Service Placement and Migration Mobile Edge Clouds

Kin K. Leung

Electrical & Electronic Engineering and Computing Departments

Imperial College

London, UK

www.commsp.ee.ic.ac.uk/~kkleung

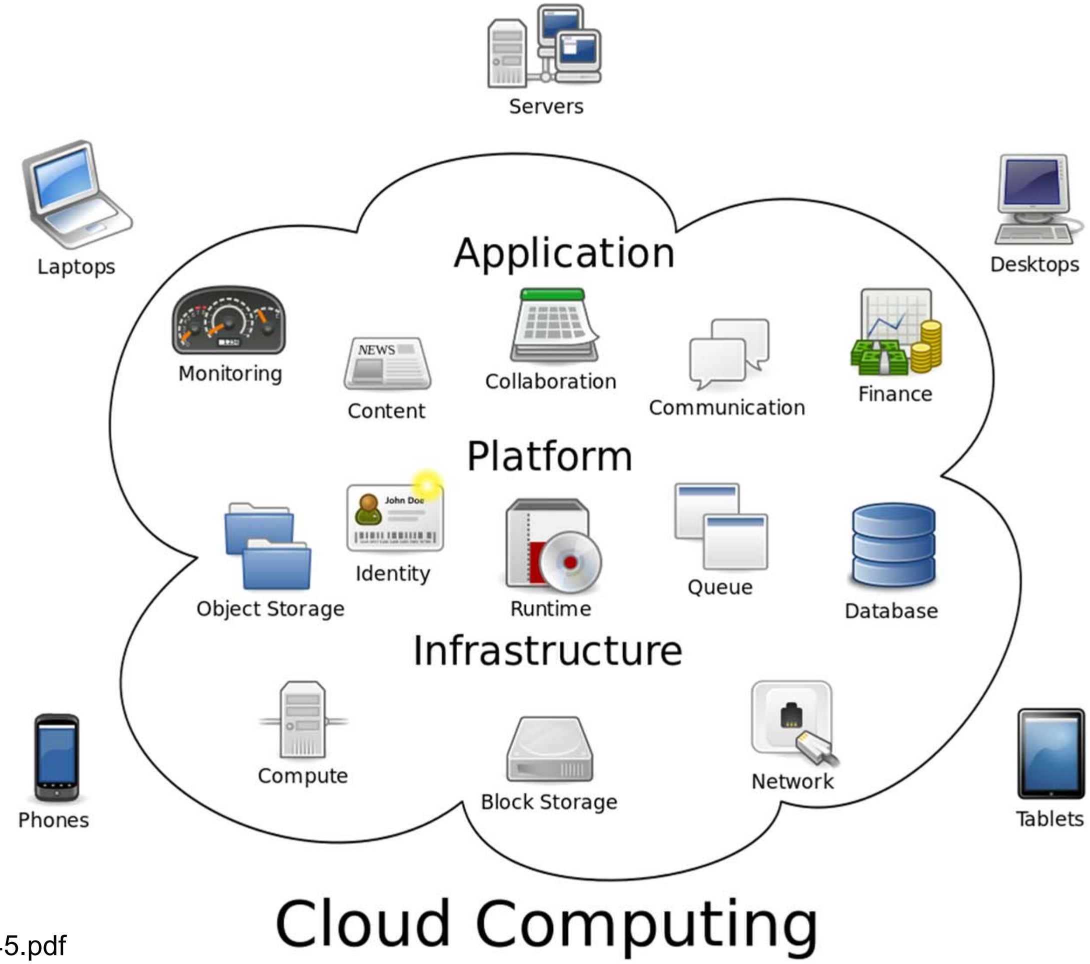
Joint work with Shiqiang Wang (IBM), Murtaza Zafer (Nyansa), Rahul Urgaonkar (Amazon), Ting He (Penn State Univ.), Kevin Chan (U.S. Army)

What Is a Cloud?

Imperial College London

Imperial College in August 2015

Example cloud-powered applications:
- Google map (map service)
- Dropbox (storage & content sharing)
- YouTube & Netflix (video streaming + encoding/decoding)



Source: https://en.wikipedia.org/wiki/Cloud_computing

The NIST definition:

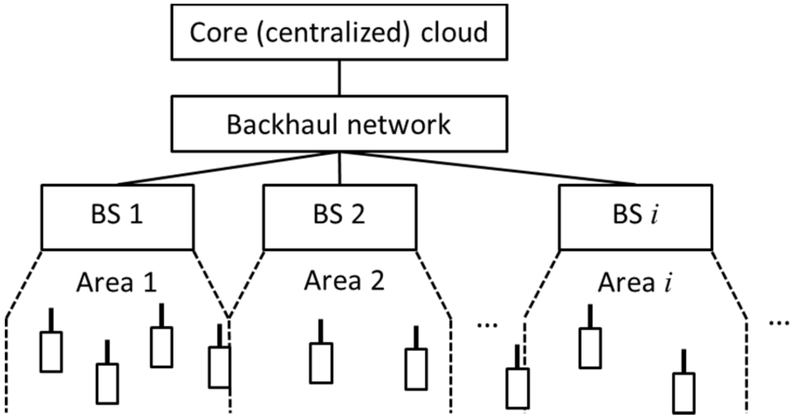
“**Cloud computing** is a **model for enabling** ubiquitous, convenient, on-demand **network access to a shared pool of configurable computing resources** (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

Source: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

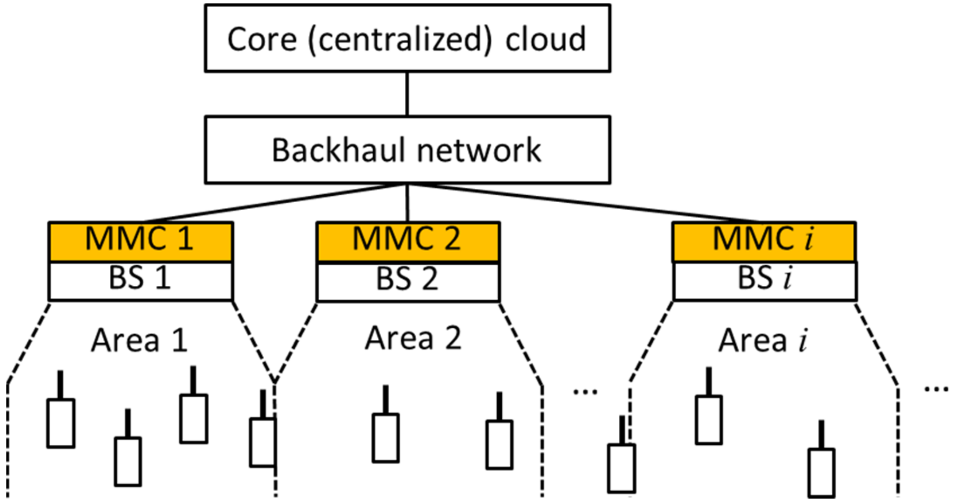
What Is a Mobile Micro-Cloud?

Imperial College London

Traditional cloud: Computation at the core (centralized) cloud



Mobile micro-cloud (MMC): Computation distributed across the core, edge & device



Benefits of MMC

- Reduce delay (beneficial for delay-sensitive applications)
- Reduce total communication bandwidth
- More secure due to limited information dissemination
- Increase availability and reliability in dynamic environments

Status Quo

- Commercial proposals (Nokia & IBM in 2013)
- Standardization - ETSI Industry Specification Group (ISG) for Mobile-edge Computing launched Sept. 2014
- Only preliminary research on MMCs exists in the literature

Why Delay Matters?

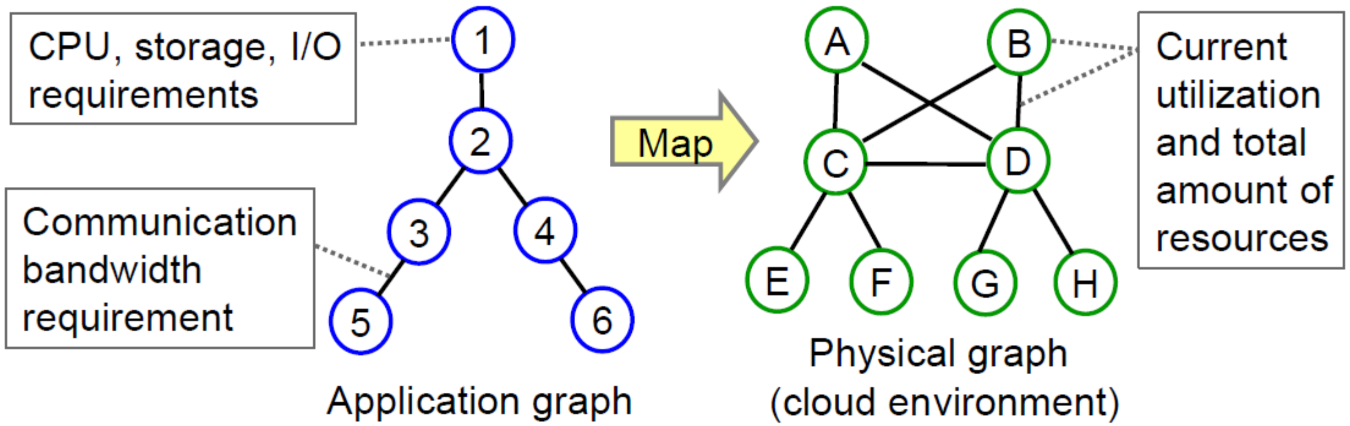
Imperial College London

Source: http://5glab.de

Service Placement Problem

A service (application) contains different connected components.

