Algorithms for Inventory Routing

Yang Jiao
Joint work with R. Ravi

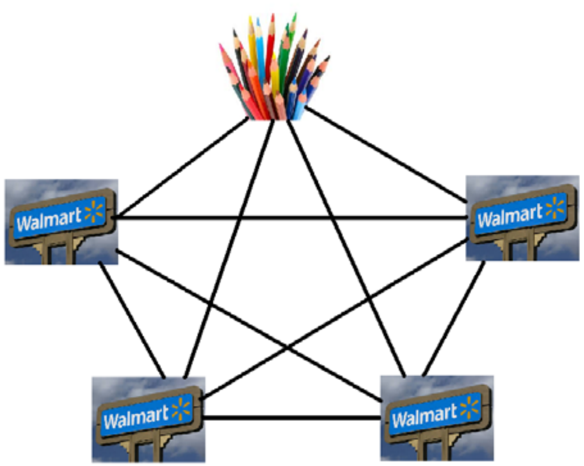
July 13, 2017

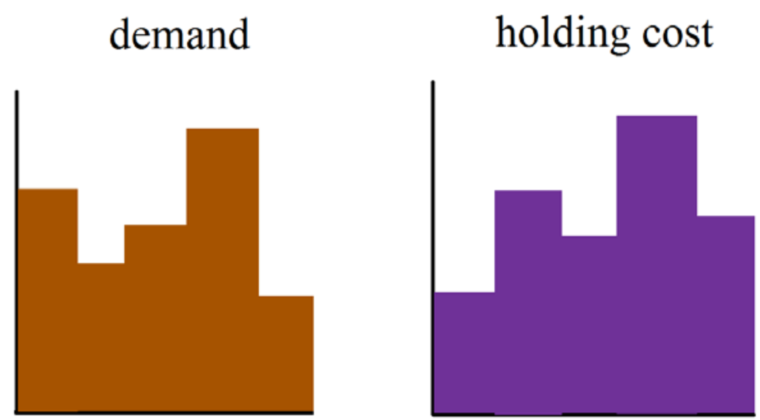
Outline

- Preliminaries
- Related Work
- Results
  - Provable bounds
  - Heuristics

Motivation

- Chain store tells a product supplier the demand patterns and storage costs per store location per day
- Supplier wants to minimize total delivery and storage costs





Problem Definition

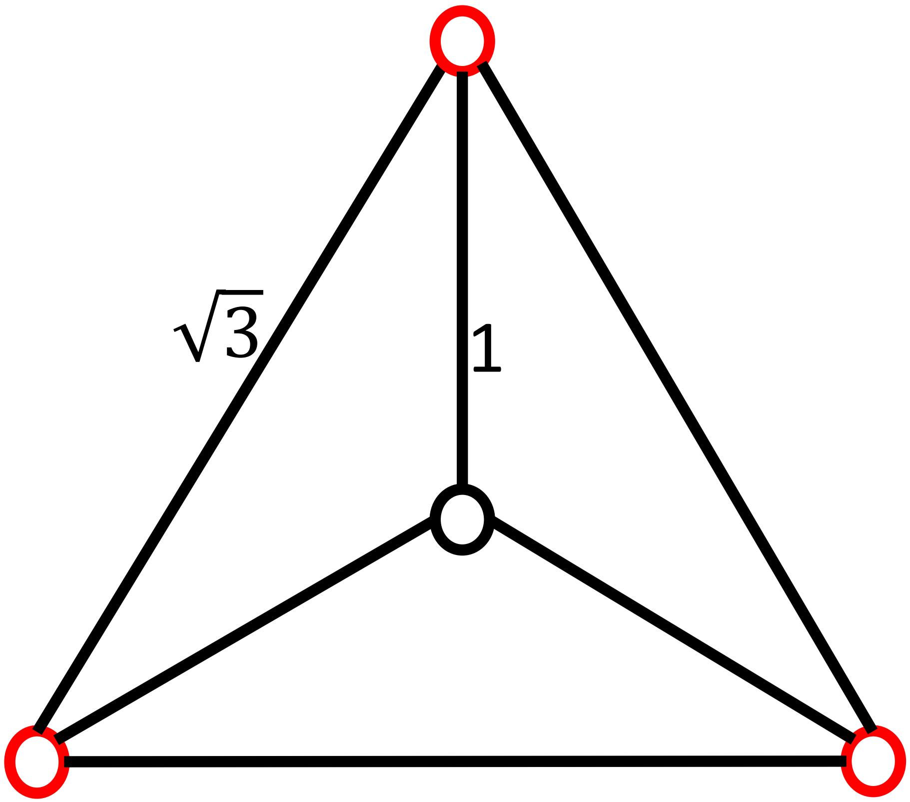
Input

- N stores on $G = (V, E)$ with metric distances w
- Depot at r, infinite capacity vehicle at r
- Discrete timeline $1, \ldots, T$
- Demand d_t^v of store v at time t, must be satisfied by time t
- Monotone holding costs $h_{s,t}^v$

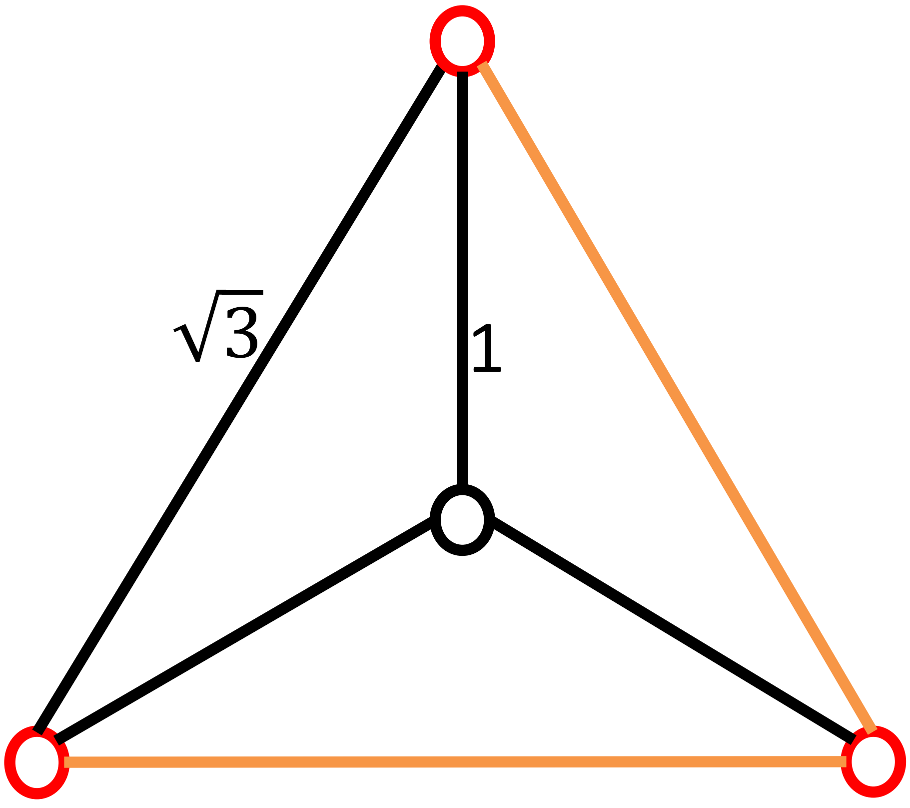
Objective

Tour $T_s \subset G$ per time $s \leq T$ minimizing total tour cost and holding cost

