Optimizing the Crowdsourcing-based Bike Station Rebalancing Scheme

Yubin Duan and Jie Wu
Dept. of Computer and Info. Sciences, Temple University
Temple University, USA

1. Introduction

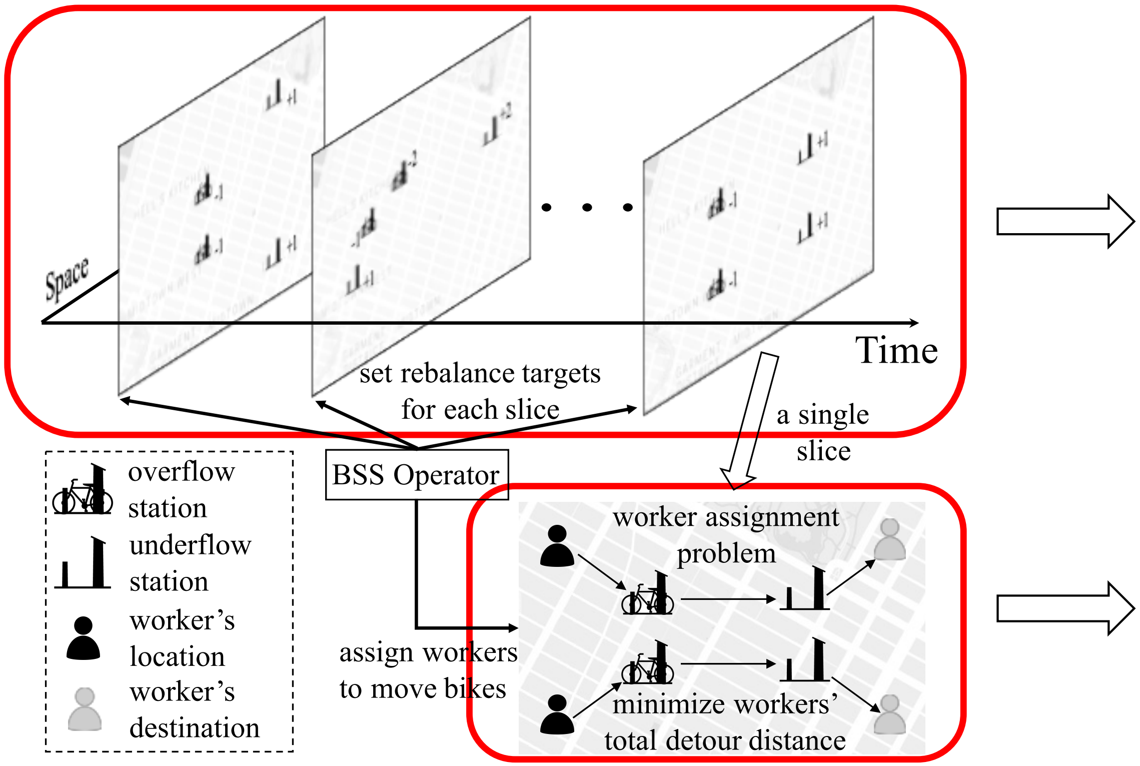
- Rebalancing bike sharing systems (BSSs)
  - Underflow station: lack of bikes, users cannot rent bikes
  - Overflow station: full of bikes, users cannot return bikes

- Existing rebalance scheme:
  - Truck-based approach[1]: hires trucks to transport bikes
  - User-based approach[2]: offers users monetary incentives

[1] Rebalancing bike sharing systems: A multi-source data smart optimization (KDD '16)
[2] Incentivizing users for balancing bike sharing systems (AAAI '15)

Motivation

- Crowdsourcing-based approach
  - BSS operator posts rebalancing targets
  - Recruits workers to move bikes
    - Workers have their own sources and destinations
    - Workers not only receive rewards but also save their travel time