How to place and execute these components on the physical cloud system?



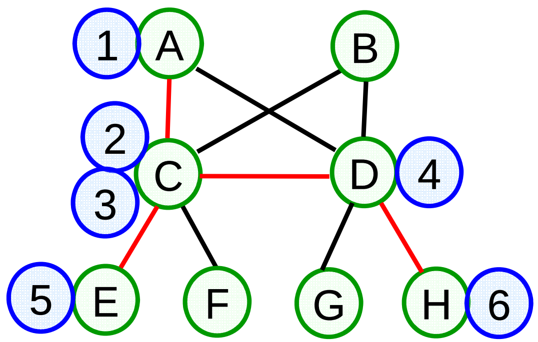
There are network connections among different servers in the cloud and also between core and micro-clouds.

Goals for service placement decisions

- Minimizing resource consumption
- Load balancing
- Ultimately: Maintain the quality of cloud services

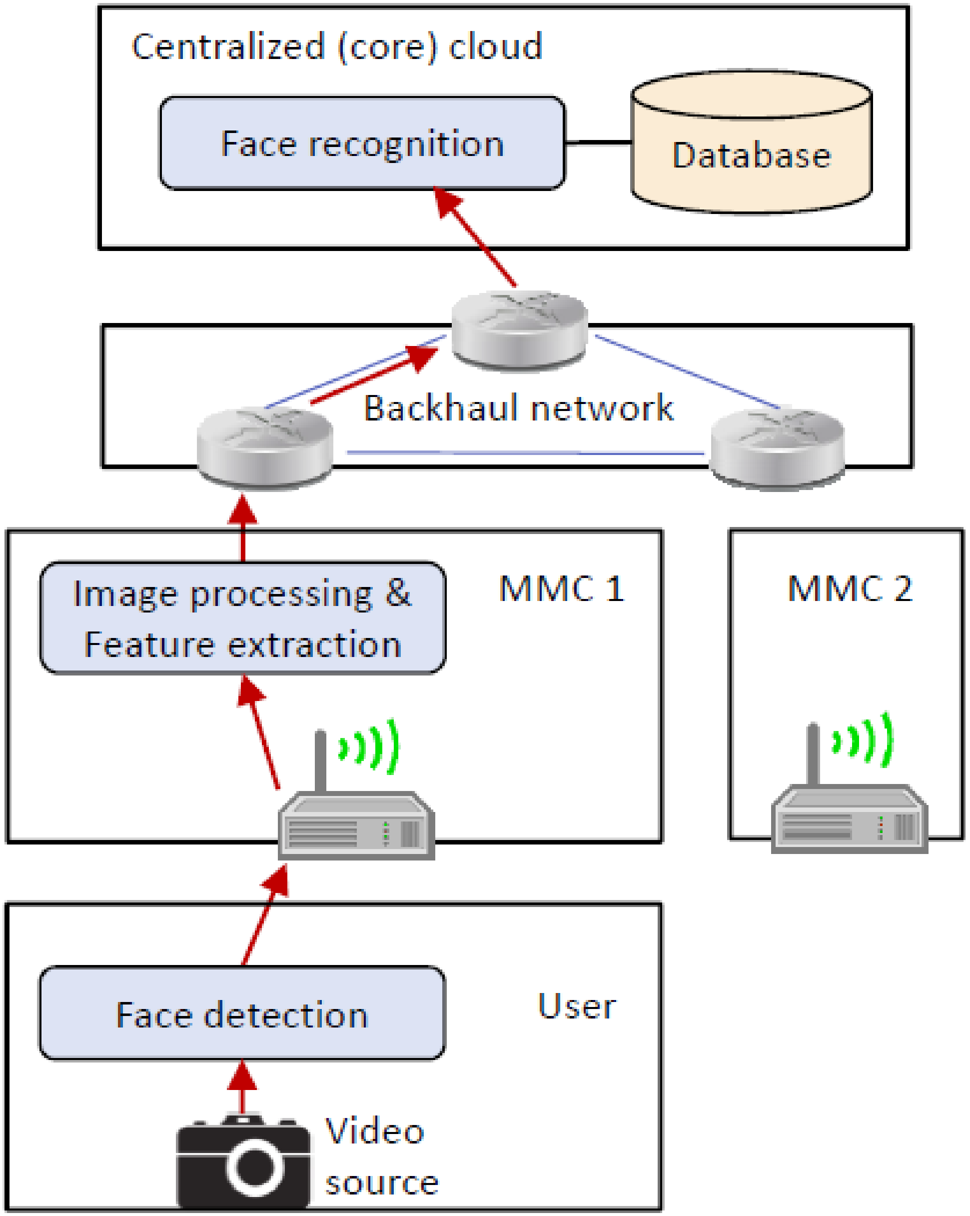
Challenges

- NP-hard in most cases
- User or network dynamics at network edge (unique for the distributed micro-cloud environment)



Example mapping result

A Face Recognition Example

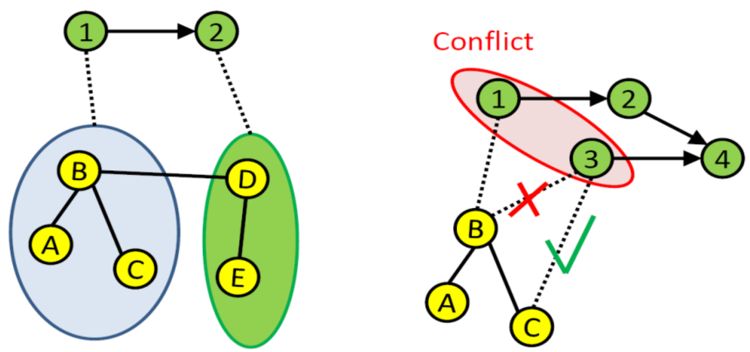


MILP Approach to Offline Placement

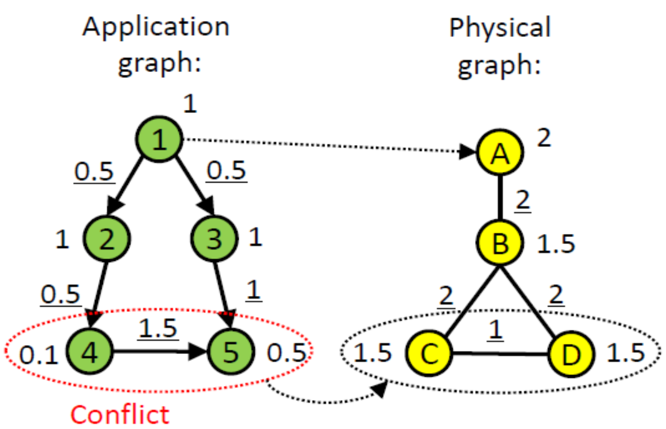
Imperial College London

Mixed-integer linear program (MILP) formulation

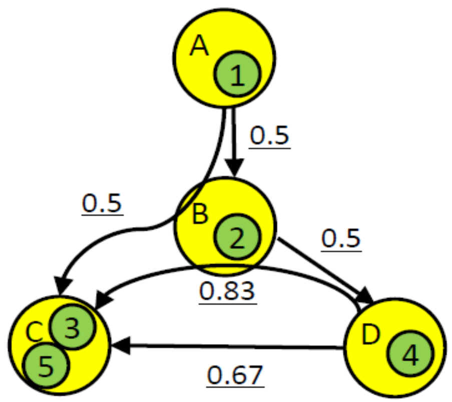
- Given: Resource requirements specified on app. graph
- Objective: Jointly consider total resource consumption and load balancing
- Constraints:
  - o Capacity constraints
  - o Domain and conflict constraints (e.g. for security)



Example Result



(a)



(b)

(a) Problem setting

(b) Mapping result

Why is it not ideal?

- The problem is NP-hard (even for simple cases), so the MILP approach gives no performance guarantee
- No straightforward extension to online service arrivals
- No mechanism to handle dynamic network variations

Objective function (not unique)