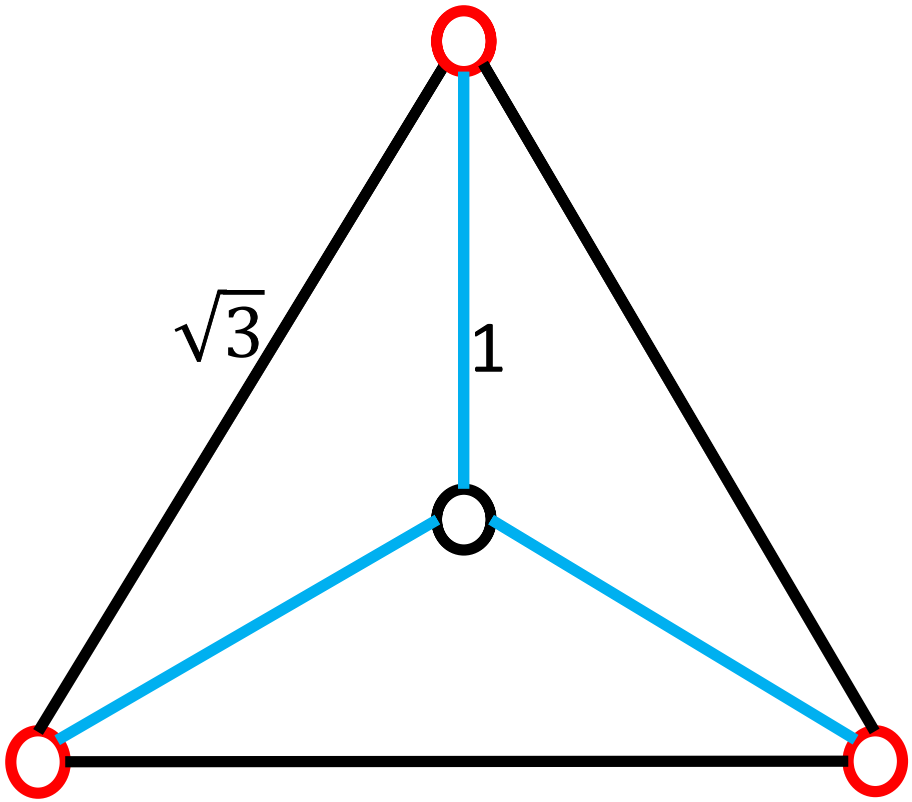
Steiner Tree



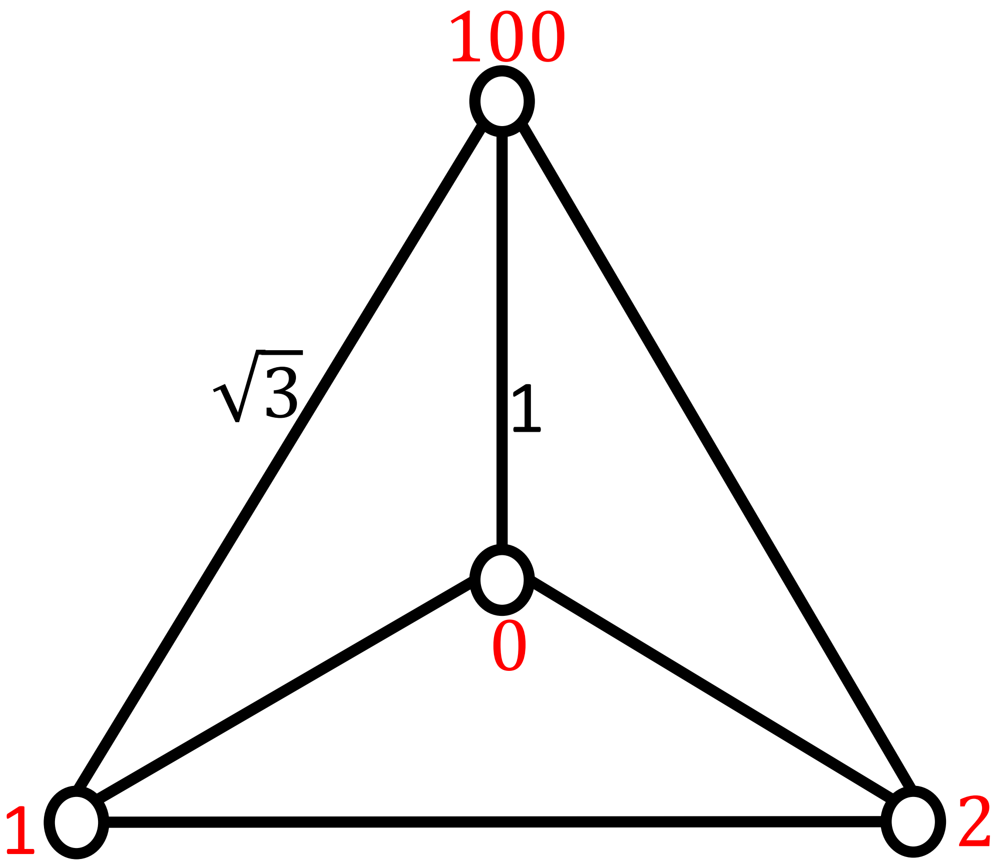
Steiner Tree



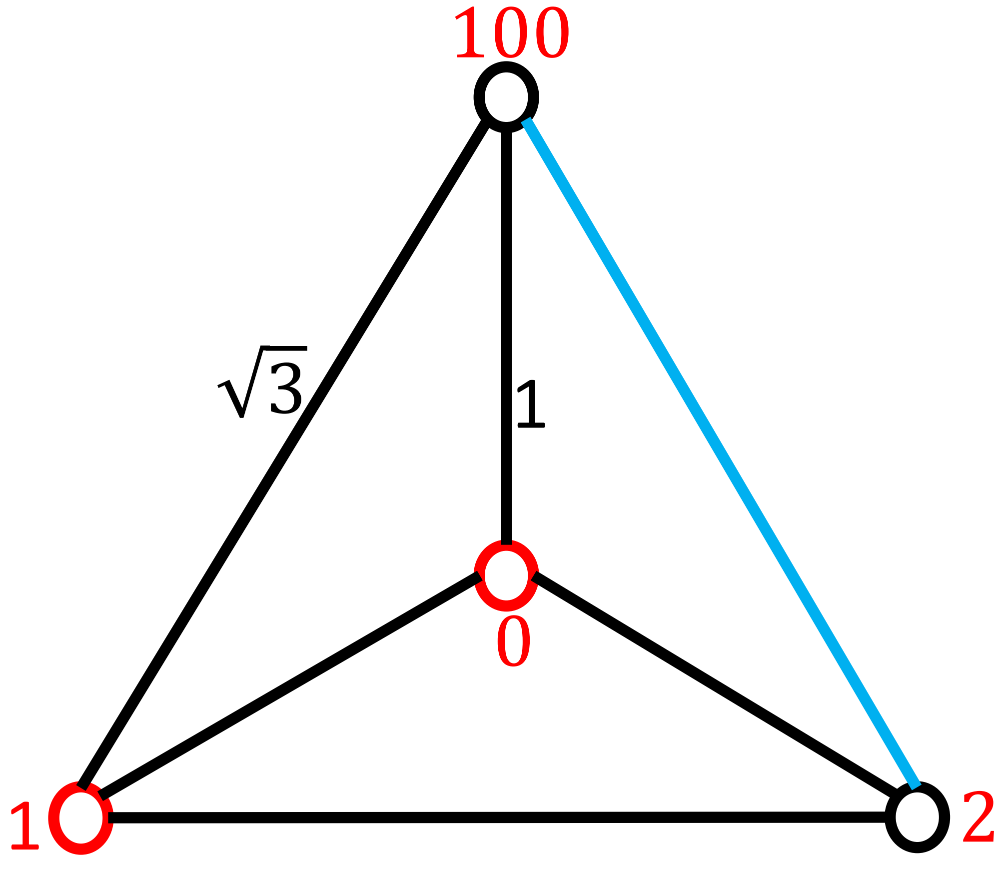
Steiner Tree



Prize Collecting Steiner Tree

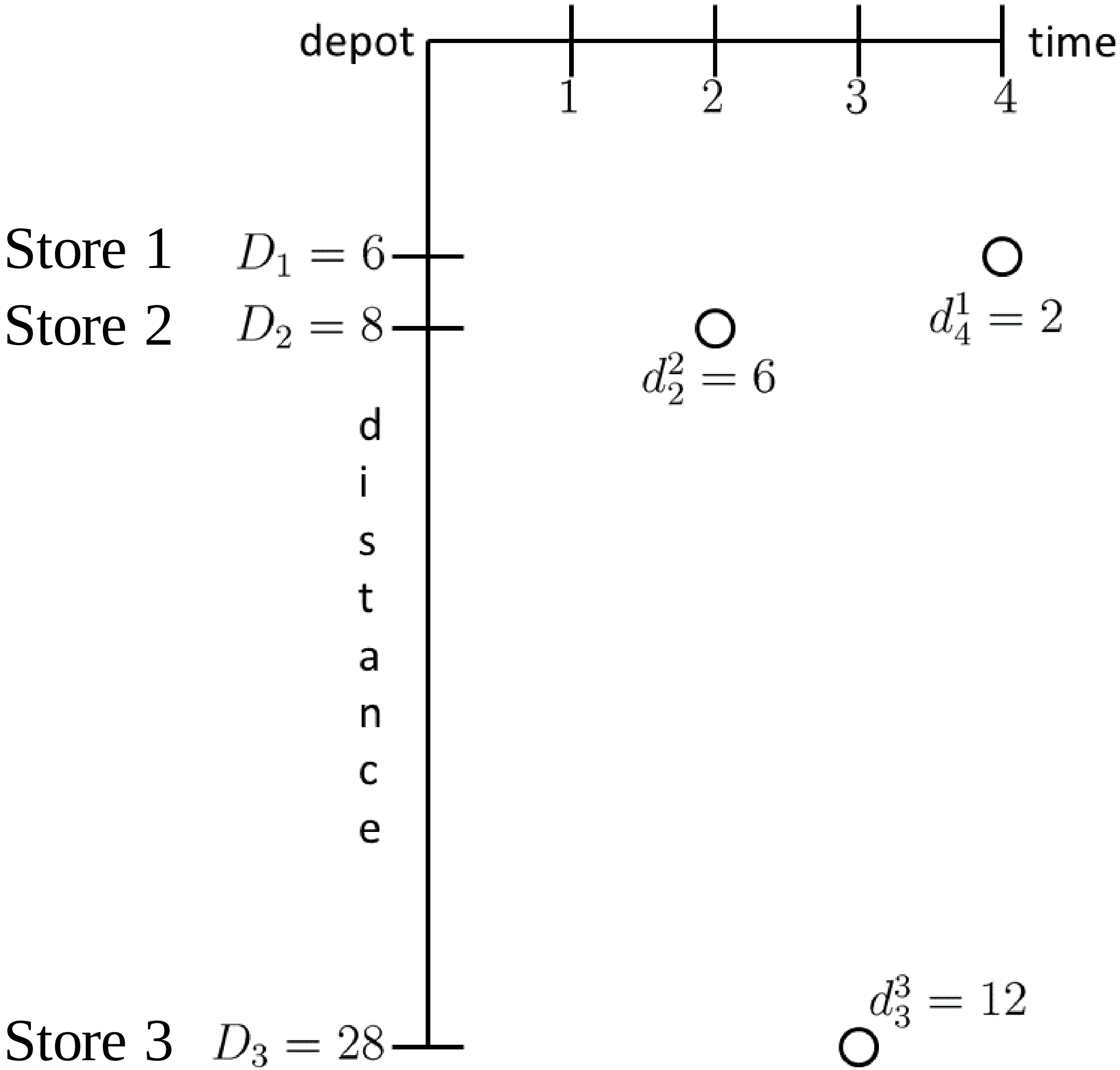


Prize Collecting Steiner Tree

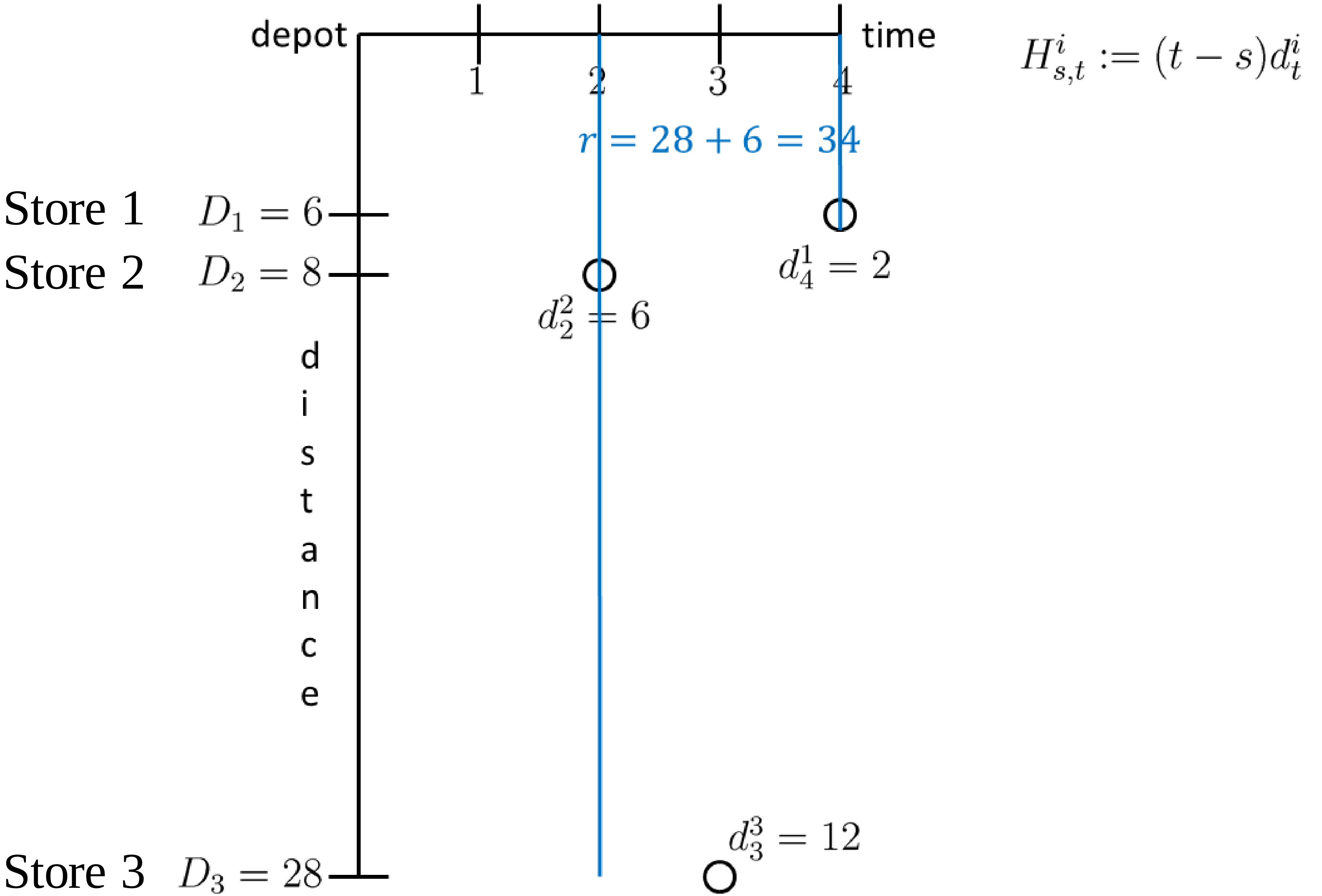


Example of IRP

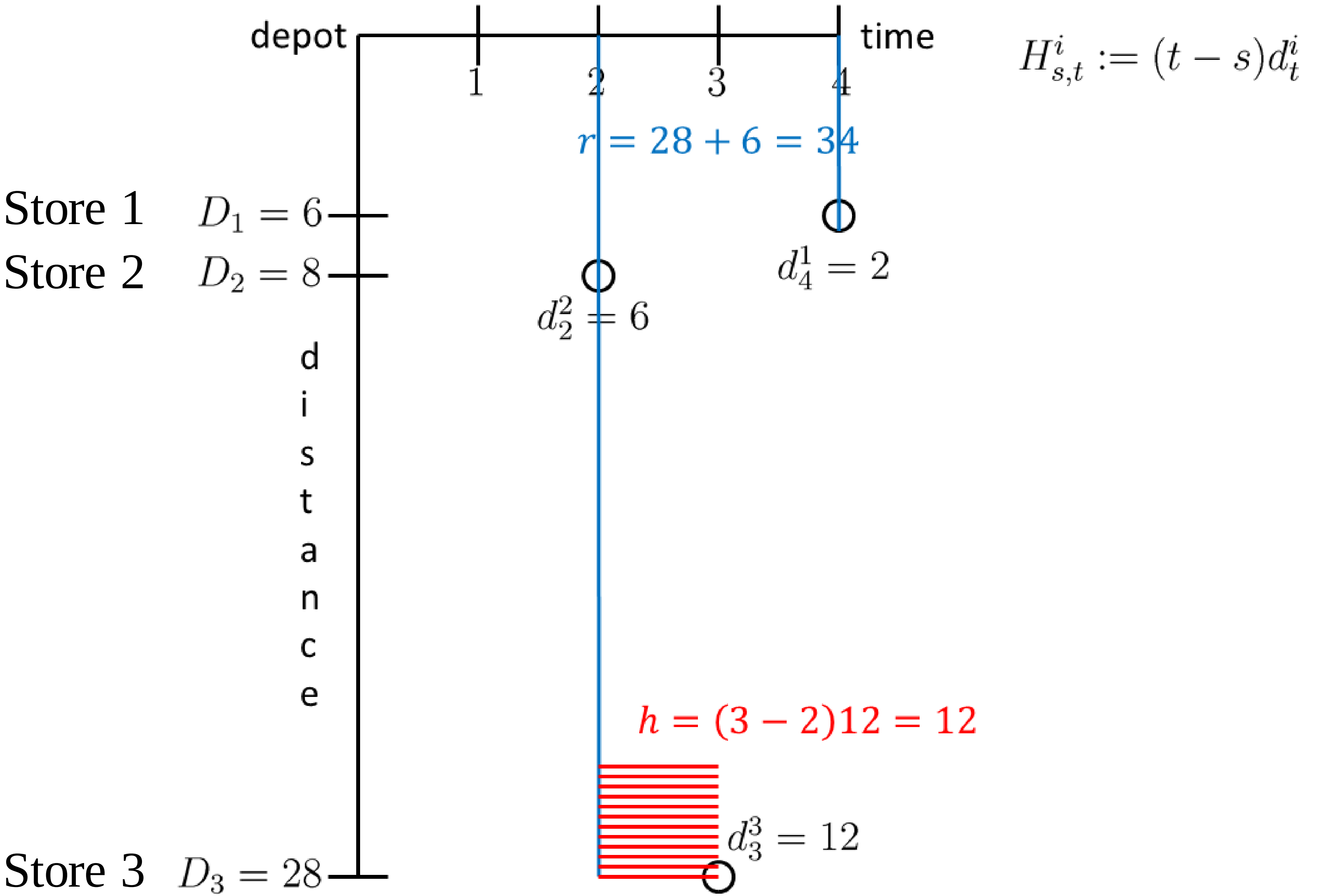
$$H^i_{s,t} := (t-s)d^i_t$$



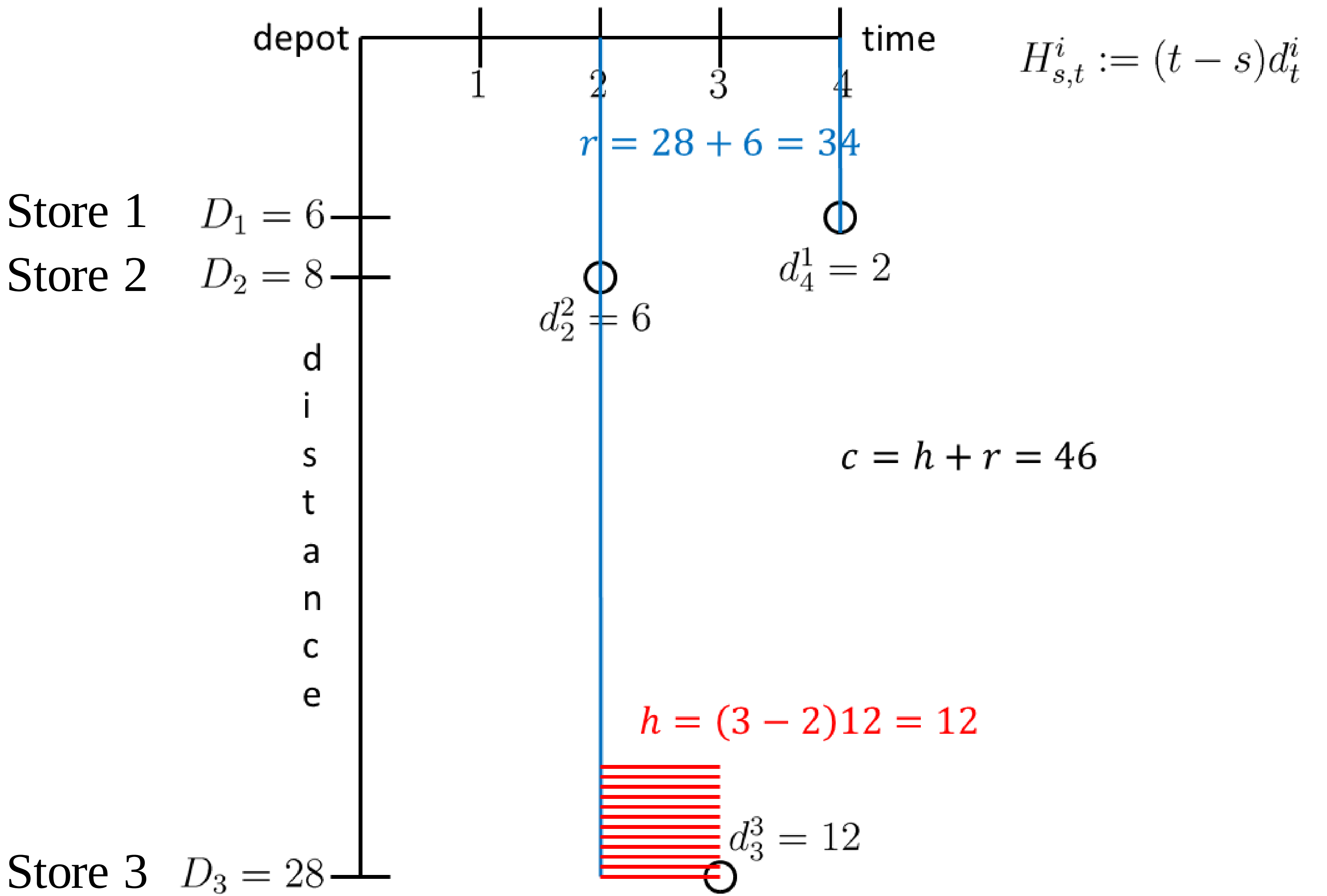
Example of IRP



Example of IRP



Example of IRP



Outline

- Preliminaries
- Related Work
- Results
  - Provable bounds