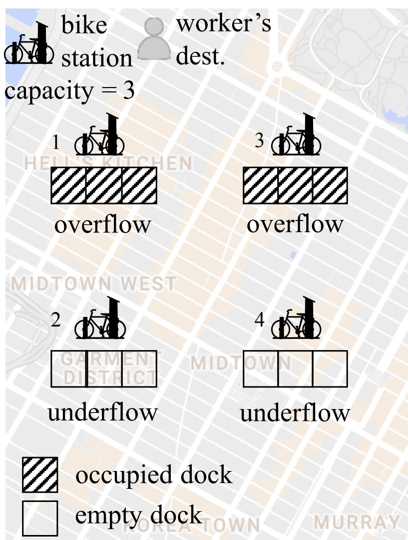


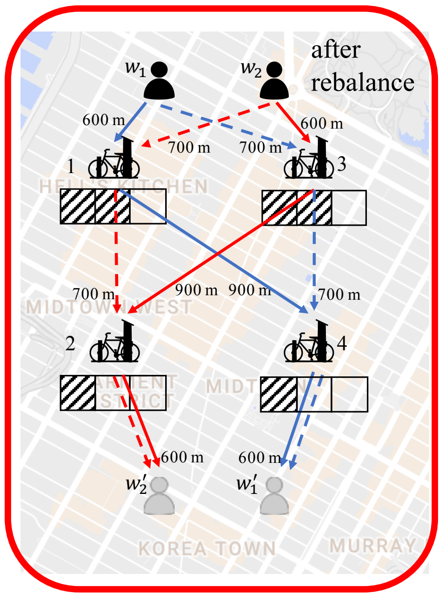
User dynamics cause bike imbalance

For a time slice, BSS operator recruits workers for rebalancing

Objective

- Try to minimize the overall worker detour
  - A complex optimization problem in spatial & temporal domains
    - Spatial domain: detour distances are related with worker assignment



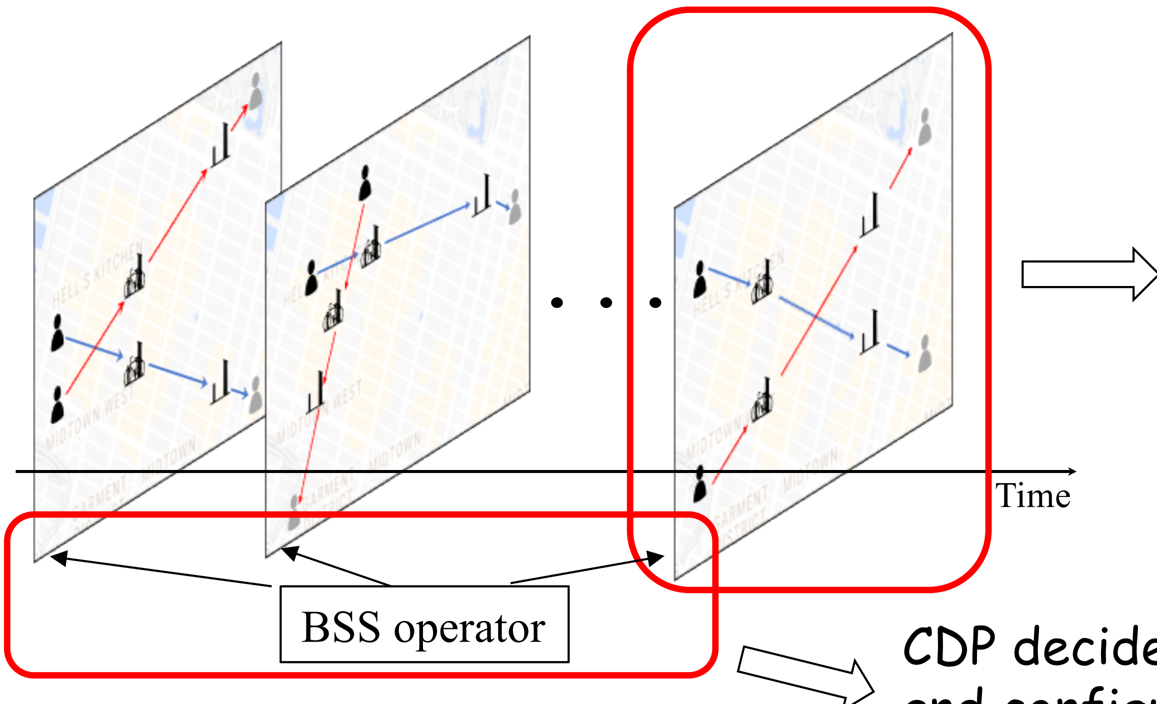


- solid lines: 4200m
- dashed lines: 4000m

    - Temporal domain: number of moved bikes is related with length of look-ahead time period

2. Problem Formulation

- Partition the complex problem
  - Spatial domain: worker assignment problem (WAP)
  - Temporal domain: configuration design problem (CDP)

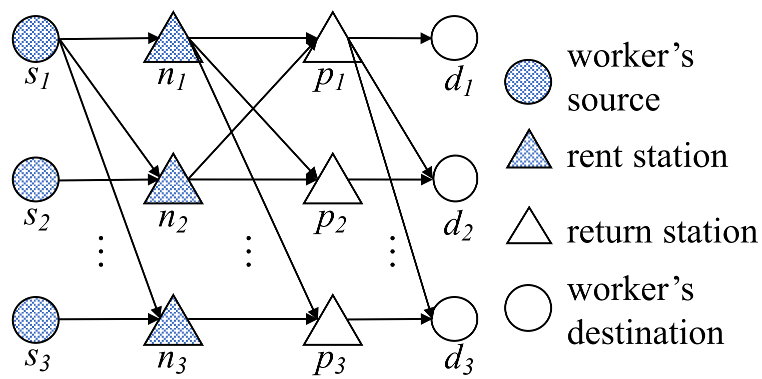


WAP decides worker movement in each slice

CDP decides rebalance frequency and configuration (i.e. number of bikes to move at each station)