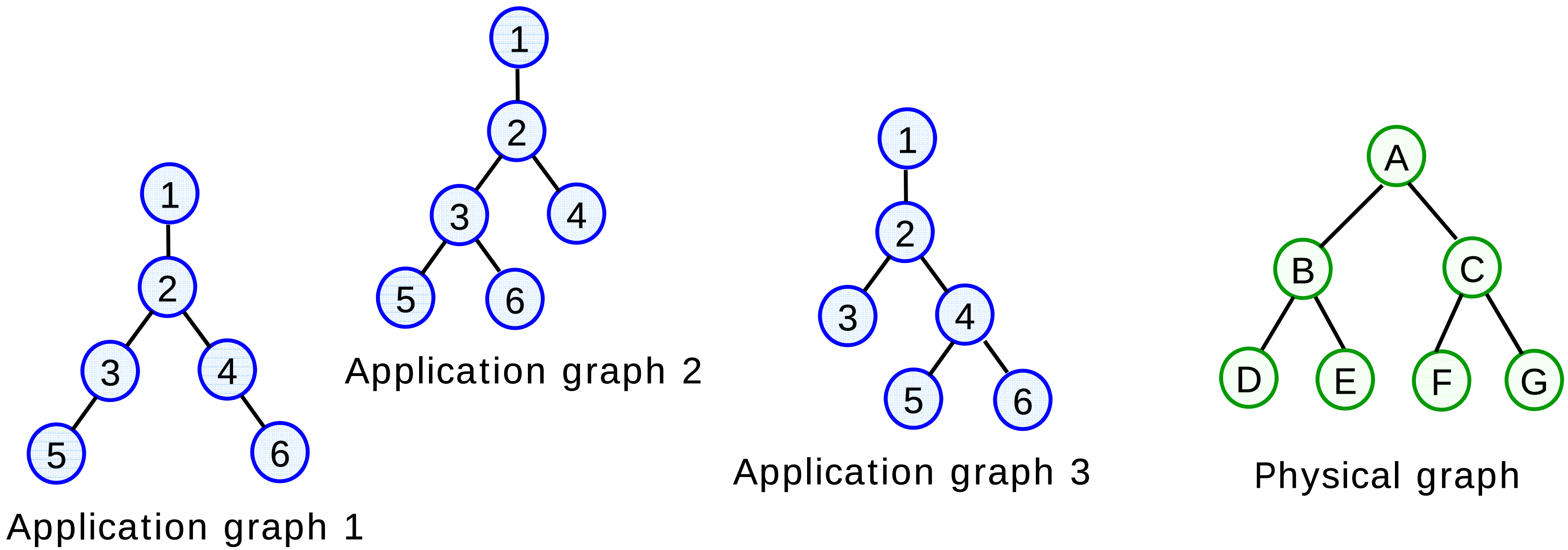
$$
\begin{aligned}
\min \quad & \left\{ \max_{n \in \mathcal{N}} \alpha_n \left(\sum_{v \in V} \bar{d}_{v \to n,1} x_{v \to n} \right) \right. \\
& + \max_{l=(n_1,n_2) \in \mathcal{L}} \beta_l \left(\sum_{e \in E} \frac{f_{e \to (n_1,n_2)} + f_{e \to (n_2,n_1)}}{c_l} \right) \\
& \left. + \sum_{l=(n_1,n_2) \in \mathcal{L}} \beta'_l \sum_{e \in E} \frac{f_{e \to (n_1,n_2)} + f_{e \to (n_2,n_1)}}{c_l} \right\},
\end{aligned}
$$

Bottleneck server

Bottleneck link

Total link utilization

How to place **multiple incoming application graphs** onto a physical graph?



Goal: Develop exact and online approximation algorithms

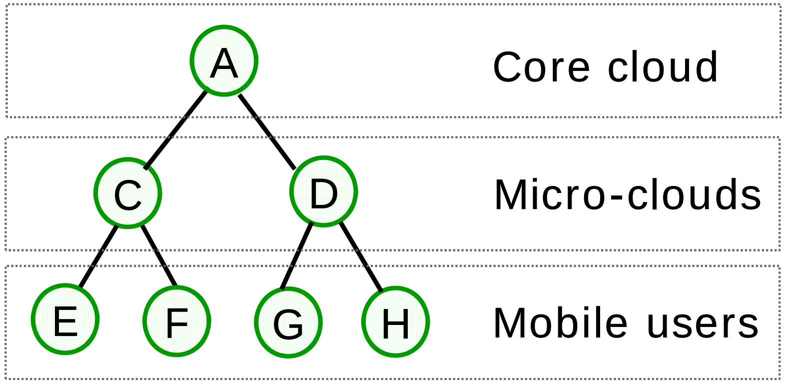
Approach to Online Service Placement

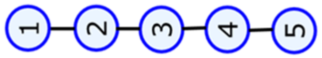
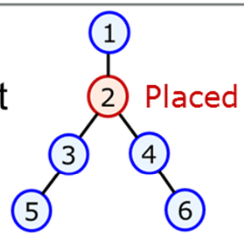
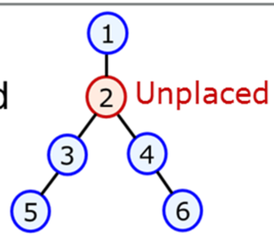
Imperial College London

- General Placement Problem is Really Tough
  - Focus on Tree-Structured Application and Physical Graphs
- Develop an Algorithm to Place Linear Application Graph
  - Obtain the optimal mapping (solution) for this special case as a "building blocking"
- Use the Above to Handle Tree Application and Physical Graphs
  - The path from the root to a leave node is a linear sub-graph
  - Allow pre-specified placement for some junction nodes
  - Develop algorithms with polynomial-logarithmic complexity for online placement

Online Placement of Application Graphs

- *Natural for micro-clouds:* Distributed cloud environment with **hierarchical** structure
- Appropriate to consider tree physical graphs
- We have obtained **exact and poly-log approximation algorithms** for offline/online service placement with load balancing as objective



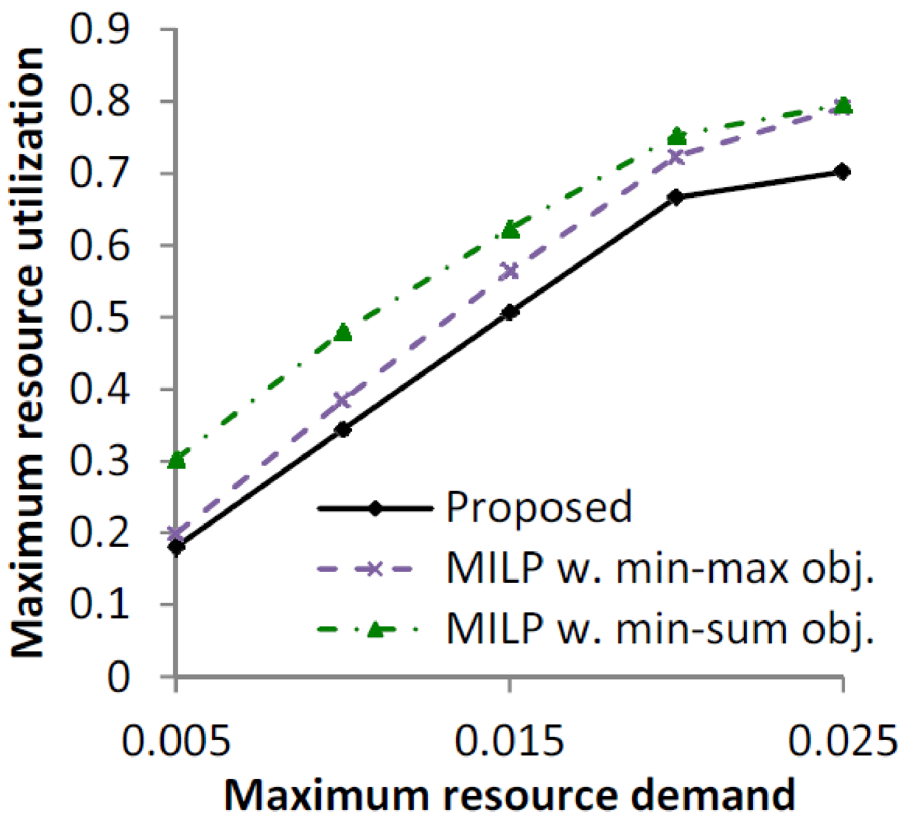
Single line application graph 	• **Exact algorithm** • Time complexity: $O(V^3N^2)$
Single/multiple application graphs with fixed placement of junction nodes 	• **Online approximation alg.** • $O(V^3N^2)$-time each graph • $O(\log N)$-competitive (w/o conflict constraints)
Single/multiple application graphs with some unplaced junction nodes 	• **Online approximation alg.** • $O(V^3N^{2+H})$-time each graph • $O(\log^{1+H} N)$-competitive (w/o conflict constraints)

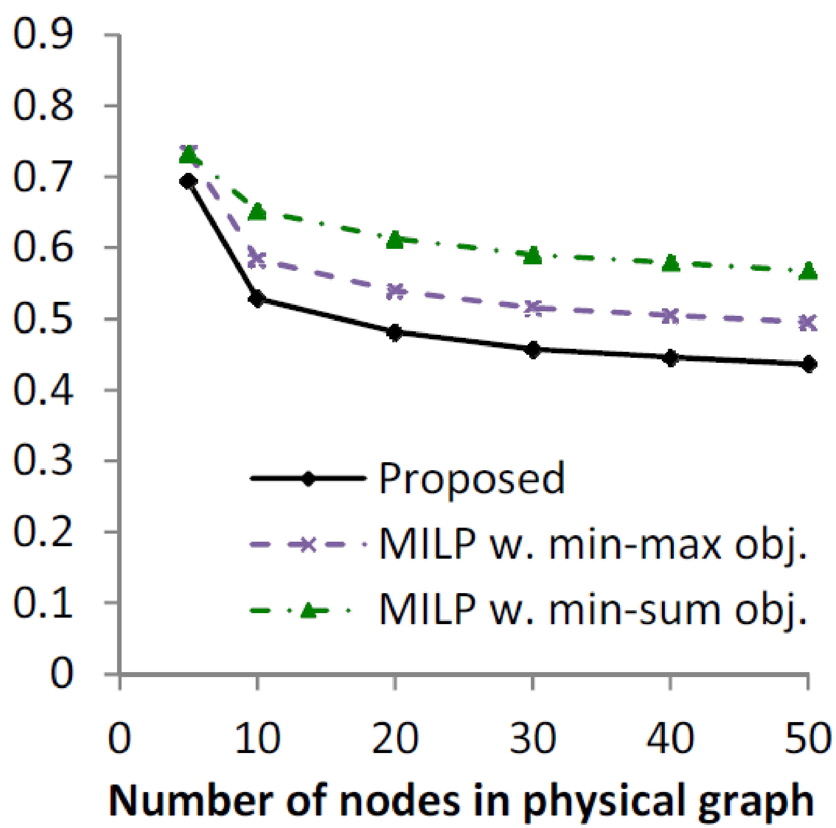
Approximation ratio = Worst case cost from algorithm / Optimal cost (OPT)
Competitive ratio = Worst case cost from online algorithm / Offline optimal cost (OPT)

