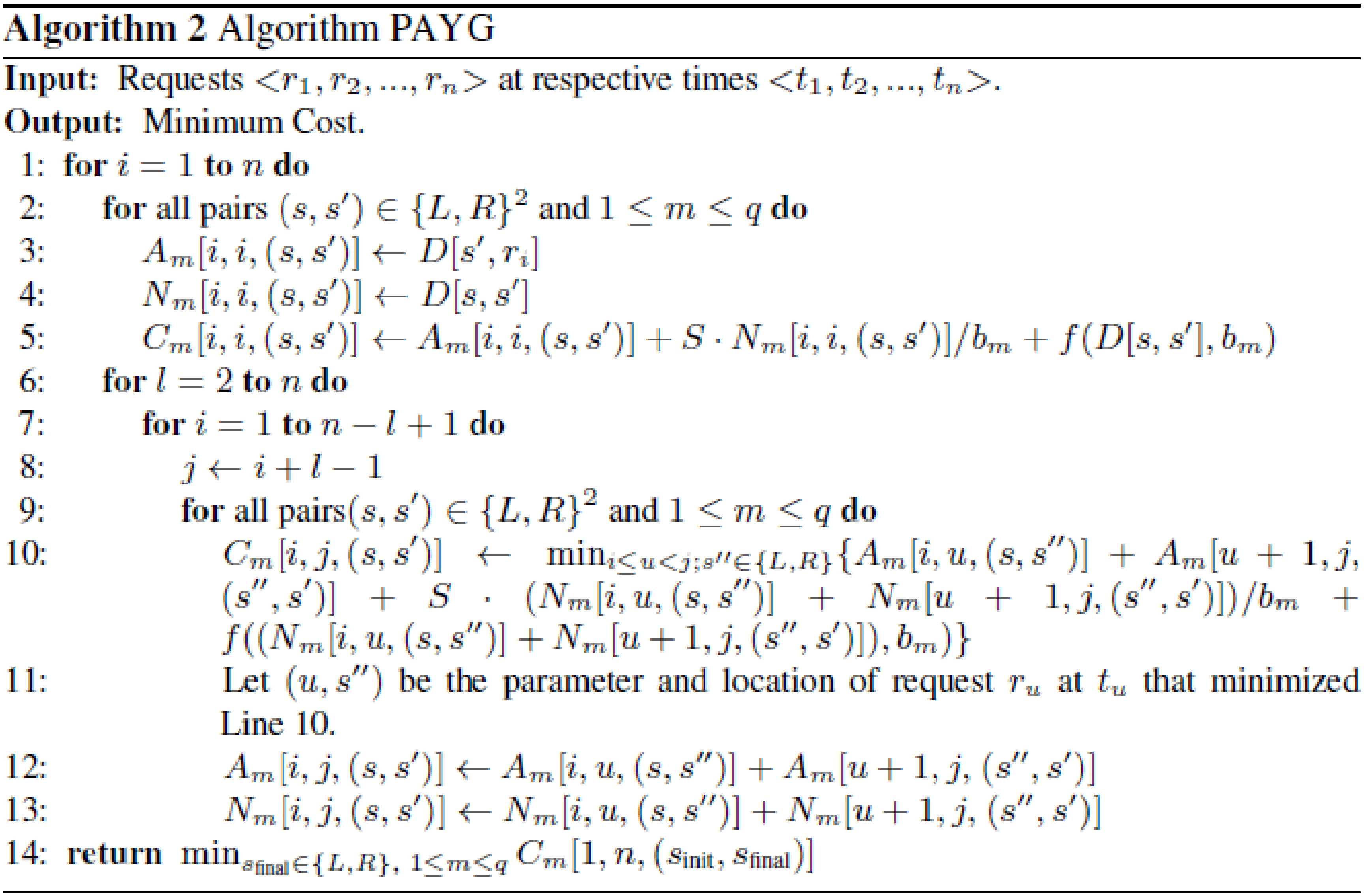


$$C_m[i,j,(s,s')] \leftarrow \min_{i \le u < j; s'' \in \{L,R\}} \{A_m[i,u,(s,s'')] + A_m[u+1,j,(s'',s')] + S \cdot (N_m[i,u,(s,s'')] + N_m[u+1,j,(s'',s')])/b_m + f((N_m[i,u,(s,s'')] + N_m[u+1,j,(s'',s')]), b_m)\}$$

The optimal result is given by:

$$\min_{s_{\text{final}} \in \{L,R\},\ 1 \le m \le q} C_m[1,n,(s_{\text{init}},s_{\text{final}})]$$

Optimal algorithm: PAYG



Algorithm 2 Algorithm PAYG

Input: Requests $<r_1, r_2, ..., r_n>$ at respective times $<t_1, t_2, ..., t_n>$.