Worker assignment problem (WAP)

- Modeled by a flow graph



- Formulation

$$\min \sum_{W,N,P} f(s_w,n)e(s_w,n)+f(n,p)e(n,p)+f(p,d_w)e(p,d_w)$$ Minimize moving distance

$$s.t. \sum_{N} f(s_w,n)=1, \ \sum_{P} f(p,d_w)=1, \forall w \in W \quad (1)$$ Assignment constraint

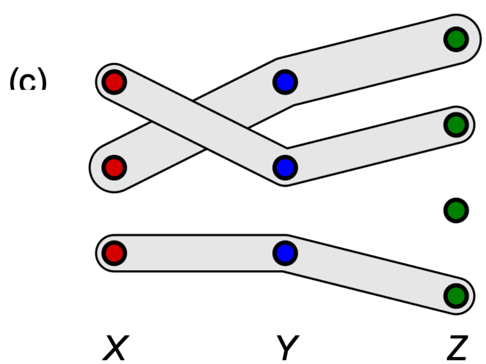
$$\sum_{W} f(s_w,n)=|\rho_n|, \sum_{W} f(p,d_w)=|\rho_p|, \forall n \in N, p \in P \quad (2)$$ Target constraint

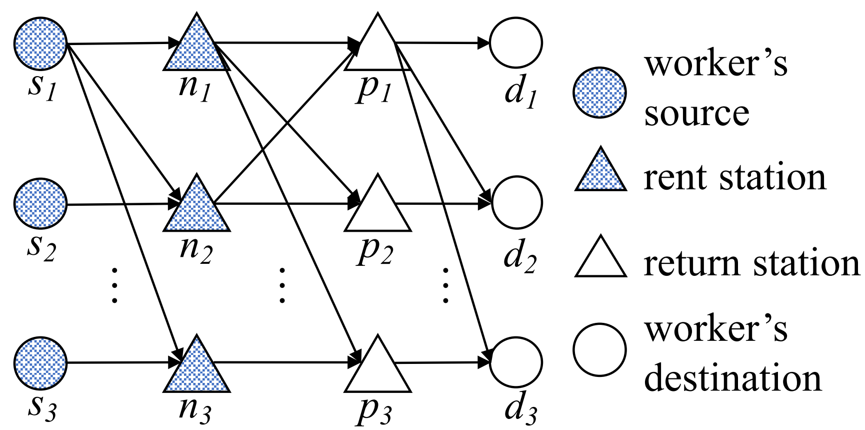
$$f(n,p)=\sum_{W}(f(s_w,n)\cdot f(p,d_w)), \forall n \in N, p \in P \quad (3)$$ Consistency constraint

$$f(s_w,n), f(p,d_w) \in \{0,1\}, \ f(n,p) \in \mathbb{N} \quad (4)$$ Flow-rate constraint

Hardness of the WAP

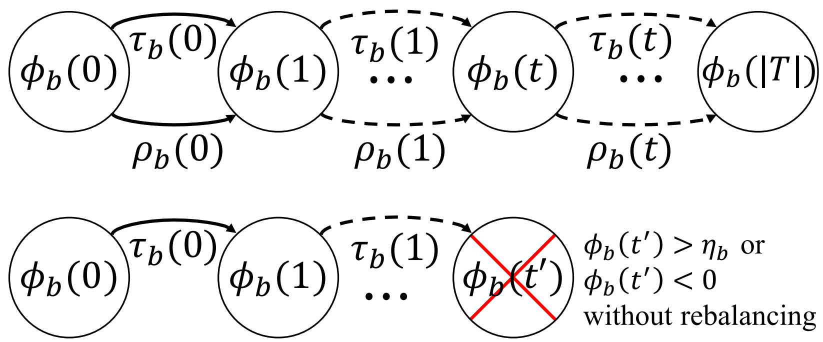
- NP-hard in general weighted graphs
- Reduced from the weighted 3D matching problem
  - Each assignment is equivalent to choosing a (worker, rent station, return station) combination





Configuration design problem (CDP)