H – maximum number of unplaced junction nodes on any path from the root to a leaf in the application graph

Imperial College London

Maximum resource utilization with unplaced junction nodes

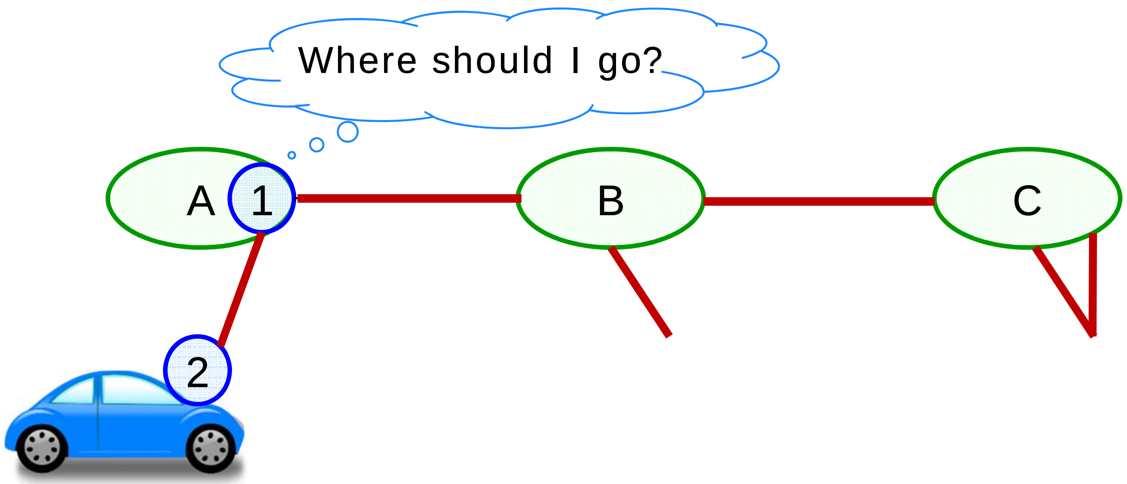




Dynamic Service Placement/Migration

Imperial College London

After initial placement, mobile users may move!

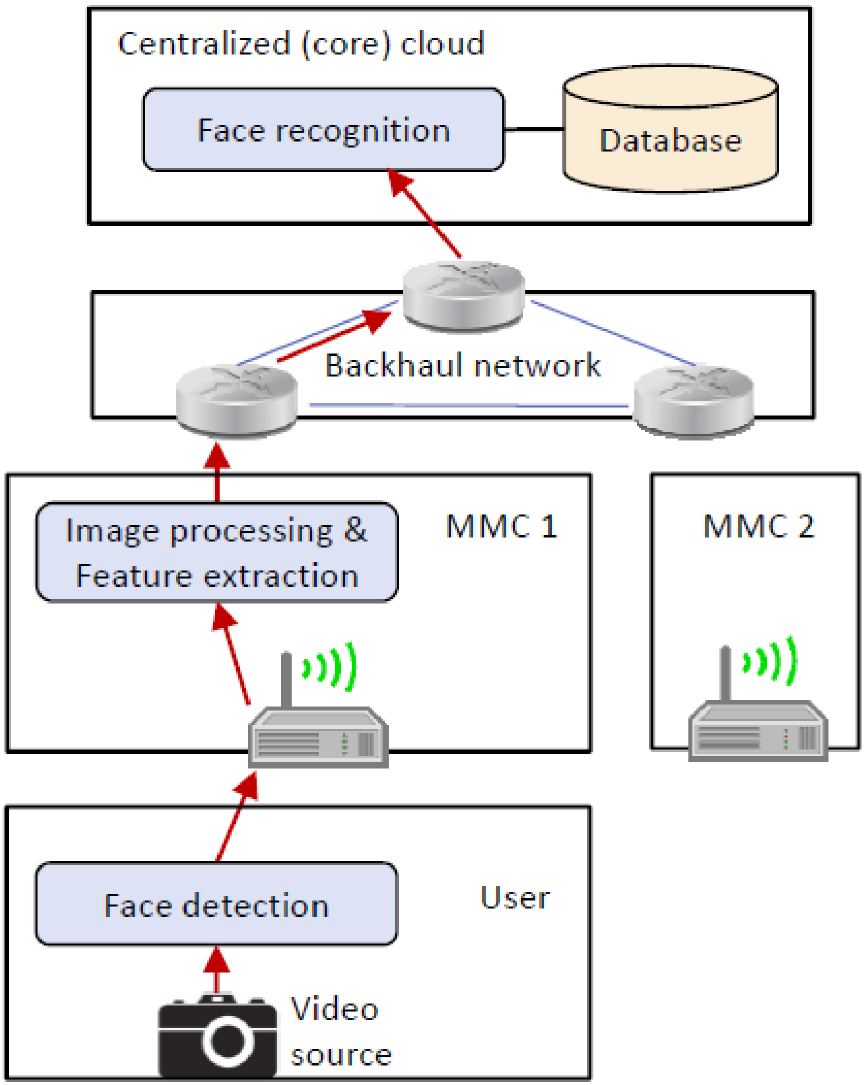


Tradeoff:

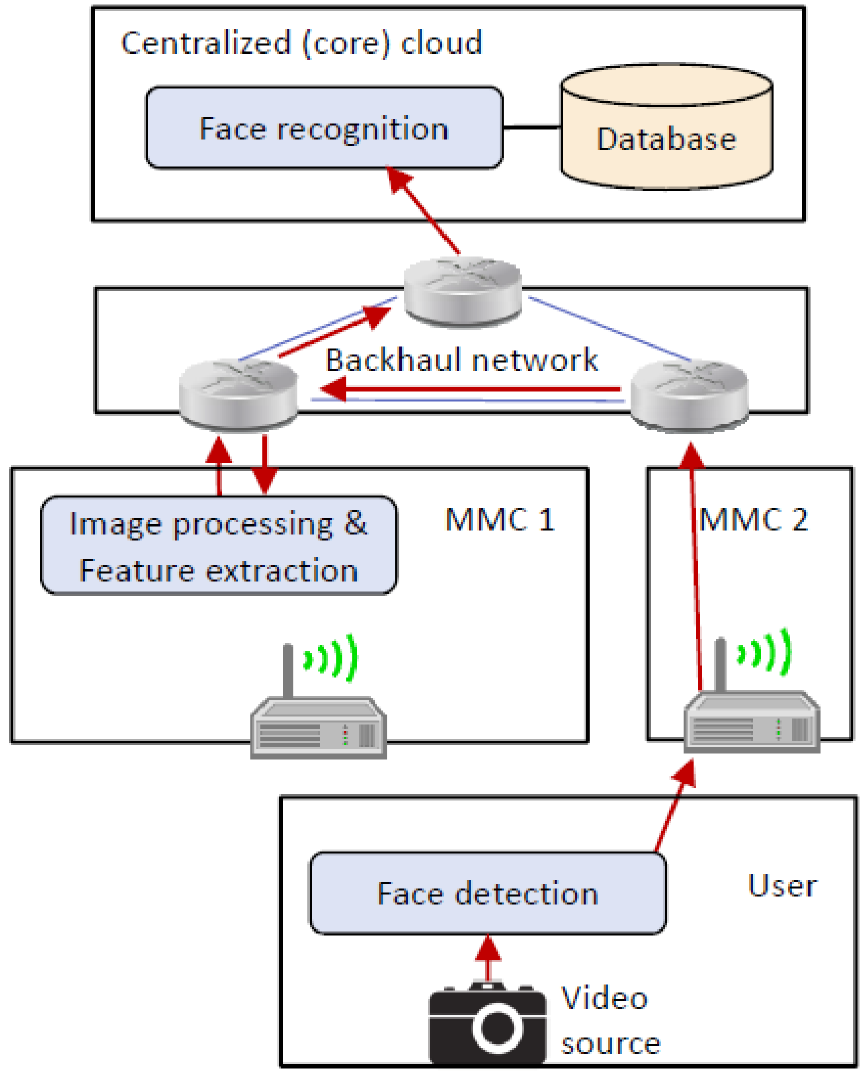
Migration cost
vs.
Performance gain after migration

Observation – Migration may be only beneficial in a long term. We need prediction and buffering mechanisms.