  - Heuristics

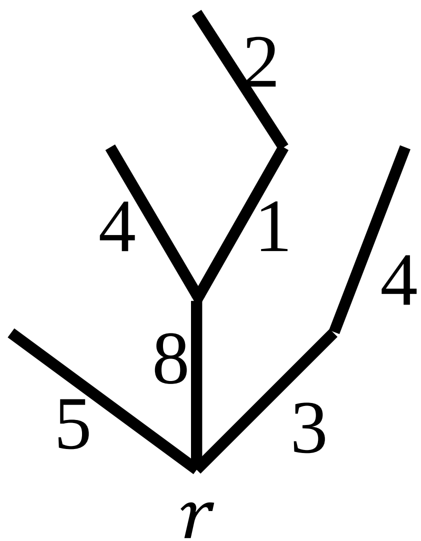
Background - Theoretical

An **α-approx. algorithm** $\boldsymbol{A}$ for a minimization problem is a polyn. time algorithm s.t. for all instances I,

$$c\big(A(I)\big) \leq \alpha OPT(I)$$

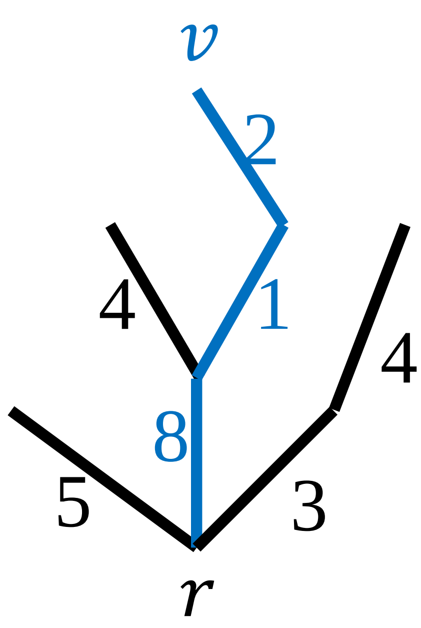
Background - Theoretical

In a **graph metric**, the distance between any pair $\{u, v\}$ is the length of the shortest path in the given graph.



Background - Theoretical

In a **graph metric**, the distance between any pair $\{u, v\}$ is the length of the shortest path in the given graph.