- Modeled by discretized time series



- Formulation

$$\min \sum_{t \in T} \sum_{b \in B} |\rho_b(t)|$$ Minimize number of moved bikes

$$s.t. \quad 0 \leq \phi_b(t) \leq \eta_b,\ \forall b \in B,\ \forall t \in T$$ Capacity constraint

$$\sum_{b \in B} \rho_b(t) = 0,\ \forall t \in T$$ Matching constraint

$$\rho_b(t) \in \mathbb{N},\ \forall b \in B,\ \forall t \in T$$ Rebalancing-target constraint

3. Algorithm Design for WAP

- Two-Round Matching (TRM) algorithm
  - first round: matching underflow and overflow stations
  - second round: matching workers and paired stations

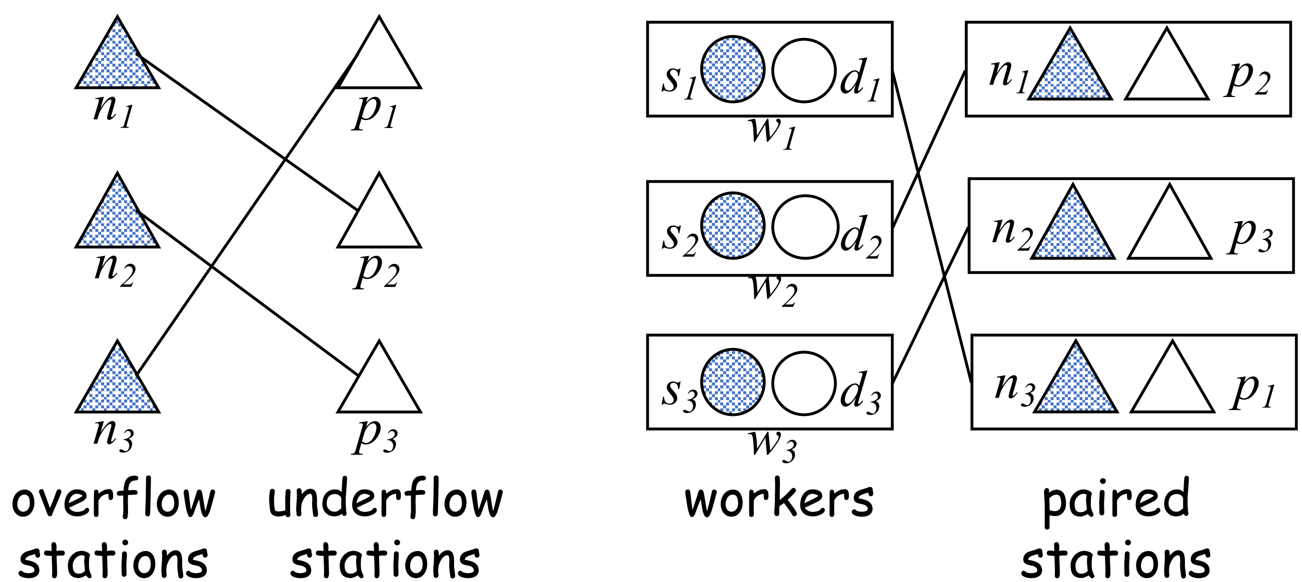
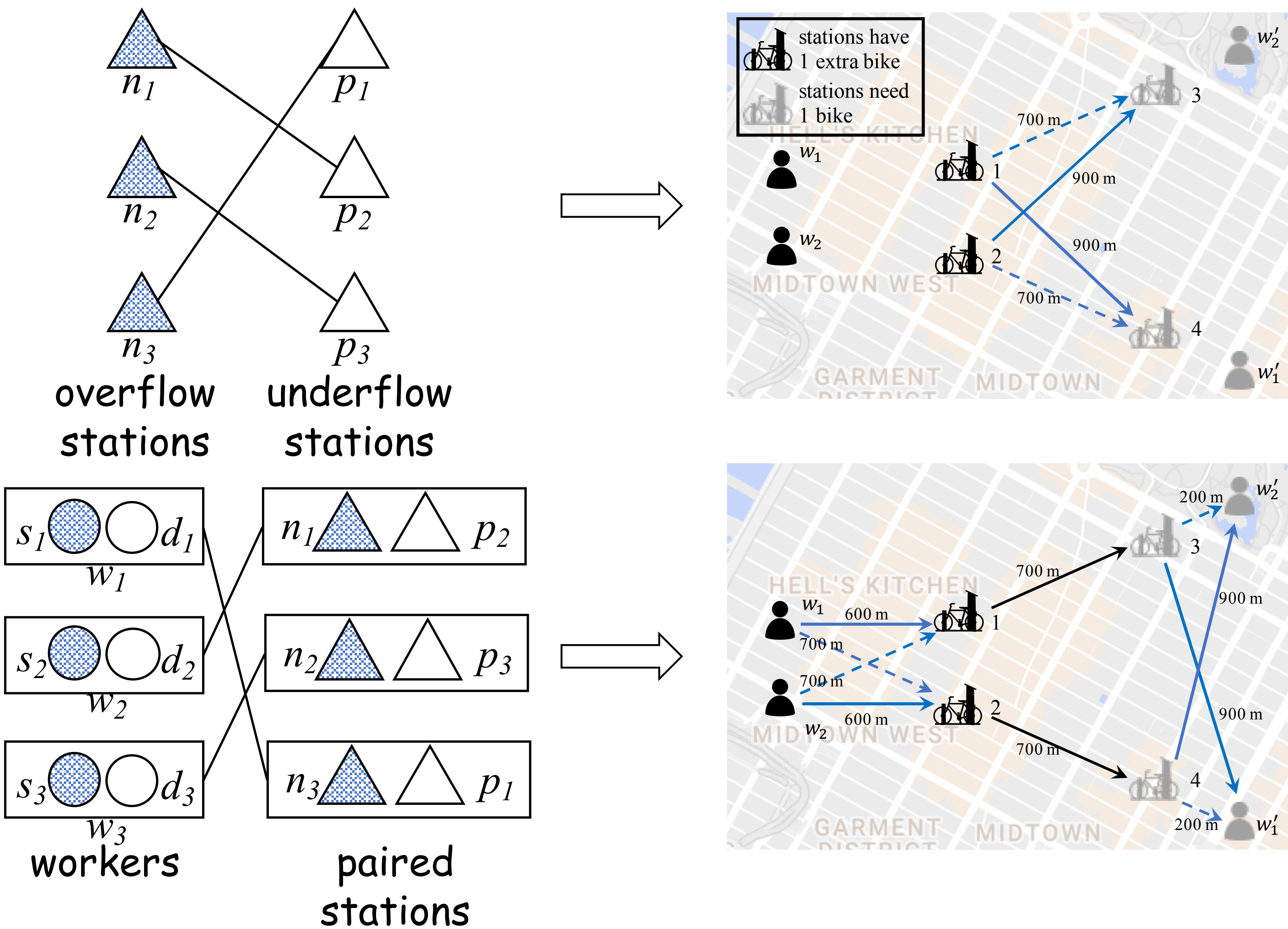
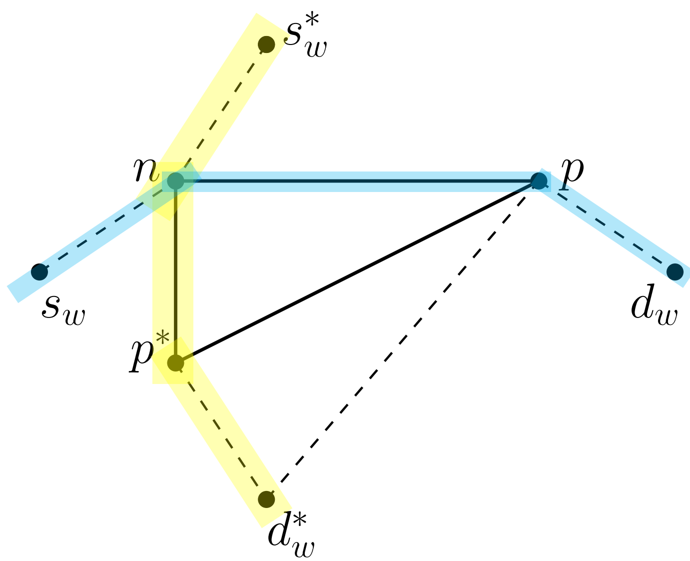