Face Recognition Example



(a)



(b)

Dynamic Service Placement/Migration

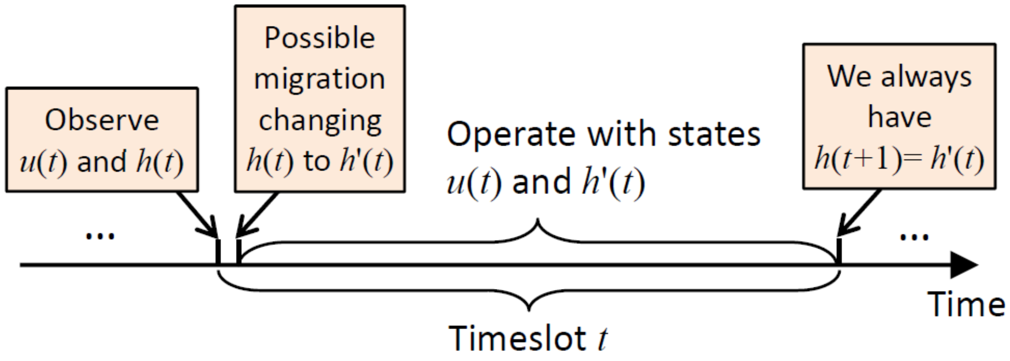
Imperial College London

We have considered three approaches suitable for different scenarios

- Approach 1: For homogeneous user mobility and cost functions
  - ➢ Markov decision process (MDP)
- Approach 2: For general but predictable mobility and costs
  - ➢ Online placement with arrivals/departures of service instances
- Approach 3: For scenarios allowing the buffering of user requests
  - ➢ Generalized Lyapunov optimization

MDP Approach to Service Migration

Imperial College London



$u(t)$: user location
$h(t)$: service location

Timeslots can have uniform or non-uniform length

- Objective (consider migration and data transmission costs):

discounted sum cost

migration cost + communication cost

$$\min \quad V_\pi(s_0) = \lim_{t \to \infty} \mathbb{E}\left\{ \sum_{\tau=0}^{t} \gamma^\tau C_{a_\pi}(s(\tau)) \middle| s(0) = s_0 \right\}$$

migration policy

discount factor ($0 < \gamma < 1$)

State ($u(t)$, $h(t)$): user & service locations

- Consider Markovian movement of users
- Markov decision process (MDP)
- The MDP can potentially have a very large state space

MDP Approach to Service Migration

Imperial College London

We ask ourselves…

- How to simplify the MDP (Markov Decision Process) to avoid state explosion?
- Can we approximate the original MDP with a simplified MDP? If yes, what is the approximation error?
- Can we find a closed-form solution to the discounted sum cost of an MDP?
- How to apply the theoretical model to practice?

Main contributions

- Provable structural property: Only migrate to a location closer to the user
- 1-D mobility with constant cost
  - » Threshold policy is provably optimal
  - » Modified policy iteration utilizing the existence of optimal threshold policy – more efficient than standard algorithms for solving MDPs
- 2-D mobility with constant-plus-exponential cost
  - » Approximate with 1-D MDPs with provable constant approximation error
  - » Closed-form solution to the discounted sum cost of the simplified MDP
  - » Verified by using real-world mobility statistics

Constant Cost Model for 1-D Mobility

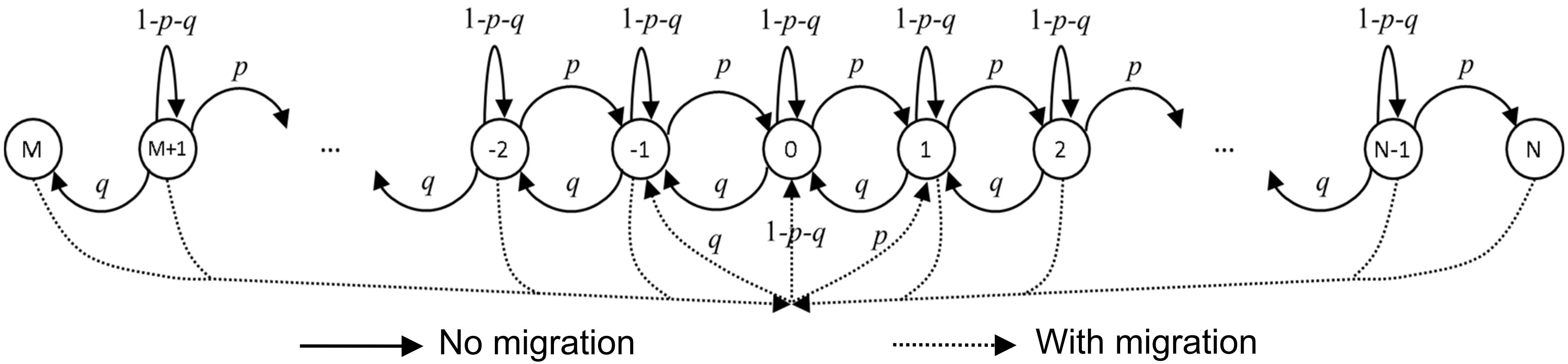
Imperial College London

States e represent the distance between user and service locations (in terms of base stations with micro-cloud server)

Cost definition:

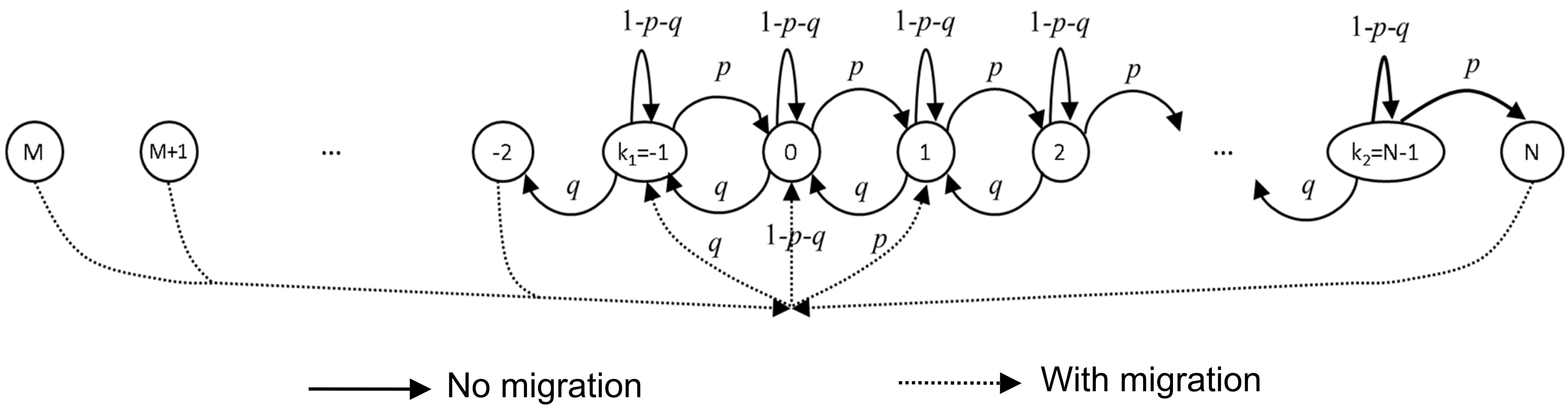
$$C_a(e) = \begin{cases} 0, & \text{if no migration or data transmission, i.e., } e = a(e) = 0 \\ \xi, & \text{if only data transmission, i.e., } e = a(e) \neq 0 \\ 1, & \text{if only migration, i.e., } e \neq a(e) = 0 \\ \xi + 1, & \text{if both migration and data transmission, i.e., } e \neq a(e) \neq 0 \end{cases}$$

Corollary: Migrating to locations other than the current location of the mobile user is not optimal.



Optimal Threshold Policy for Migration: Migrate after moving too far away

Proposition: There exists a threshold policy (k_1, k_2), where $M < k_1 \leq 0$ and $0 \leq k_2 < N$, such that when $k_1 \leq e \leq k_2$, the optimal action for state e is $a^*(e)$ = {not migrate}, and when $e < k_1$ or $e > k_2$, $a^*(e)$ = {migrate}.



Cost with Given Thresholds (k_1, k_2)

Imperial College London

Discounted sum cost:

$$\mathbf{v}_{(k_1,k_2)} = [V(k_1-1)\ V(k_1)\cdots V(0)\cdots V(k_2)\ V(k_2+1)]^{\mathrm{T}}$$

One-timeslot cost:

$$\mathbf{c}_{(k_1,k_2)} = \left[\begin{array}{ccccccccc} 1 & \underbrace{\xi \ \cdots \ \xi}_{-k_1 \text{ elements}} & 0 & \underbrace{\xi \ \cdots \ \xi}_{k_2 \text{ elements}} & 1 \end{array}\right]^{\mathrm{T}}$$

Modified transition matrix:

$$\mathbf{P}'_{(k_1,k_2)} = \begin{bmatrix} P_{0,k_1-1} & \cdots & P_{00} & \cdots & P_{0,k_2+1} \\ P_{k_1,k_1-1} & \cdots & P_{k_1,0} & \cdots & P_{k_1,k_2+1} \\ \vdots & & \vdots & & \vdots \\ P_{0,k_1-1} & \cdots & P_{00} & \cdots & P_{0,k_2+1} \\ \vdots & & \vdots & & \vdots \\ P_{k_2,k_1-1} & \cdots & P_{k_2,0} & \cdots & P_{k_2,k_2+1} \\ P_{0,k_1-1} & \cdots & P_{00} & \cdots & P_{0,k_2+1} \end{bmatrix}$$