Output: Minimum Cost.

```
 1: for $i = 1$ to $n$ do
 2:     for all pairs $(s, s') \in \{L, R\}^2$ and $1 \leq m \leq q$ do
 3:         $A_m[i, i, (s, s')] \leftarrow D[s', r_i]$
 4:         $N_m[i, i, (s, s')] \leftarrow D[s, s']$
 5:         $C_m[i, i, (s, s')] \leftarrow A_m[i, i, (s, s')] + S \cdot N_m[i, i, (s, s')]/b_m + f(D[s, s'], b_m)$
 6:     for $l = 2$ to $n$ do
 7:         for $i = 1$ to $n - l + 1$ do
 8:             $j \leftarrow i + l - 1$
 9:             for all pairs$(s, s') \in \{L, R\}^2$ and $1 \leq m \leq q$ do
10:                 $C_m[i, j, (s, s')] \leftarrow \min_{i \leq u < j; s'' \in \{L,R\}} \{A_m[i, u, (s, s'')] + A_m[u + 1, j, (s'', s')] + S \cdot (N_m[i, u, (s, s'')] + N_m[u + 1, j, (s'', s')])/b_m + f((N_m[i, u, (s, s'')] + N_m[u + 1, j, (s'', s')]), b_m)\}$
11:                 Let $(u, s'')$ be the parameter and location of request $r_u$ at $t_u$ that minimized Line 10.
12:                 $A_m[i, j, (s, s')] \leftarrow A_m[i, u, (s, s'')] + A_m[u + 1, j, (s'', s')]$
13:                 $N_m[i, j, (s, s')] \leftarrow N_m[i, u, (s, s'')] + N_m[u + 1, j, (s'', s')]$
14: return $\min_{s_{\text{final}} \in \{L,R\},\ 1 \leq m \leq q} C_m[1, n, (s_{\text{init}}, s_{\text{final}})]$
```

Online algorithm: ONC

- We do an amortization by migrating only when the access cost(C) exceeds the migration cost.
- If no contract is available for current migration, ONC checks if a longer contract would have been better for the past requests.
- Otherwise, ONC checks whether a shorter contract should be chosen.

Online algorithm: ONG

- A counter C_1 records the number of the migrations performed so far while C_2 denotes the total access costs.
- If the access cost C_2 reaches the migration cost plus marginal migration contract costs (i.e., $f(C_1+\mathbf{1}, b)-f(C_1, b)$, for bandwidth b), ONG migrates the server, increments counter C_1, and resets counter C_2.

Outline

- *Motivation*
- *Related Work*
- *System Model and Problem Formulation*
- *Algorithm*
- *Performance Evaluation*
- Conclusion

Simulation Setting

- **Duration set**: D={1, 30, 60, 100}
- **Bandwidth set** (Mbps): B={50, 100}
- **Server size**: S = 250M
- **Unit access cost**: 5
- **Request number**: n=1500 requests
- **Discount function**:

$$f_{lin} = 1.5^{\log d_i + b_j/50-1} \cdot 6$$

$$f_{sqrt} = \sqrt{d_i b_j / 50} \cdot 6$$

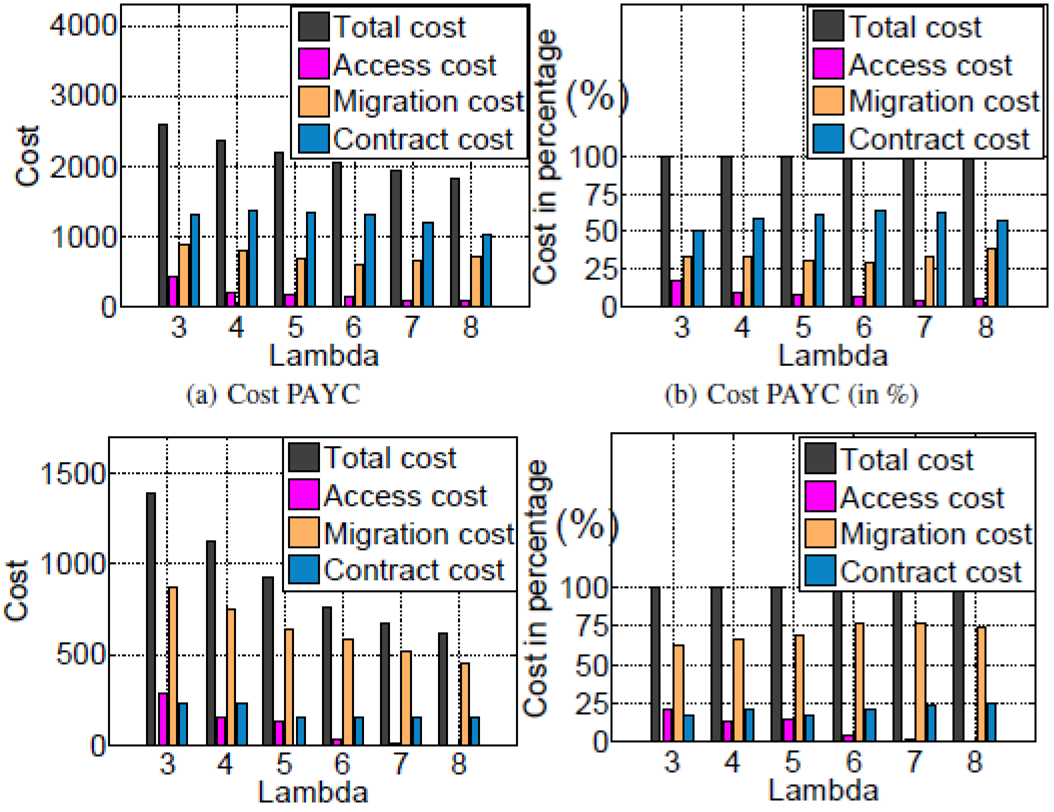
$$f_{\log} = \log(d_i b_j / 50) \cdot 6$$

Simulation Setting

- Request model:

• Requests alternate between the two virtual sites
• Markov process: stay at current site according to an exponential distribution with parameter λ, then change with probability p

Cost distribution for PAYC and PAYG



(a) Cost PAYC

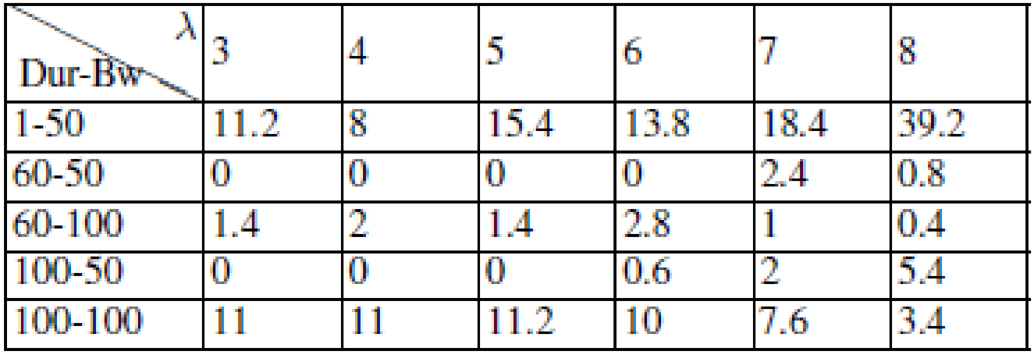
(b) Cost PAYC (in %)

(c) Cost PAYG

(d) Cost PAYG (in %)

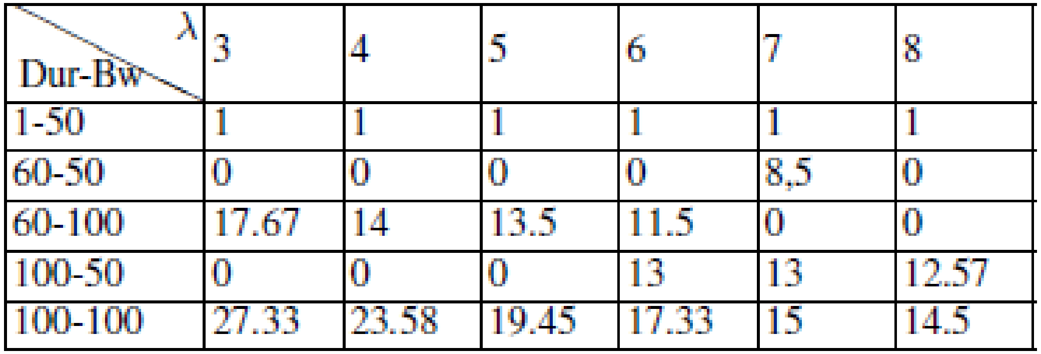
Contract Distribution

Table 1. Distribution of purchased contracts (discount function f_{lin}).