Illustration of the TRM



Performance analysis

- The TRM is a **3-approximation** algorithm
- Proof sketch



Optimality of the two rounds of matching:

$$\sum_{n\in N} dis(n,p) \le \sum_{n\in N} dis(n,p^*)$$

$$\sum_{n\in N} (dis(s_w,n)+dis(p,d_w)) \le \sum_{n\in N} (dis(s_w^*,n)+dis(p,d_w^*))$$

Triangle inequality:

$$dis(p,d_w^*) \le dis(p,p^*)+dis(p^*,d_w^*)$$

$$dis(p,p^*) \le dis(n,p)+dis(n,p^*)$$

Combining:

$$\sum_{n\in N}(dis(s_w,n)+dis(n,p)+dis(p,d_w))$$

$$\le \sum_{n\in N}((dis(s_w^*,n)+3dis(n,p^*) + dis(p^*,d_w^*))$$

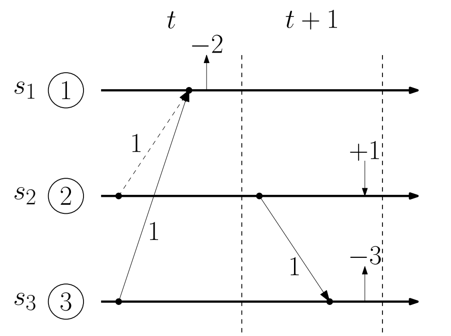
$$\le 3\sum_{n\in N}(dis(s_w^*,n)+dis(n,p^*)+dis(p^*,d_w^*)) = 3OPT.$$