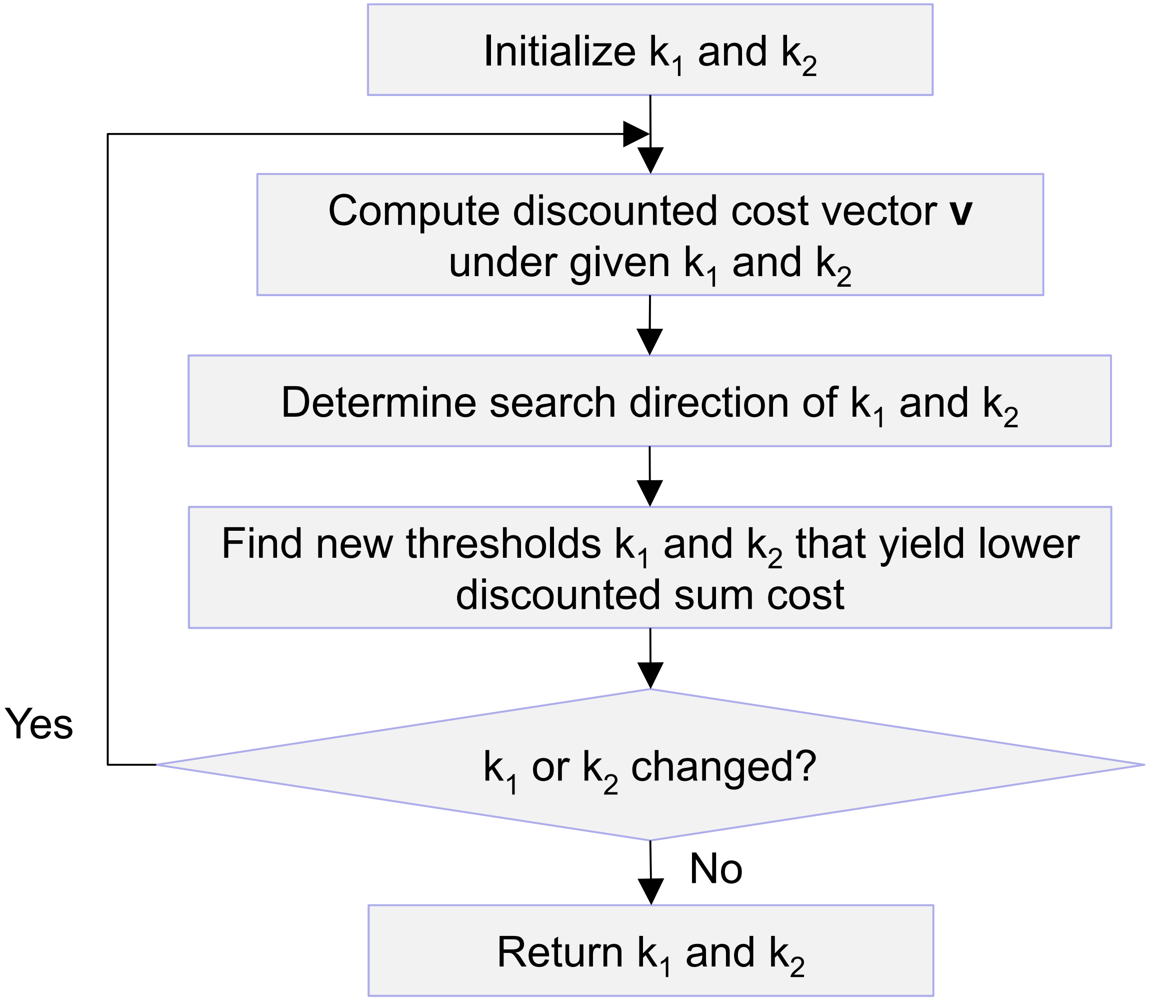
Balance equation:

$$\mathbf{v}_{(k_1,k_2)} = \mathbf{c}_{(k_1,k_2)} + \gamma \mathbf{P}'_{(k_1,k_2)} \mathbf{v}_{(k_1,k_2)}$$

Solving **v**:

$$\mathbf{v}_{(k_1,k_2)} = \left(\mathbf{I} - \gamma \mathbf{P}'_{(k_1,k_2)}\right)^{-1} \mathbf{c}_{(k_1,k_2)}$$

Modified Policy Iteration Algorithm



- The threshold-pair (k_1^*, k_2^*) is different in every iteration, otherwise the loop terminates.
- The number of iterations is O(|M|N).

Constant-Plus-Exponential Cost Model for 2-D Mobility

- Migration cost

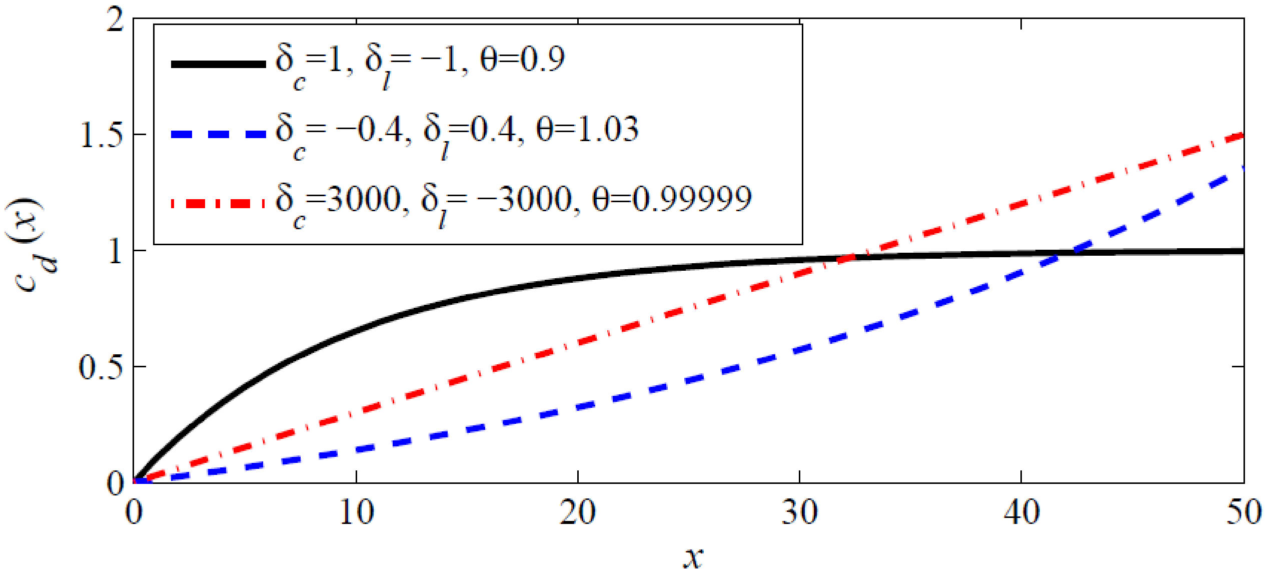
$$c_m(x) = \begin{cases} 0, & \text{if } x = 0 \\ \beta_c + \beta_l \mu^x, & \text{if } x > 0 \end{cases}$$

distance between new and old service locations

- Communication cost

$$c_d(x) = \begin{cases} 0, & \text{if } x = 0 \\ \delta_c + \delta_l \theta^x, & \text{if } x > 0 \end{cases}$$

distance between user and service locations



Simplified MDP Formulation for 2-D Mobility

Imperial College London

Use the distance between the user and service as states

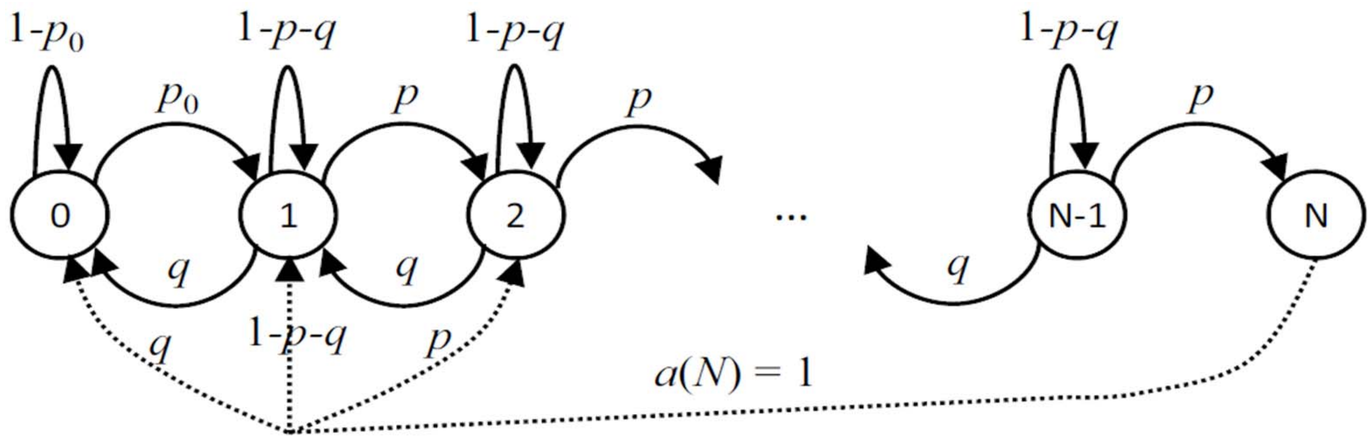
- An example in 2-dimensional space

User location: (a, b)
Service location: (x, y)

Each state is a **4-dimensional vector**.
State space can be arbitrary large.

Logical distance: # of hops between (a, b) and (x, y)

Each state is a **scalar value**.
State space is normally bounded.
(Migration must happen when beyond a certain distance.)