Related Work - Theoretical

- Polytime DP for line metric [Bienkowski et. al. ’13]
- 3-approx. for tree metric [Cheung et. al. ’16]
- Constant approx. for periodic policies [Fukunaga et. al. ‘14]
- $O(\frac{\log T}{\log \log T})$-approx. for any metric [Nagarajan and Shi ’15]

Related Work - Computational

- Branch-and-Cut [Archetti et. al. ‘07]
- Hybrid Heuristic [Archetti et. al. ‘12]
- Solving PCST to Optimality [Ljubic et. al. ‘06]
- Dual-Ascent-based Branch-and-Bound for PCST [Leitner et. al. ‘16]

Outline

- Preliminaries
- Related Work
- Results
  - Provable bounds
  - Heuristics

LP-based Methods for Line Metric

Theorem. 26-approx. by primal dual

Theorem. 5-approx. by linear programming rounding

Outline

- Preliminaries
- Related Work
- Results
  - Provable bounds
  - Heuristics

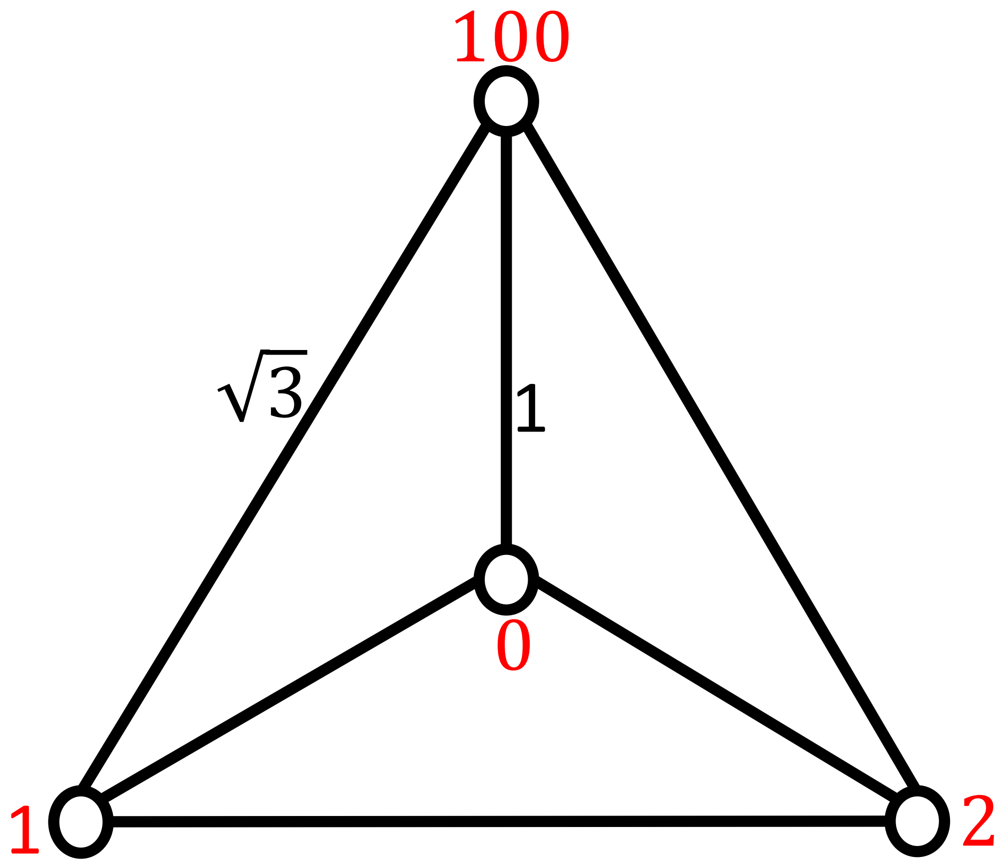
Data Generation Model

- Number of stores $N = 20$
- Number of days $T = 20$
- Demands $d_t^v \sim U(10,100)$
- Holding cost scale $H = 0.01, \ldots, 4.01$
- Linear holding costs s.t. unit holding cost $h_v \sim U(0.01, 0.05)$
- Euclidean distances s.t. $X, Y \sim U(0,500)$