4. Algorithm Design for CDP

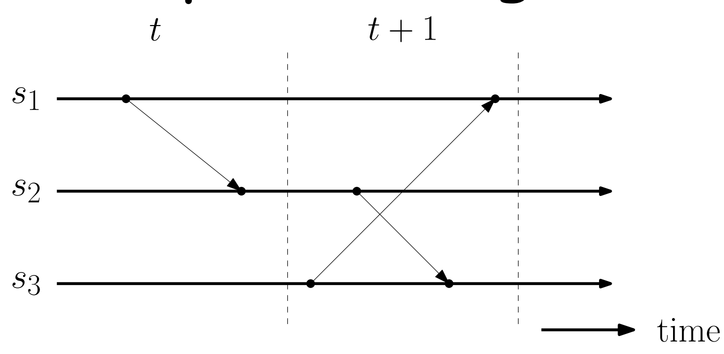
- k-slice Greedy Algorithm (kGA)
  - k is chosen by the BSS operator
  - Maybe infeasible when k is large
- Greedily Look Ahead (GLA)
  - Insight: let BSS live as long as possible
    - i.e. no overflow and underflow events
  - Procedures:
    - similar to kGA except k is greedily chosen by the algorithm
    - i.e. choose the largest k such that the problem is feasible

Performance analysis

- In general, larger k implies better performance
  - Ex: 2GA outperforming 1GA
  - Slanted arrow lines representing bike re-balancing activities between pairs of stations



- Exceptions: 1GA outperforming GLA

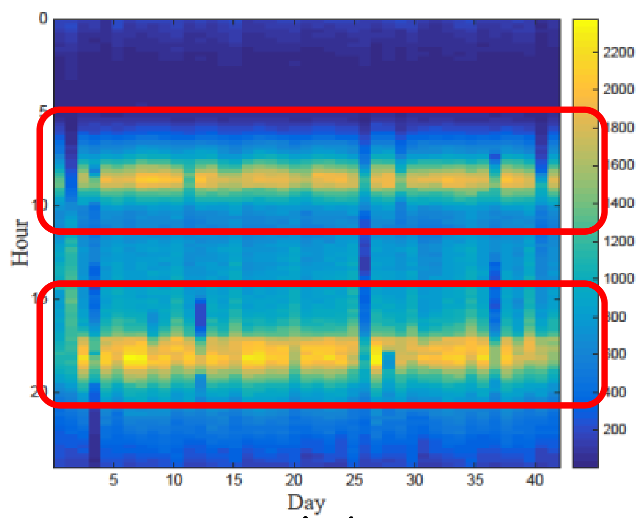


4. Experiment

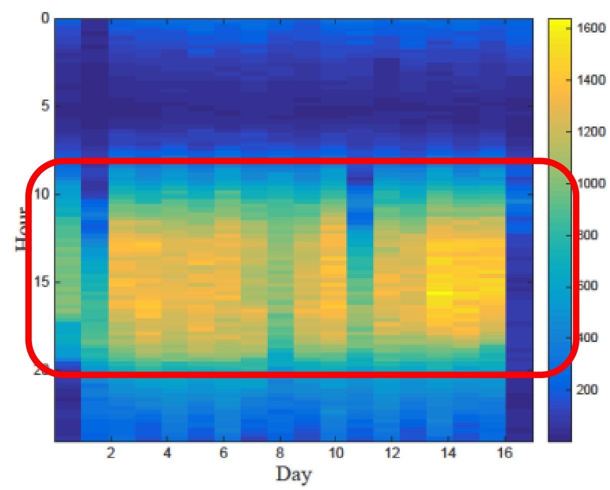
- NYC Citi Bike dataset

Data Source	New York City	
Time Span	8/1/17 to 9/30/17	
Weekdays (Weekends)	43 (17) days	
Bike Data	# Stations	328
	# Bikes	6,000
	# Trips	1.5+ million

- Usage patterns (temporal imbalance)