- Exactly optimal for uniform or single-directional 1-D user mobility

What Does the Simplified MDP Bring Us

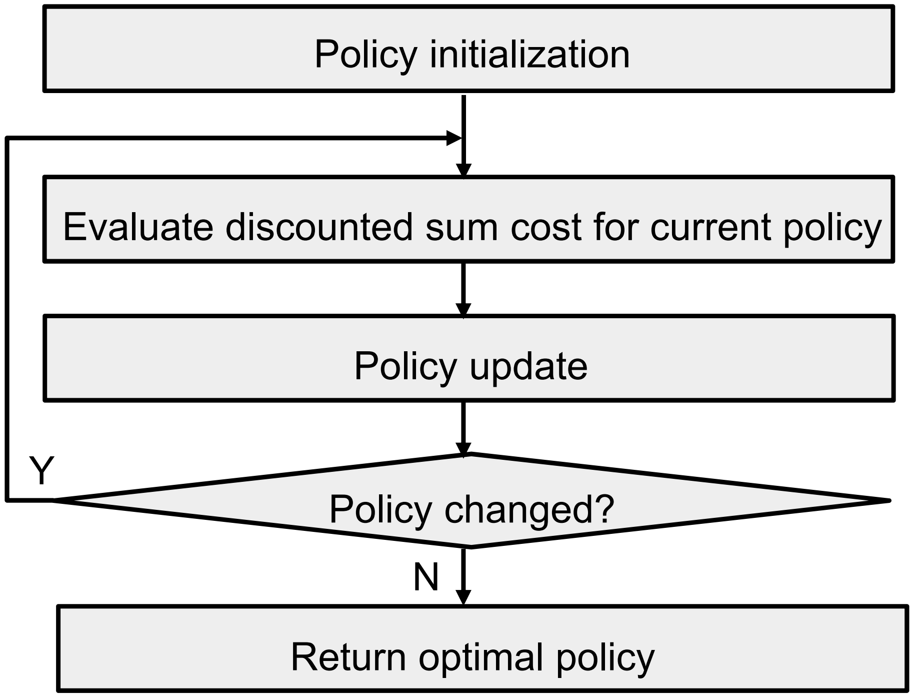
Closed-form solution to the discounted sum cost for a given service migration policy

- By solving difference equations
- Results simplify the policy search procedure
- Theoretical importance

$$V(d) = \delta_c + \delta_l \theta^d + \gamma \sum_{d_1=d-1}^{d+1} P_{dd_1} V(d_1)$$

$$V(d) = A_k m_1^d + B_k m_2^d + D + \begin{cases} H \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2 \theta \neq 0 \\ Hd \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2 \theta = 0 \end{cases}$$

Modified policy iteration mechanism with $O(N^2)$ complexity for each iteration
(Standard policy iteration has $O(N^3)$ complexity due to matrix inversion)



Using the Distance-Based MDP for 2-D Mobility

Imperial College London

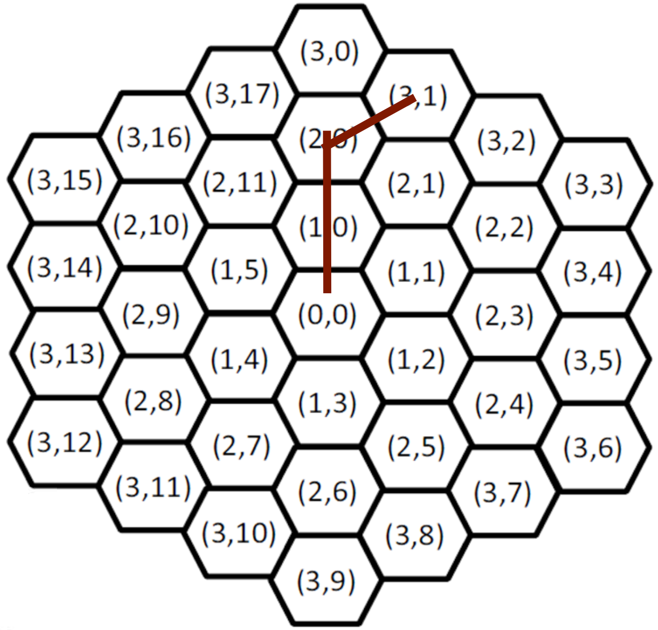
Consider uniform random walk mobility

- Large-scale average, each user is a sample path
- User moves to one of its neighboring cells with probability r

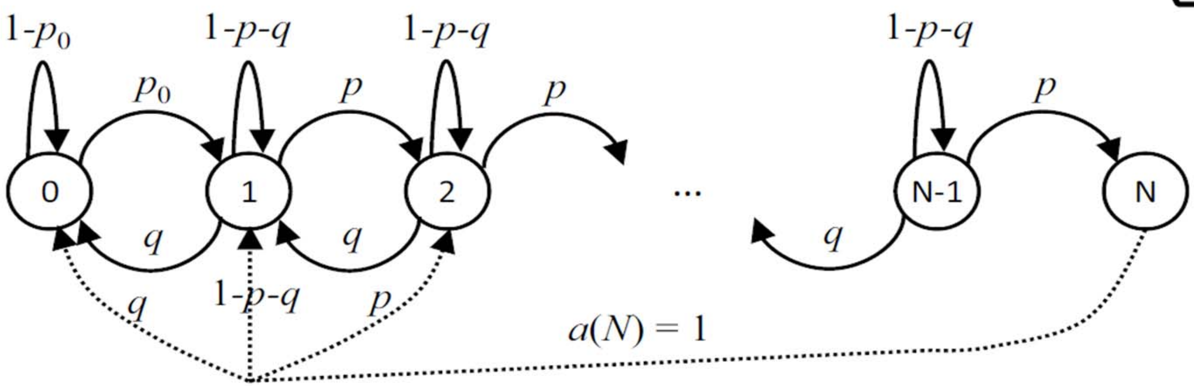
2-D difference model for hexagon cell structure

- For distance-based model with N states, the 2-D model has $M=3N^2+3N$ states

Standard policy iteration: **$O(N^6)$**
Proposed approach: **$O(N^2)$**



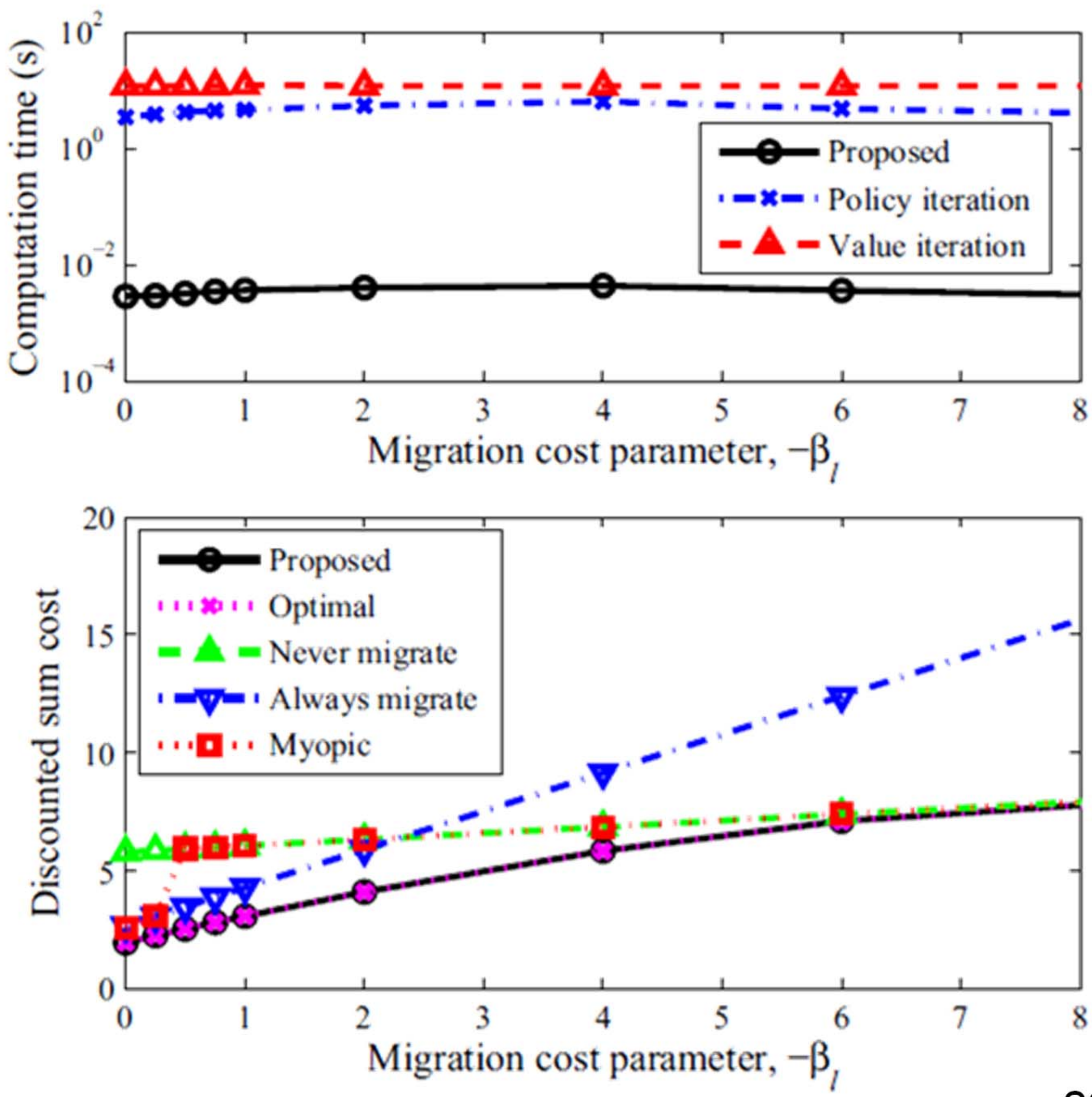
Find the policy from the distance-based MDP, with parameters $p_0=6r$, $p=2.5r$, $q=1.5r$



Always migrate on shortest path

Numerical Comparison: Exact vs. Approx.