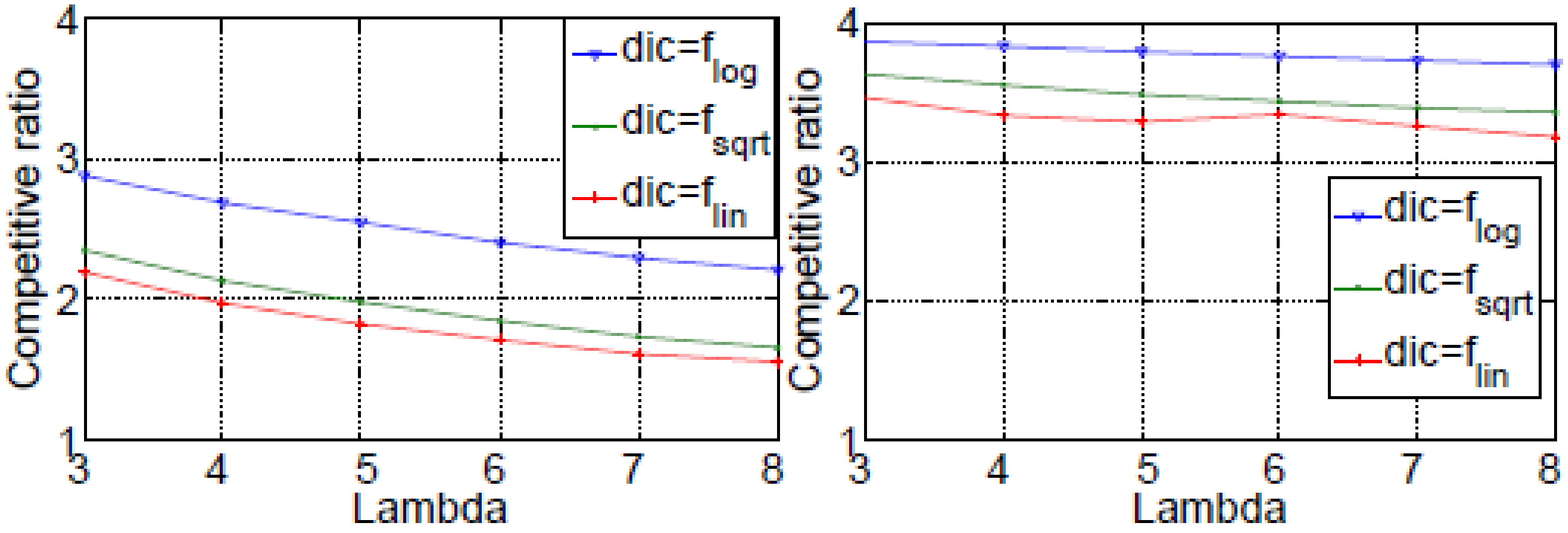
λ Dur-Bw	3	4	5	6	7	8
1-50	11.2	8	15.4	13.8	18.4	39.2
60-50	0	0	0	0	2.4	0.8
60-100	1.4	2	1.4	2.8	1	0.4
100-50	0	0	0	0.6	2	5.4
100-100	11	11	11.2	10	7.6	3.4

Table 2. Number of migrations for each contract (discount function f_{lin}).



λ Dur-Bw	3	4	5	6	7	8
1-50	1	1	1	1	1	1
60-50	0	0	0	0	8,5	0
60-100	17.67	14	13.5	11.5	0	0
100-50	0	0	0	13	13	12.57
100-100	27.33	23.58	19.45	17.33	15	14.5

Competitive ratio



(a) Competitive ratio for ONC.

(b) Competitive ratio for ONG.