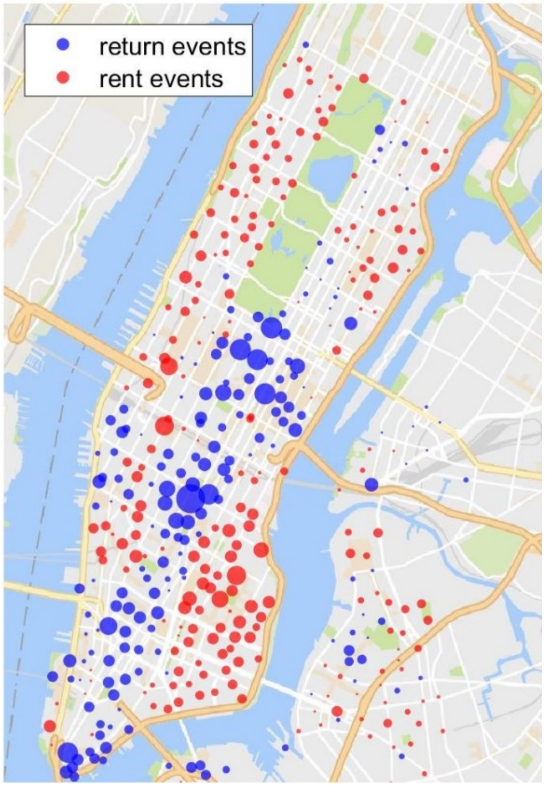


weekdays

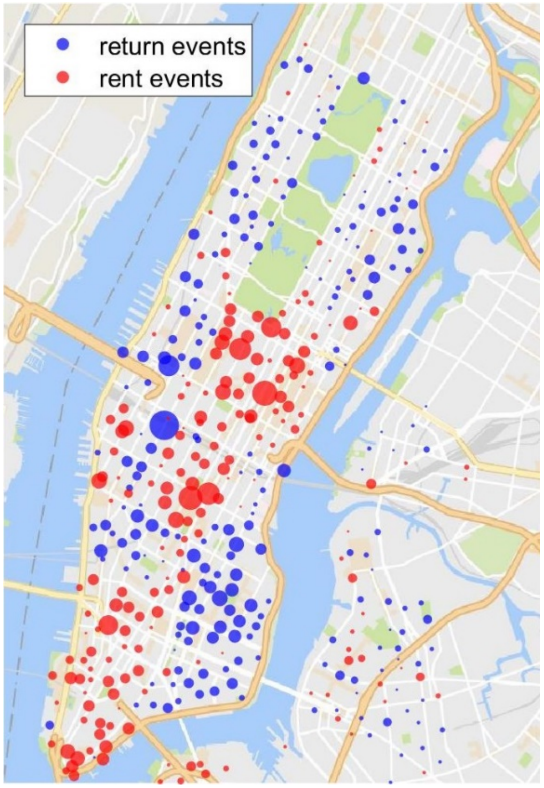


weekends

Usage patterns



AM rush hours (8:00 - 10:00 AM)



PM rush hours (5:00 - 7:00 PM)