- 2-D mobility
- Solving the original 2-D model consumes about 1,000 times more computation time
- Approximation result is very close to true optimum



Apply Real-World Mobility Statistics

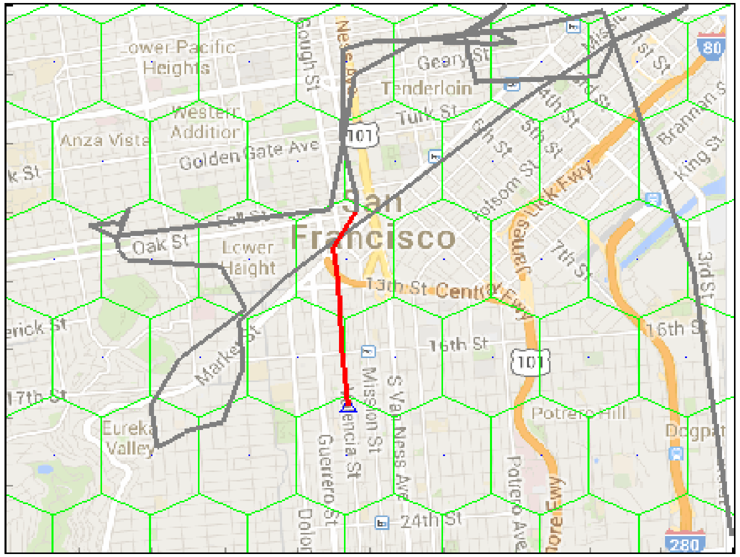
Estimate model parameters from the cell association history

- Define a time-window to look back
- Update migration policy at a specific interval

Only a subset of base stations have capacity-limited MMCs connected to them

- Only place on base stations with MMCs
- “Relocate” services on capacity-exceeded MMCs

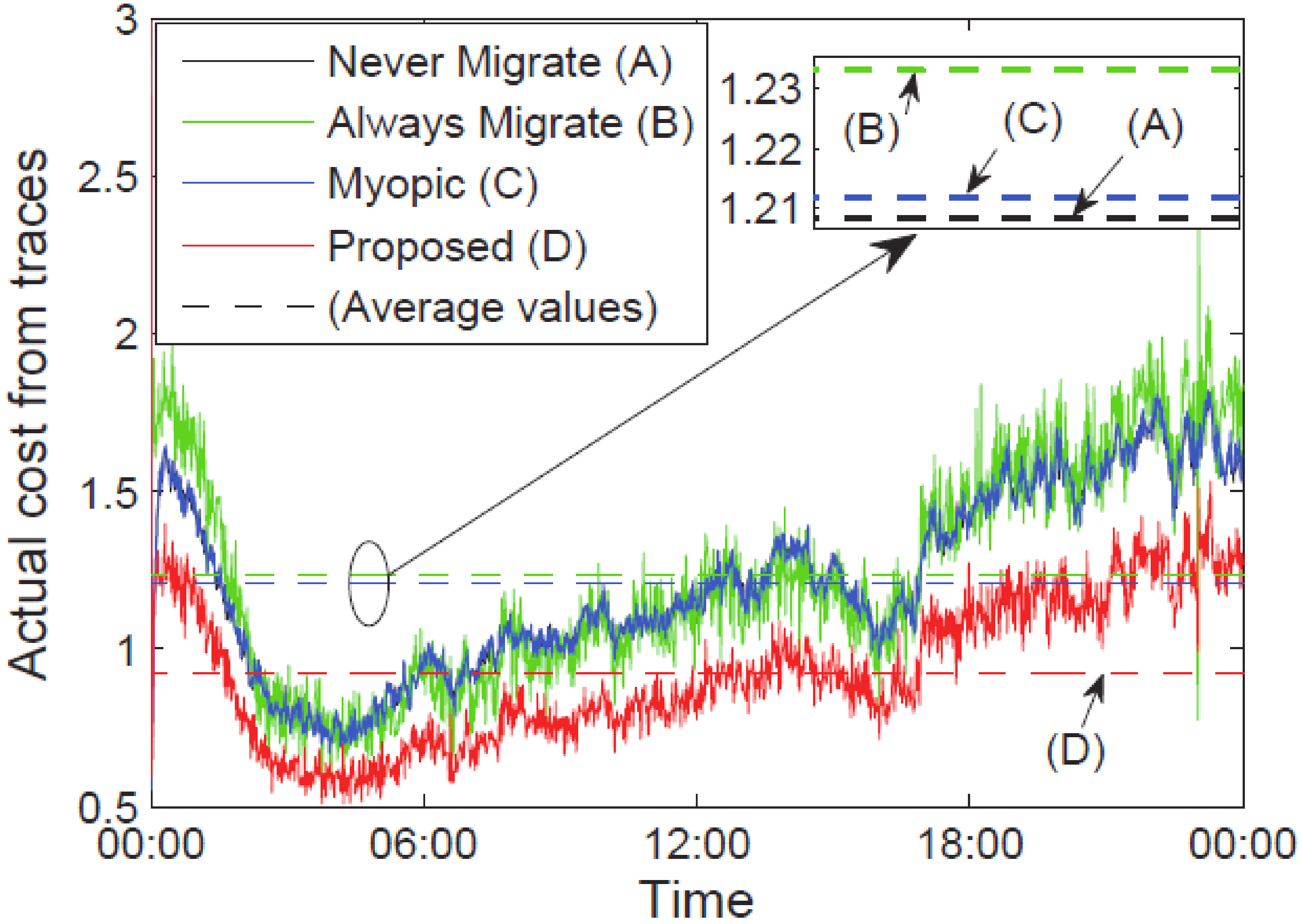
Simulation used mobility traces of San Francisco taxis [1], [2] with hexagonal cell structure



[1] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, “A parsimonious model of mobile partitioned networks with clustering,” in Proc. of COMSNETS, Jan. 2009.
[2] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, “CRAWDAD data set epfl/mobility (v. 2009-02-24),” Downloaded from http://crawdad.org/epfl/mobility/, Feb. 2009.

Simulation Results Using SF Taxi Trace

Imperial College London



Concluding Remarks

Imperial College London

Mobile Edge Clouds

- Important cloud architecture to shift computation to the network edge
- Efficient use of infrastructure to support mobility and network dynamics
- Potential support of time critical applications

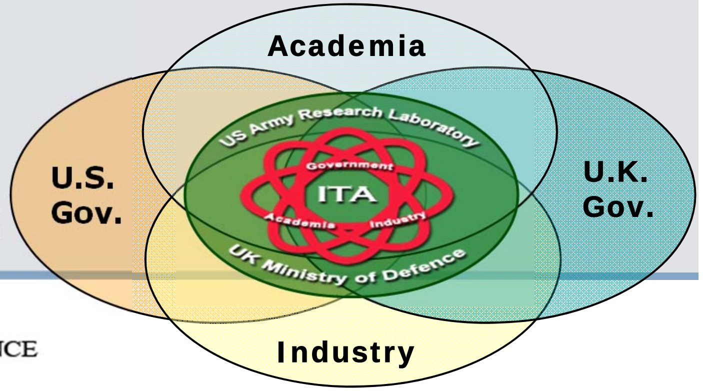
Main contributions

- Proposed service placement solutions with provable performance
- Developed service migration algorithms using MDP and complexity reduction (2D to 1D) techniques
- Verified the proposed methods using taxi mobility statistics in San Francisco

Research approach

- Outstanding problems are very hard to solve!
- Appropriate to identify unique characteristics of scenarios of interest (e.g., hierarchical structure for micro-mobile clouds)
- Develop exact solutions for simple cases (e.g., linear application graph) and extend and approximate complicated scenarios of interest

Acknowledgments



- Collaborators
  - Shiqiang Wang (IBM), Murtaza Zafer (Nyansa), Rahul Urgaonkar (Amazon), Ting He (Penn State Univ.), Kevin Chan (U.S. Army)
  - Research funding: U.S./U.K. ITA Project

- Publications

A. Machen, S. Wang, K.K. Leung, B.J. Ko and T. Salonidis, "Live Service Migration in Mobile Edge Clouds," to appear in *IEEE Communications Magazine* 2017.

S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer and K.K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1002-1016, Apr. 2017.

S. Wang, M. Zafer, and K.K. Leung, "Online Placement of Multi-Component Applications in Edge Computing Environments," *IEEE Access*, vol. 5, pp. 2514-2533, Feb. 2017.

R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan and K.K. Leung, "Dynamic Service Migration and Workload Scheduling in Edge-Clouds," *Performance Evaluation,* Vol. 91, pp. 205-228, Sept. 2015.

S. Wang, K. Chan, R. Urgaonkar, T. He, and K. K. Leung, "Emulation-based study of dynamic service placement in mobile micro-clouds,"IEEE MILCOM 2015.

S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," Proc. of IFIP Networking 2015.

S. Wang, R. Urgaonkar, K. Chan, T. He, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," IEEE ICC 2015.

S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," IEEE MILCOM 2014.

R. Urgaonkar, S.Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," IFIP Performance 2015, Oct. 2015.

Imperial College

Thank you