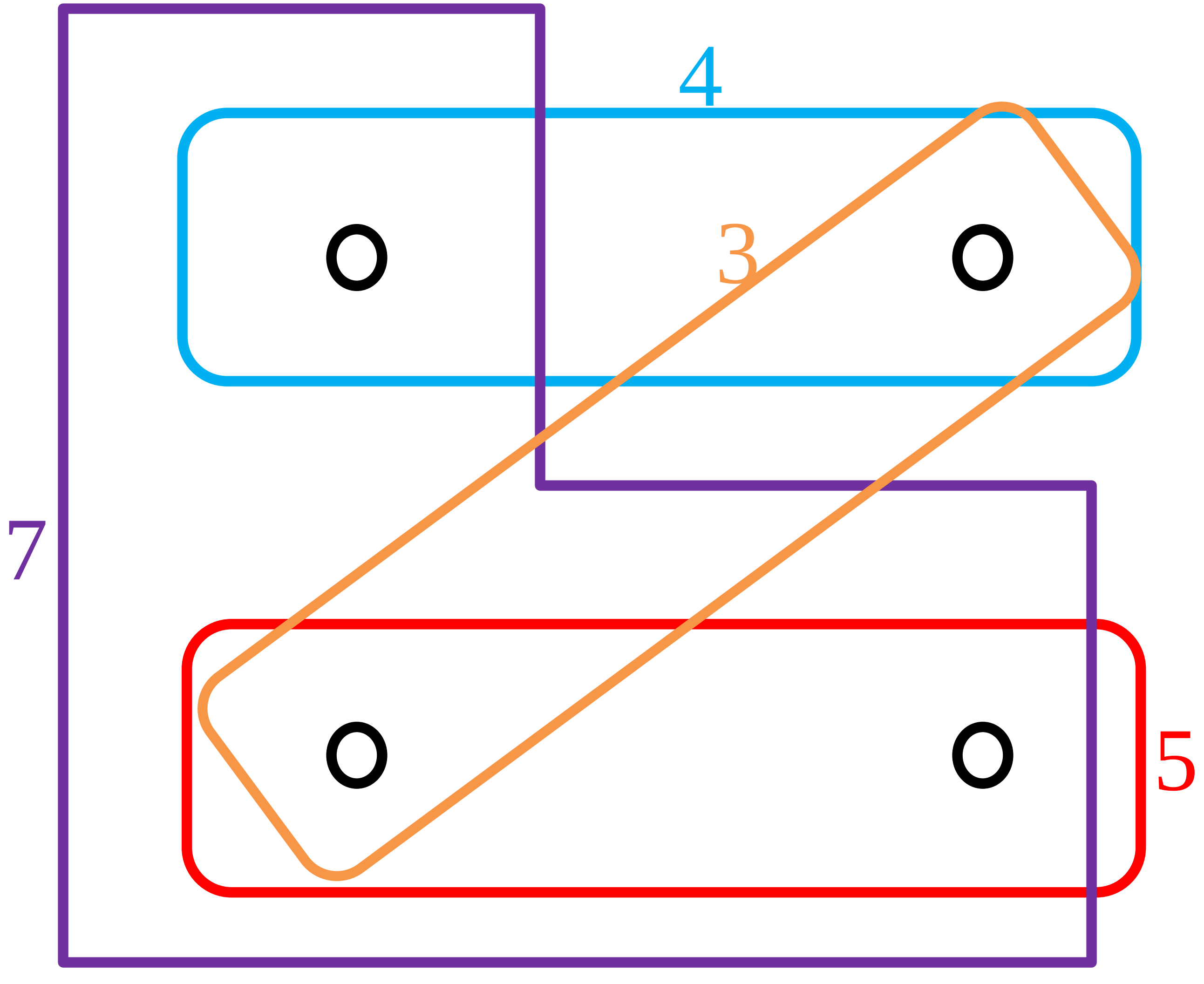
Heuristics for Metric Inventory Routing

- Greedy
- Add local search
- Delete local search
- Delete-add local search

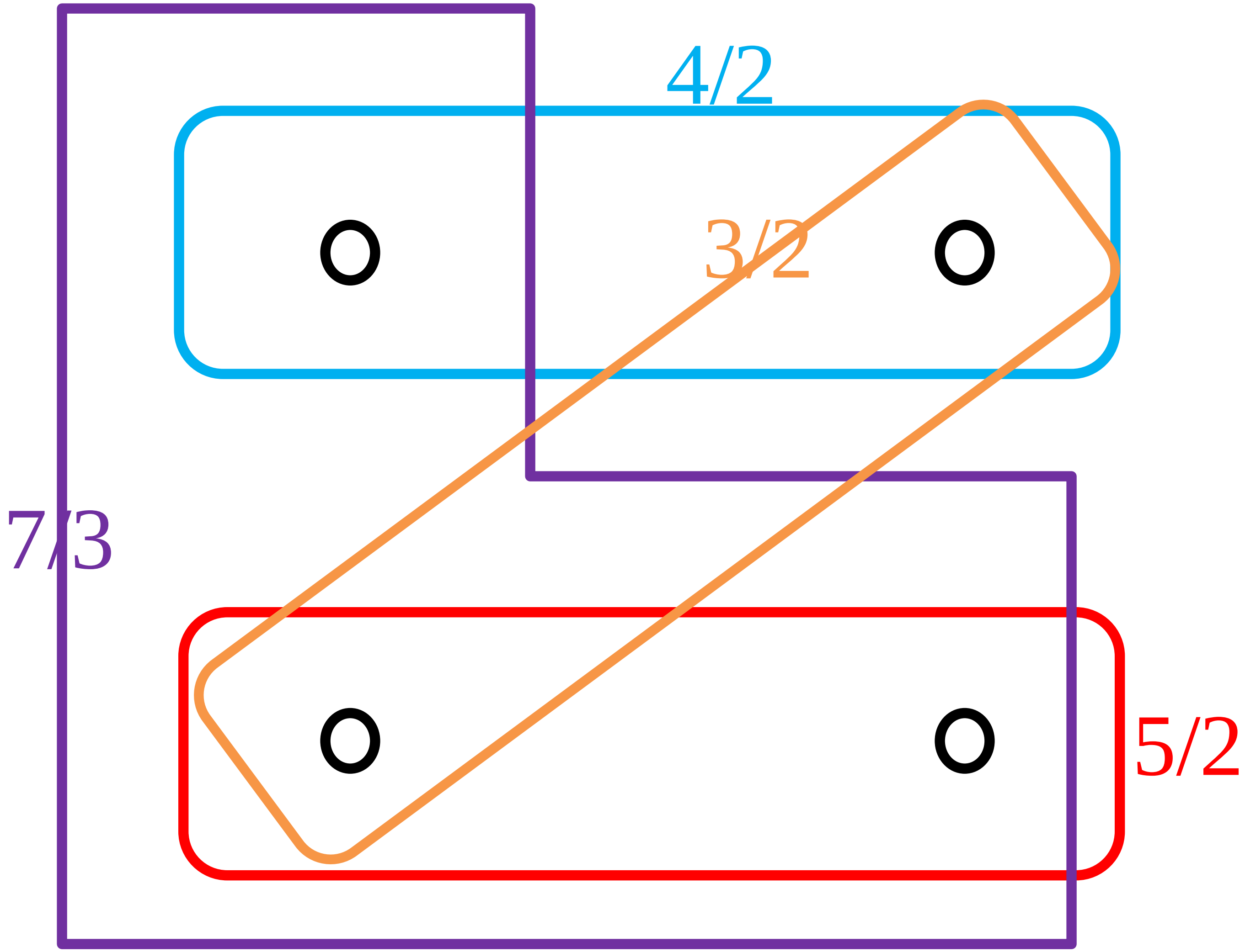
Prize Collecting Steiner Tree



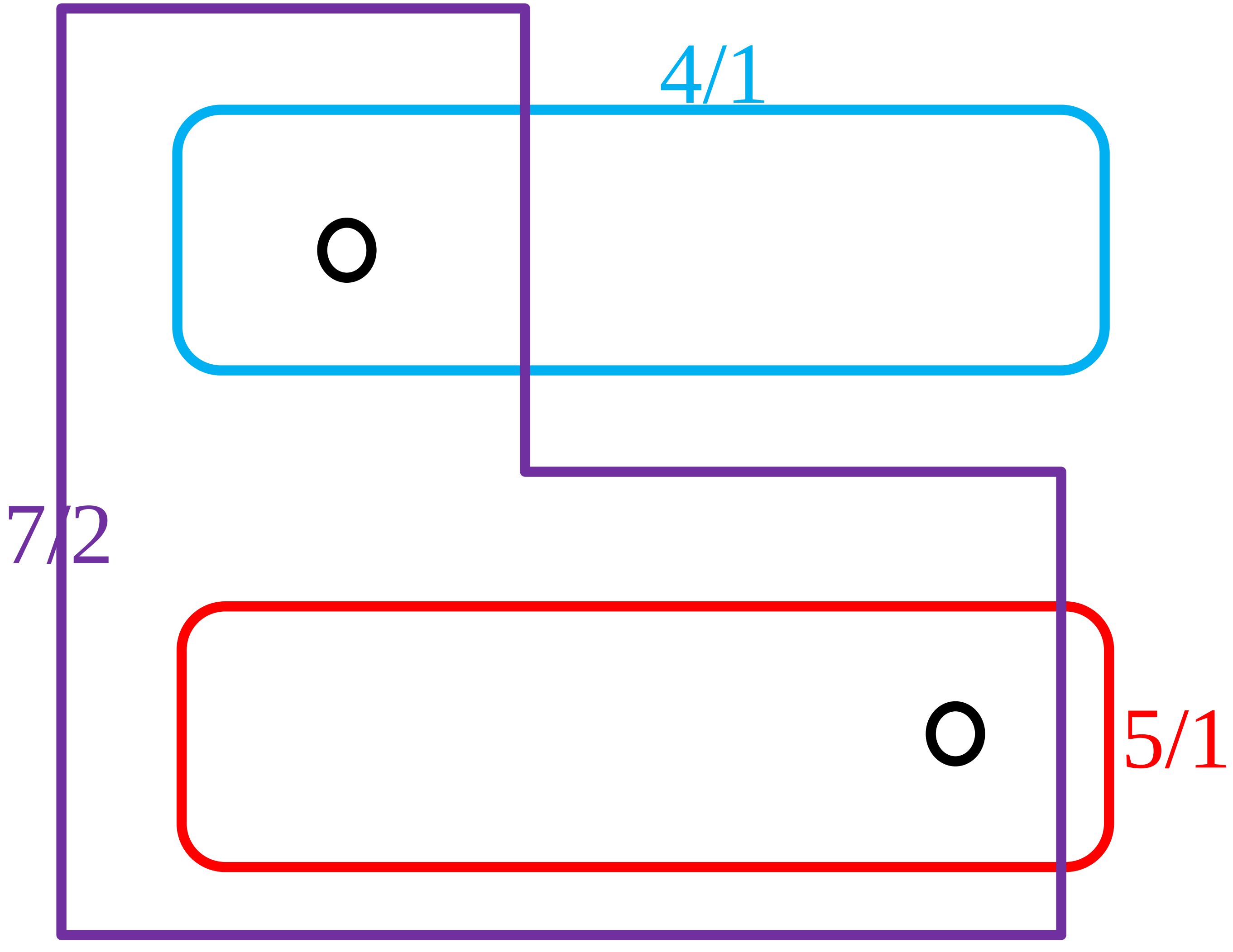
Set Cover



Greedy Set Cover



Greedy Set Cover

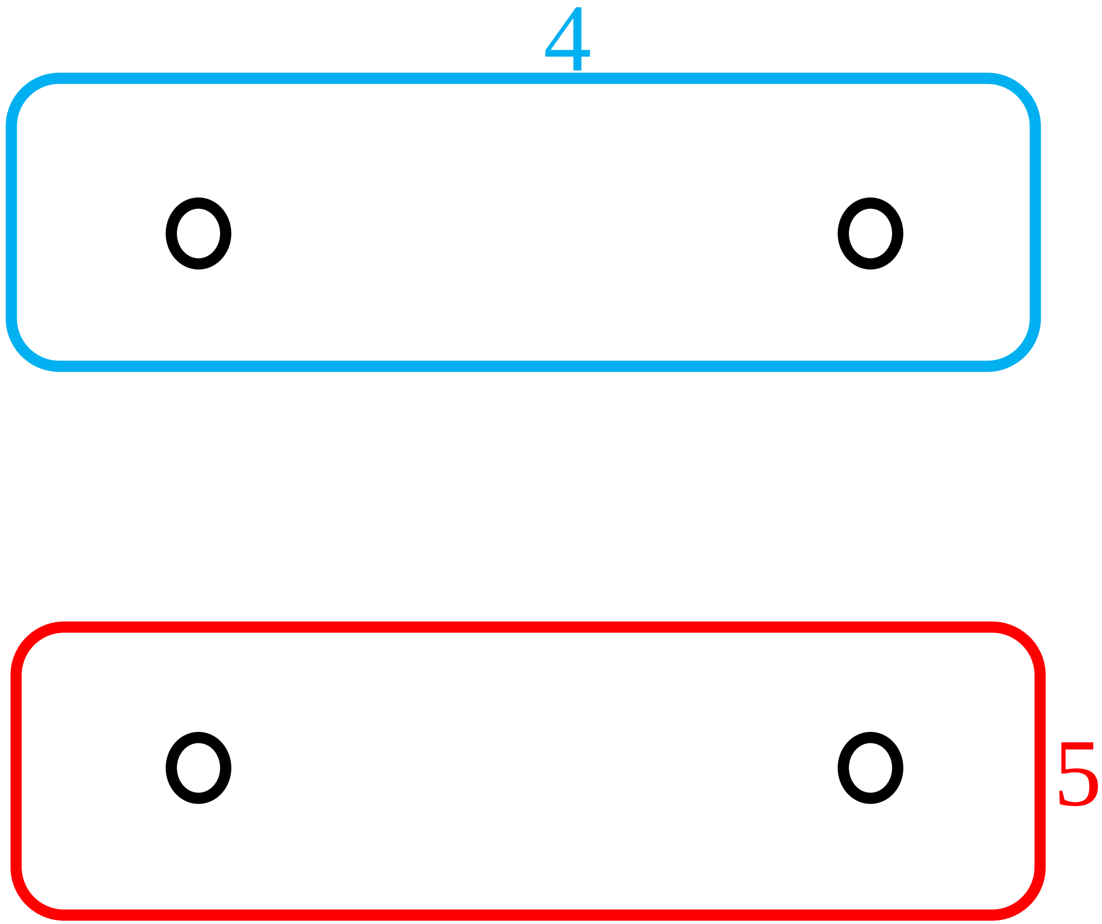


Greedy Set Cover

∞

∞

Set Cover



Greedy Set Cover

While $|U| > 0$

– Find set S of min density $c(S)/|U \cap S|$

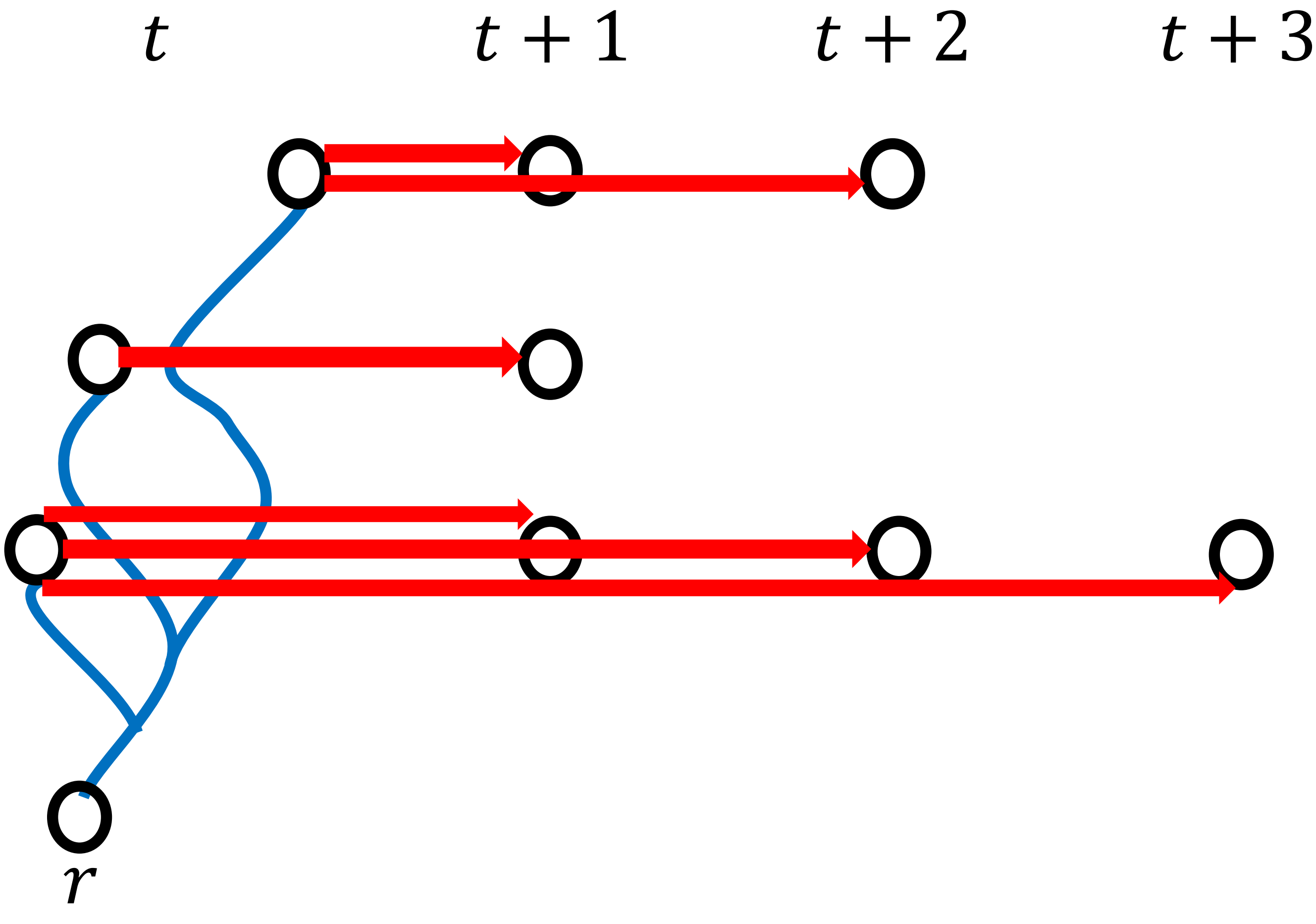
– $U \leftarrow U \setminus S$

$$\Rightarrow c(e_i) \leq \frac{OPT}{n - i + 1}$$

$$\Rightarrow \text{total cost} \leq H_n OPT$$

$$\in O(\log n)\ OPT$$

Greedy for IRP



Greedy for IRP

T_t - tree at time t

D - unserved demands

r - routing cost function

h - holding cost function

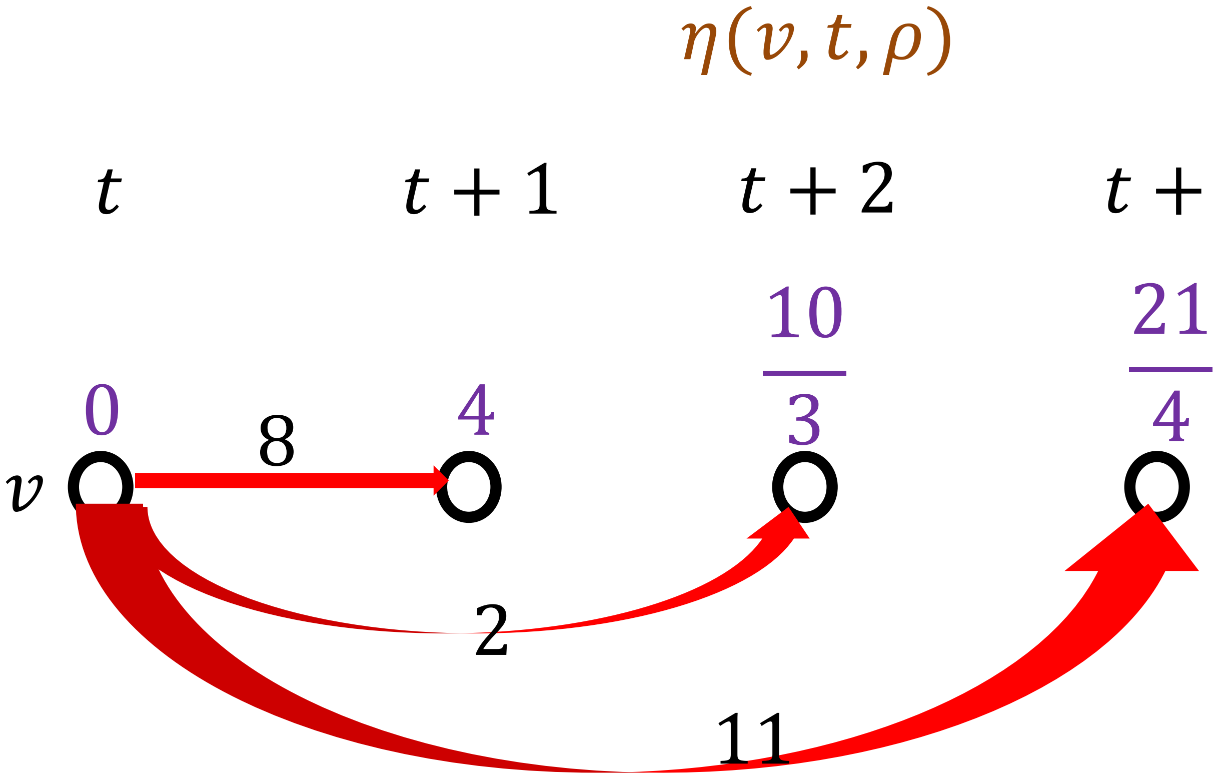
$T_t \leftarrow \emptyset \; \forall \; t$

While $|D| > 0$

- Find day t, a tree T, and coverage set $D(T) \subset D$ minimizing the density $\frac{r(T) + h\left(D(T)\right)}{|D(T)|}$
- $D \leftarrow D \setminus D(T)$
- $T_t \leftarrow T_t \cup T$

Approximate Min Density Set

$\rho = 5$



Approximate Min Density Set

T_t - tree at time t

D - unserved demands

r - routing cost function

h - holding cost function

- Guess best density value $\rho = \frac{r(T) + h(D(T))}{|D(T)|}$ and time t of visit
- $\eta(v, t, \rho) =$ max # uncovered demand points at store v time $\geq t$ such that the average holding cost to serve them stays $\leq \rho$
- PCST instance:

  Penalties $\pi(v) \coloneqq \eta(v, t, \rho) * \rho$

  Edge weights $w(e) \coloneqq w_{IRP}(e)$

Approximate Min Density Set

T_t - tree at time t

D - unserved demands

r - routing cost function

h - holding cost function

How good is the density?

T an optimal PCST tree ⇒

- $r(T) \leq 2\, dual(T)$
  $\leq 2\, \pi(T)$