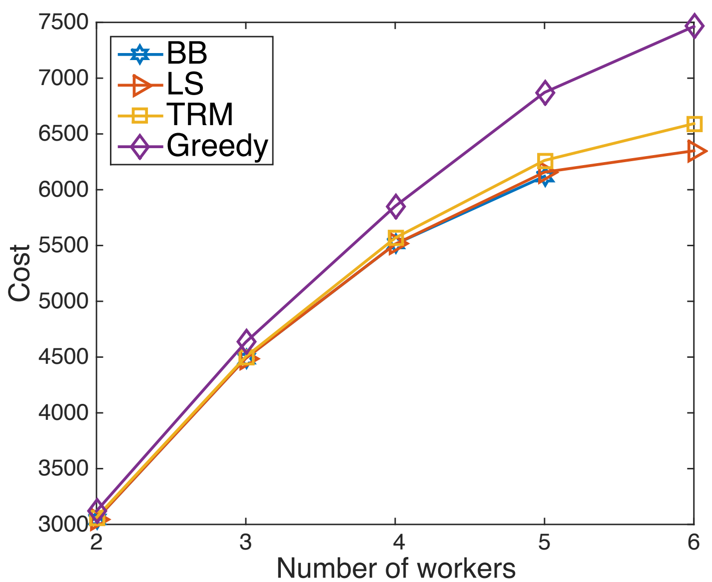
Experiment Setup

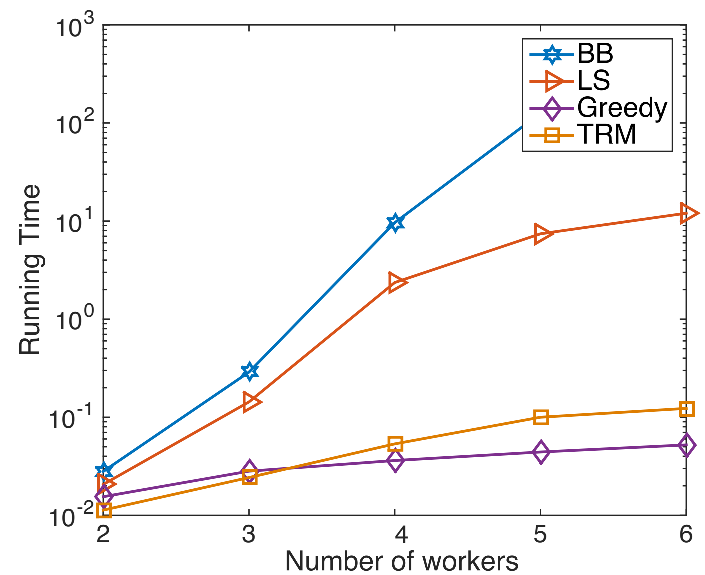
- Comparison algorithms
  - For WAP:
    - Branch-and-bound (BB)
    - Local Search (LS)
    - TRM
  - For CDP:
    - 1-GA
    - 2-GA
    - GLA
- Settings:
  - Station locations are extracted from the NYC dataset
  - User demands are generated by the prediction algorithm[3]
  - Time slice length is set to 20 min to make sure workers could finish rebalancing tasks

[3] Rebalancing bike sharing systems: A multi-source data smart optimization (KDD '16)

Performance comparison for WAP

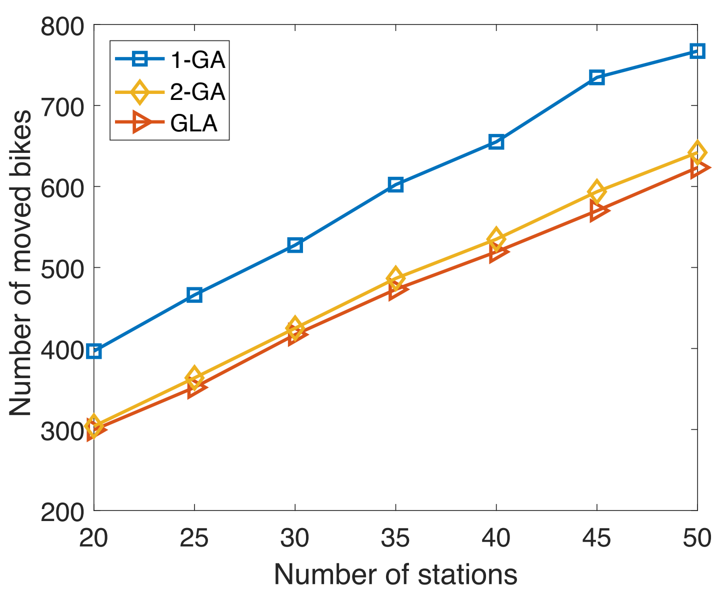
- BB and LS are extremely time consuming, cannot be applied in real-world applications
- TRM achieves near-optimal performance with theatrical bound in shorter time

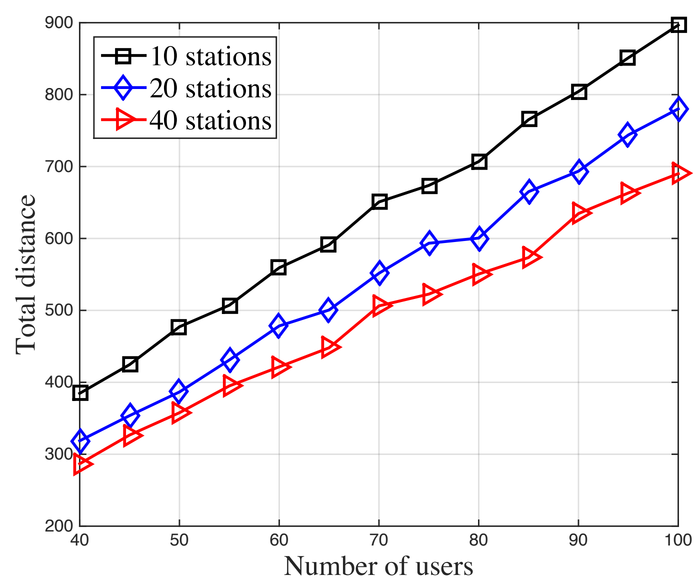




Performance comparison for CDP

- In terms of number of moved bikes, larger k usually implies better performance
- Overall performance
  - Run TRM under different station density to simulate sparse, regular and density station distribution
  - synthetic dataset that extracts real locations with different density





5. Summary

- **Crowdsourcing-based incentive** scheme
  - Recruiting workers to rebalance BSSs
- Partition the complex optimization problem
  - WAP in spatial domain and CDP in temporal domain
- Algorithmic solution
  - A **3-approximate** algorithm for WAP
  - A greedy algorithm for CDP
- Experiments on real-world dataset
  - Scalability and performance

Thank you

Q & A