Outline

- *Motivation*
- *Related Work*
- *System Model and Problem Formulation*
- *Algorithm*
- *Performance Evaluation*
- *Conclusion*

Conclusion and Future Work

- Conclusion
    - We have studied *online migration in virtual networks from an economical perspective and in two pricing models: Pay-as-You-Come and Pay-as-You-Go*
    - We have presented *optimal algorithms for each pricing model*
    - We have discussed *online algorithms for each pricing model*
- Future Work
    - Extend to live migration
    - Extend to more complicated virtual networks (more than two sites

THANK YOU!

Questions ?

Backup slides

Optimal algorithm: PAYC

One request: request from r_i at time t_i

Pay access cost if request is from remote location (r_i !=s')

If s != s' then, for bandwidth b_m, pay migration cost S/b_m and d_1-day contract cost f(d_1,b_m)

time t_i, request from r_i

s

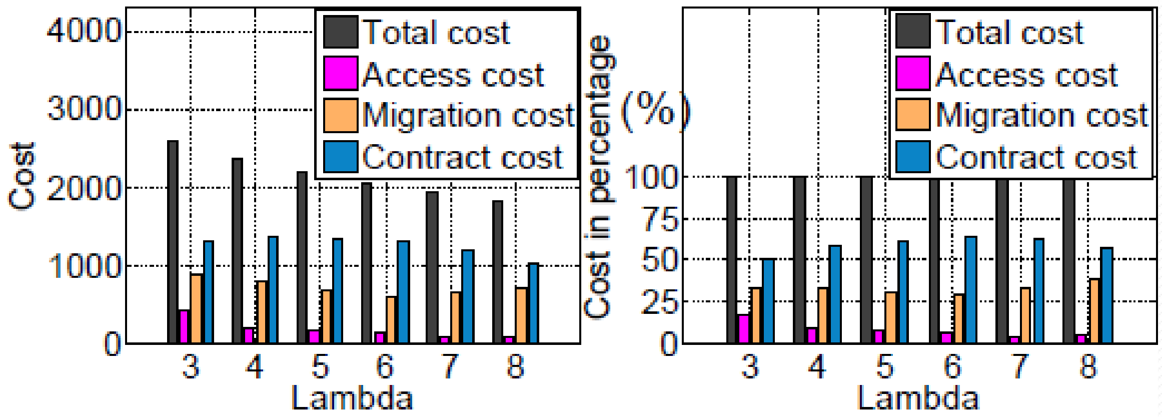
Server location s'

Access+Migration cost: $AM_m[i,i,(s,s')] \leftarrow D[s',r_i] + S \cdot D[s,s']/b_m$ m ∈[q]

Total cost: $C[i,i,(s,s')] \leftarrow \min_{1\leq m\leq q}\{AM_m[i,i,(s,s')] + f(d_1 * D[s,s'], b_m)\}$

Cost distribution for PAYC

- Observation
  - The total cost and the access cost decrease for larger lambda
  - The migration and contract stay much more stable
- Reason
  - Requests originating from one site for longer time periods render it worthwhile to migrate and buy longer contracts

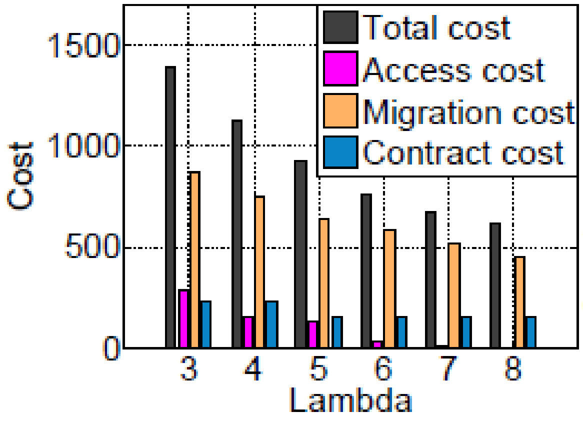


(a) Cost PAYC

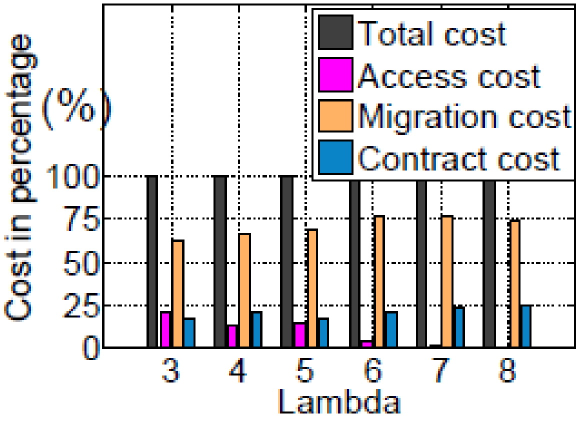
(b) Cost PAYC (in %)