  $\leq 2 \sum_{v \in T} \eta(v, t, \rho) \cdot \rho$
- $h(T) \leq \sum_{v \in T} \eta(v, t, \rho) \cdot \rho$
- $|D(T)| = \sum_{v \in T} \eta(v, t, \rho)$

⇒ Density $\leq 3\rho$

⇒ $O(\log NT)$-approx. overall

Local Search Framework

1. Initialize a feasible solution
2. Apply operation as long as $\Delta(\text{total cost}) < 0$
3. Stop when no more improvements exist or when time limit reached

Add Local Search

1. Serve all stores on day 1
2. Apply ADD(s) if $\exists\ s$ s.t. $\Delta(\text{total cost}) < 0$
3. Stop when no more improvements exist

Delete Local Search

1. Serve all stores on their deadline day
2. Apply DELETE(s) if $\exists\ s$ s.t. $\Delta(\text{total cost}) < 0$
3. Stop when no more improvements exist

Delete-Add Local Search

1. Serve all stores on their deadline day
2. Apply DELETE-ADD(s) if $\exists\ s$ s.t. $\Delta(\text{total cost}) < 0$
3. Stop within 30 seconds

Add Operation

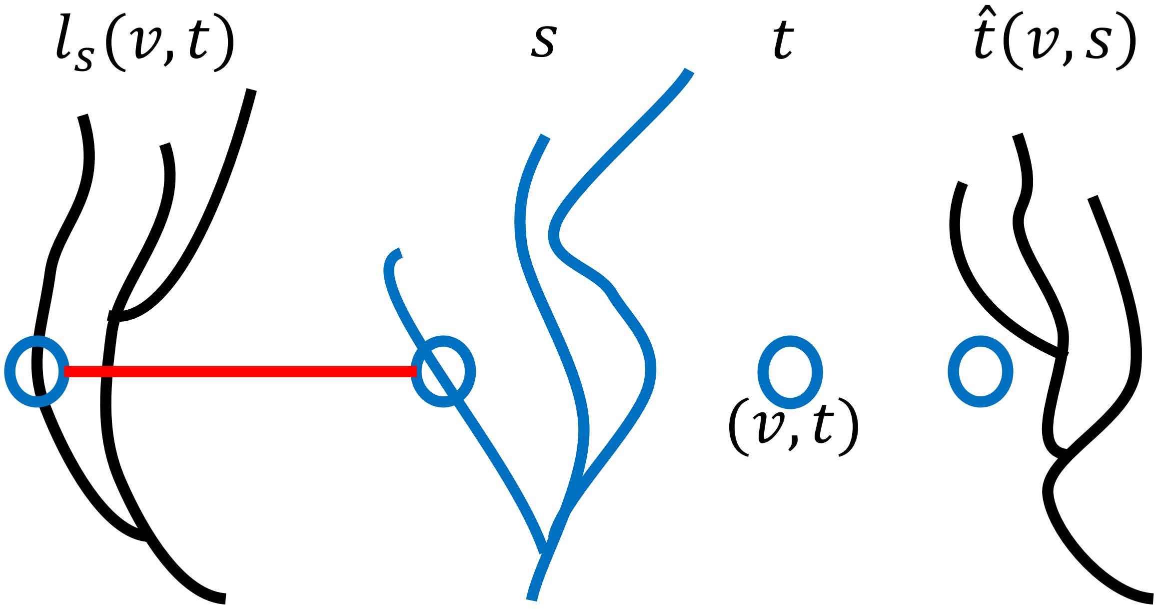
$\hat{t}(v,s)$ - latest day after s with no visit to v

$l_s(v,t)$ - latest day before s that serves (v,t)

ADD(s)

PCST instance:

Penalties



Add Operation

$\hat{t}(v, s)$ - latest day after s with no visit to v

$l_s(v, t)$ - latest visit before s that serves (v, t)

ADD(s)

PCST instance:

Penalties

$$\pi_s(v) := \sum_{t=s}^{\hat{t}(v,s)} h^v_{l_s(v,t),s}\, d^v_t$$

Edge weights $w(e) := w_{IRP}(e)$

$\Delta(\text{total cost}) = w(T_{PCST}) - \pi(T_{PCST})$

Delete Operation

$\hat{t}(v,s)$ - latest day after s with no visit to v

$l_s(v,t)$ - latest visit before s that serves (v,t)

T_t - tree at time t

DELETE(s)

Penalties

$$\pi_s(v) := \sum_{t=s}^{\hat{t}(v,s)} h^v_{l_s(v,t),s}\, d^v_t$$

$$\Delta(\text{total cost}) = -w(T_s) + \pi(T_s)$$

Delete-Add Operation

$\hat{t}(v,s)$ - latest day after s with no visit to v

$l_s(v,t)$ - latest visit before s that serves (v,t)

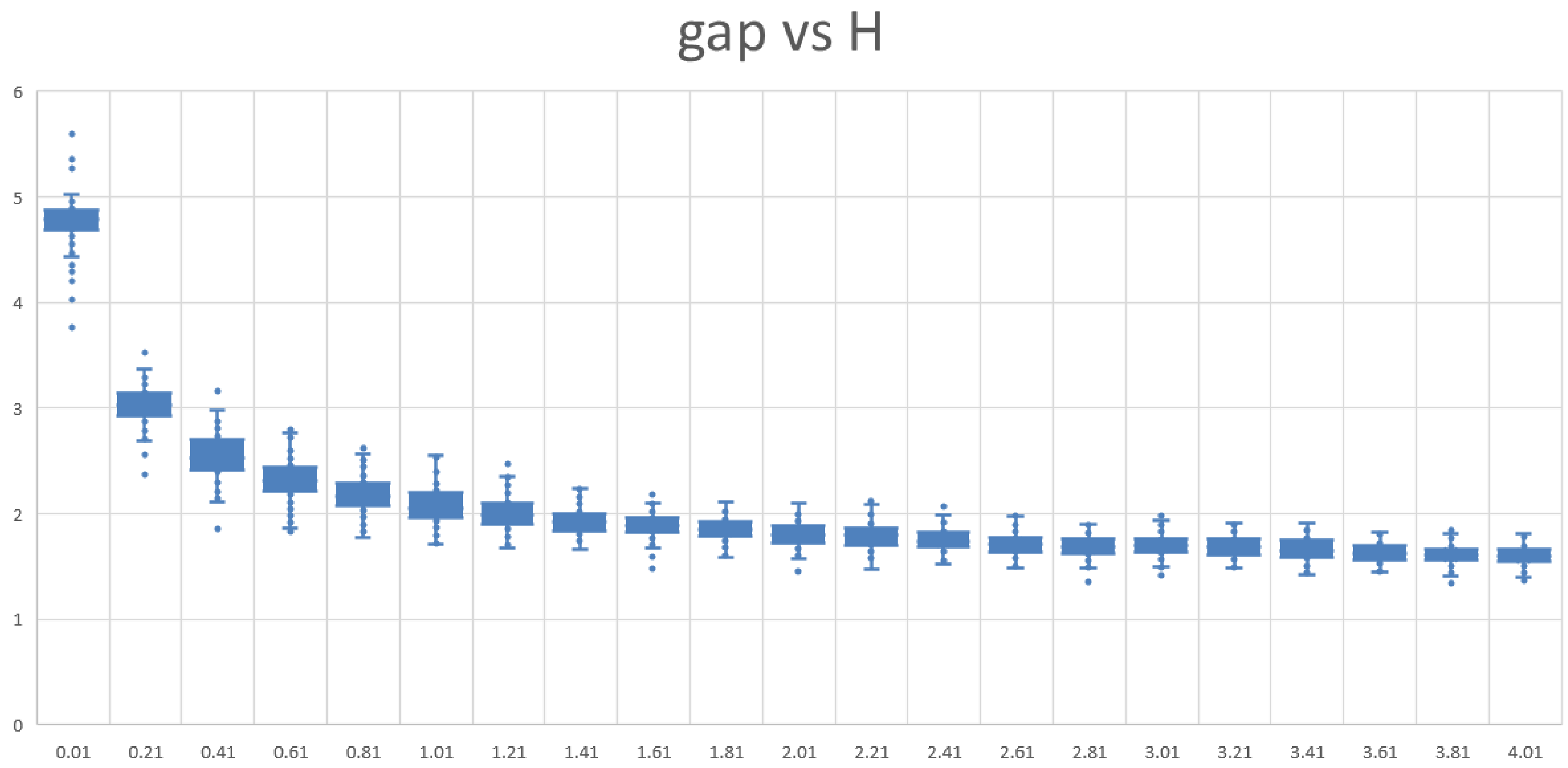
T_t - tree at time t

DELETE-ADD(s)

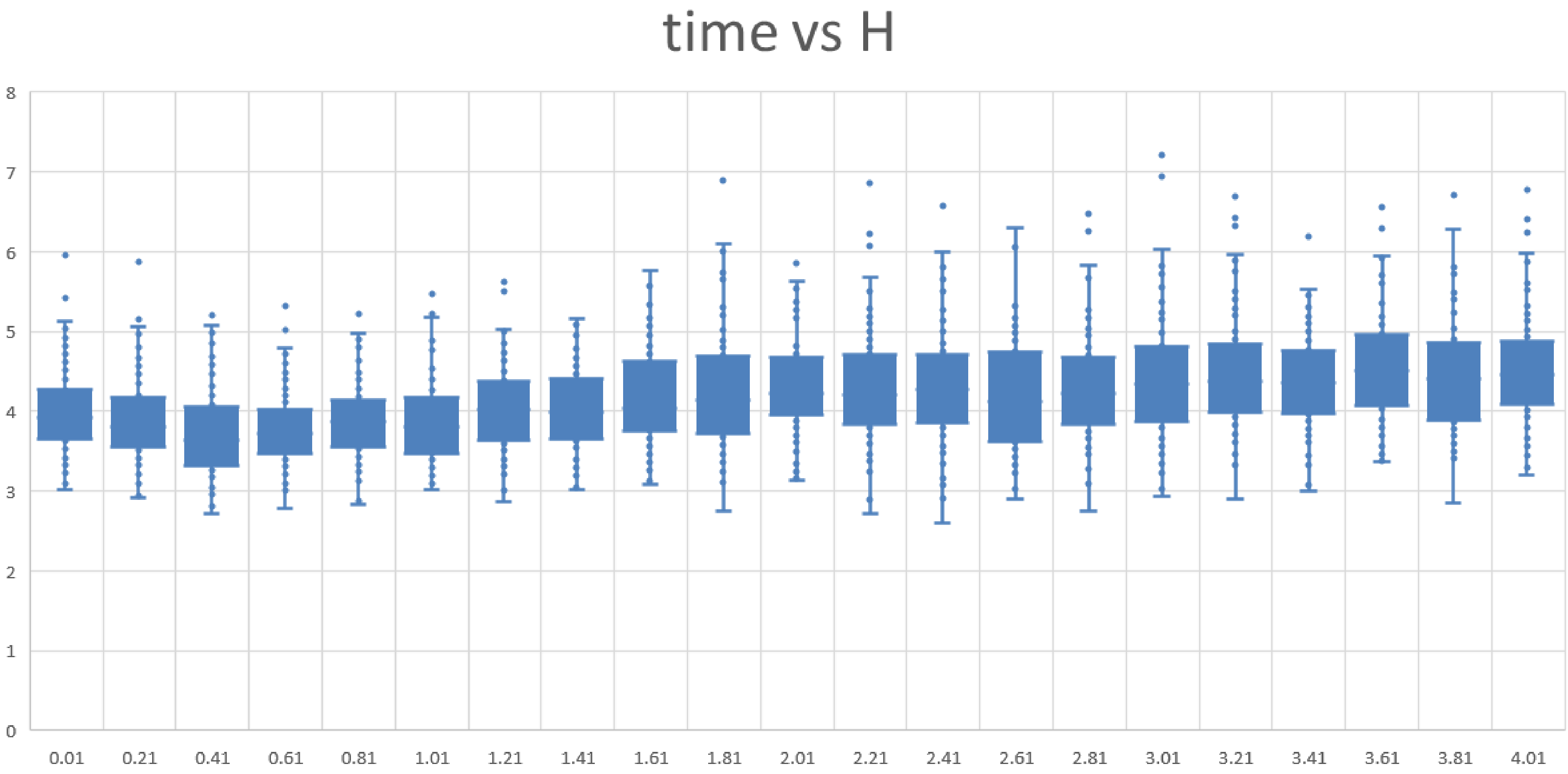
Apply DELETE(s) first, then ADD(s) on the remaining solution