Cost distribution for PAYG

- Observation:
  - The total cost is lower than that of PAYC
  - The migrations constitute a larger share of the overall costs
- Reason:
  - the contract cost is given by the number of migrations and the amount of leased bandwidth under the discount function



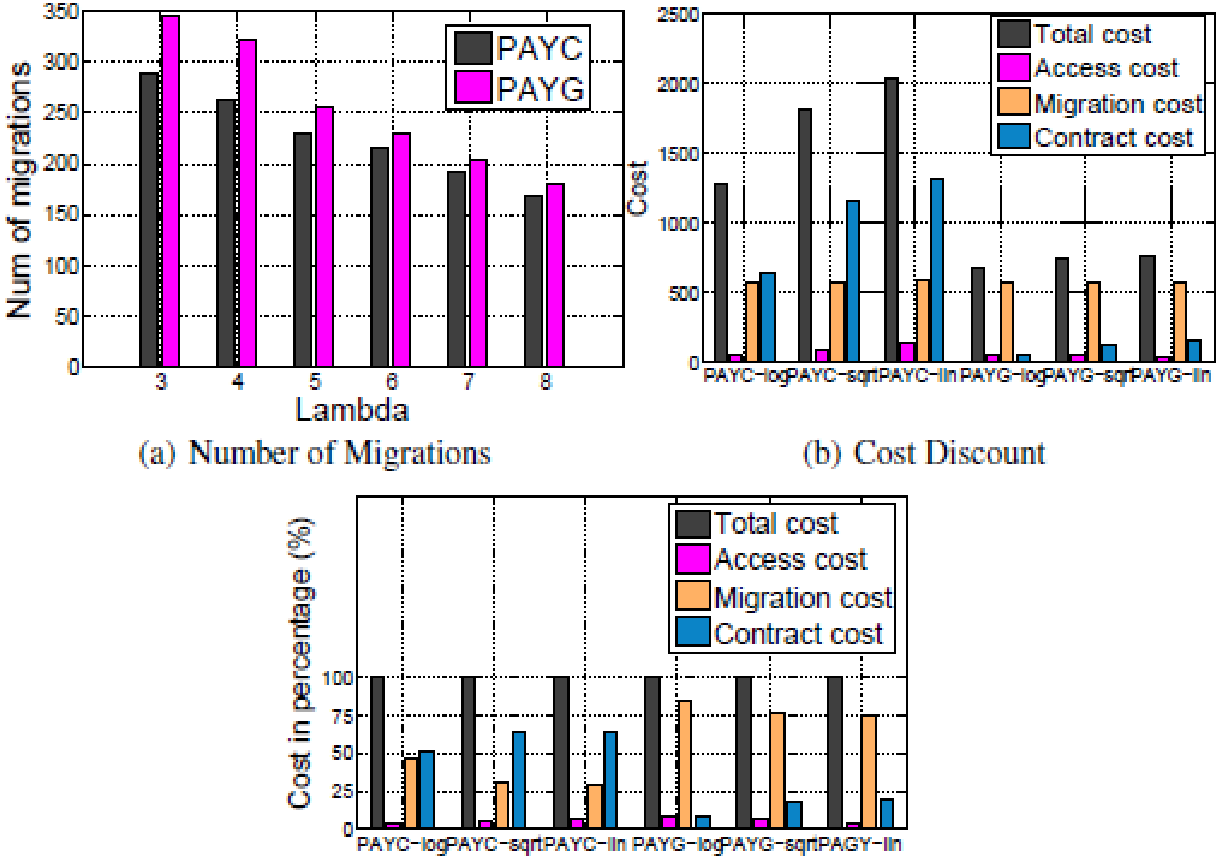
(c) Cost PAYG



(d) Cost PAYG (in %)

Number of migrations and effect of discount function

- The experiments are derived under flin discount function



(a) Number of Migrations

(b) Cost Discount

(c) Cost Discount (in %)