$$\Delta(\text{total cost}) = -w(T_s) + \pi(T_s) + w(T_{PCST}) - \pi(T_{PCST})$$

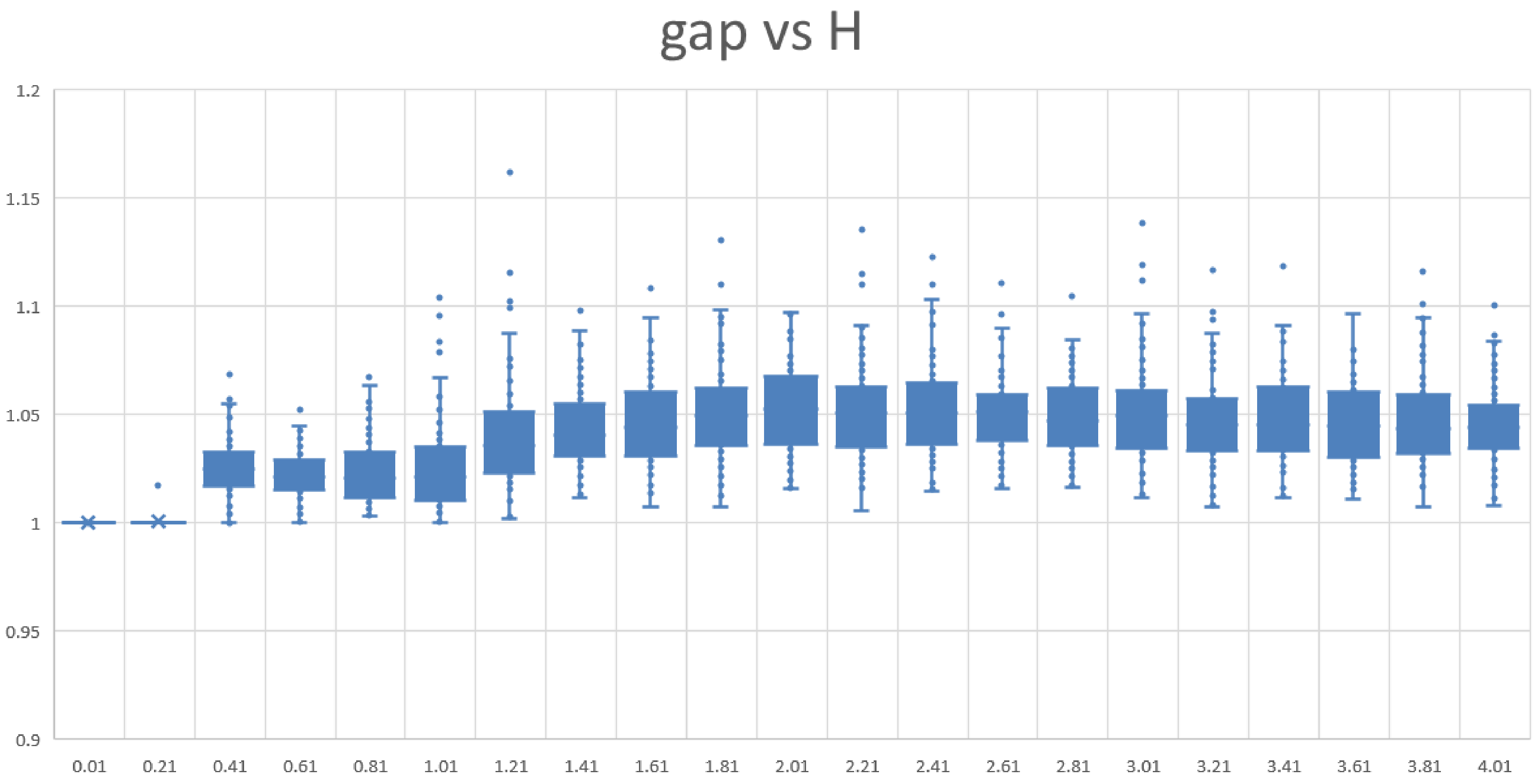
Performance of Greedy



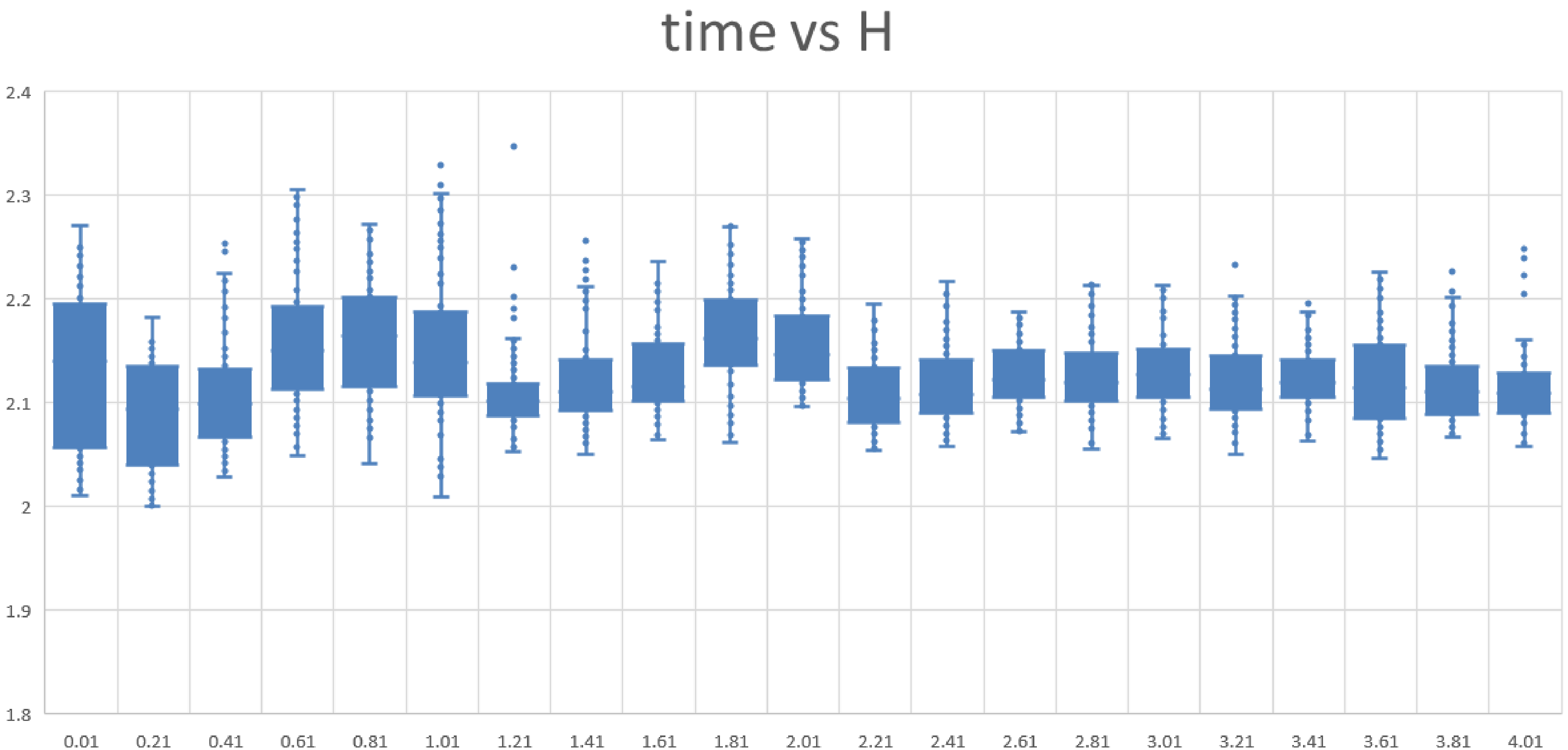
Performance of Greedy



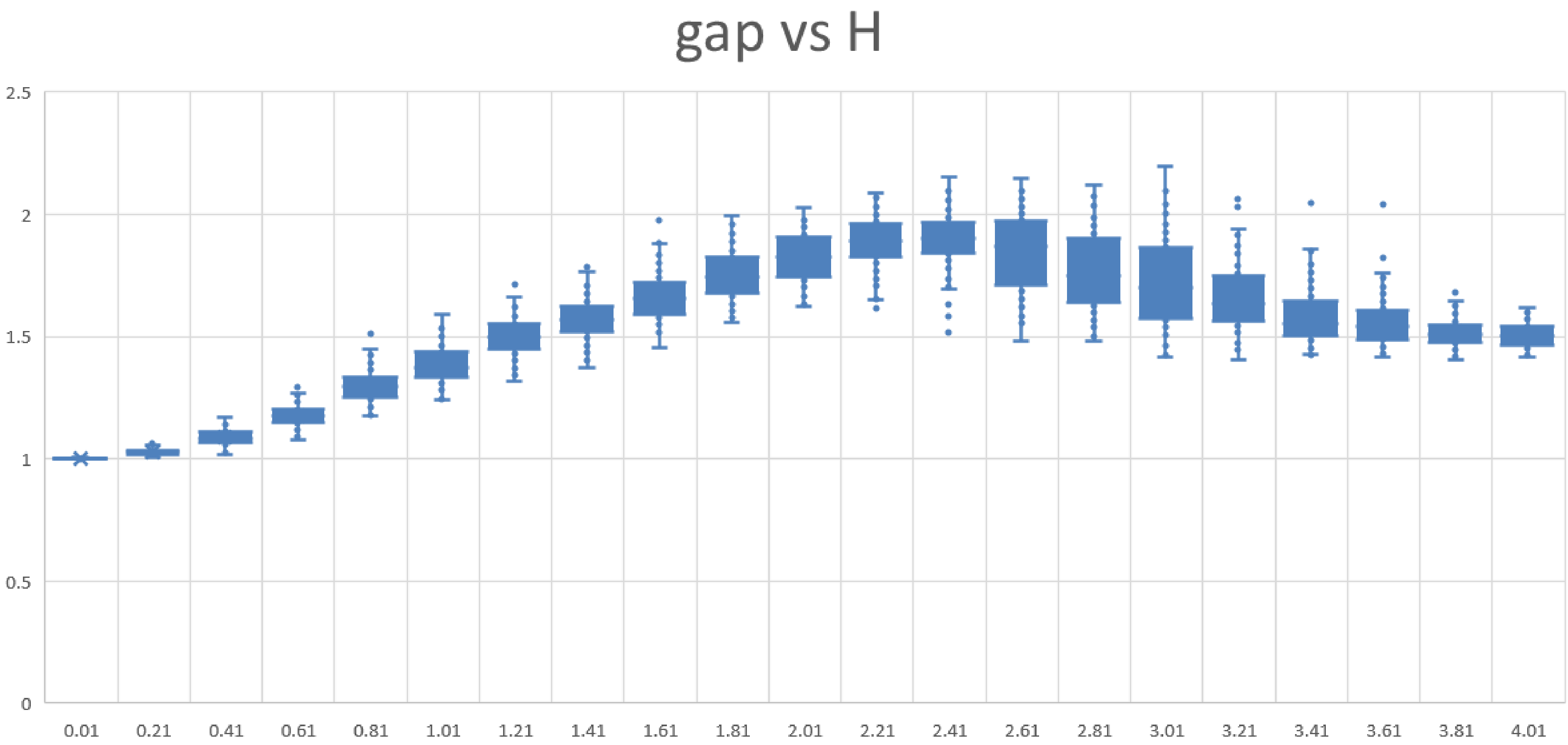
Performance of Add



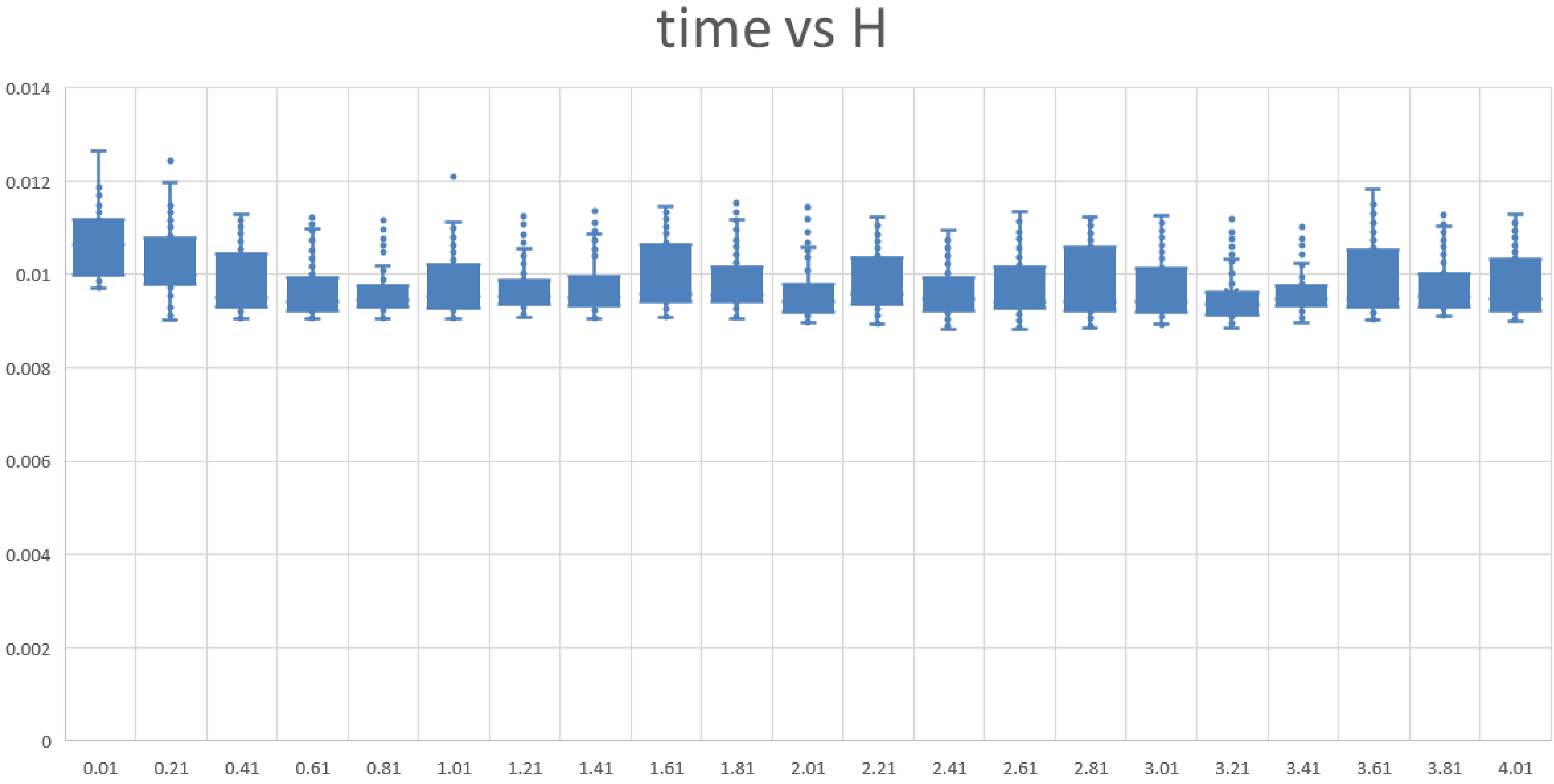
Performance of Add



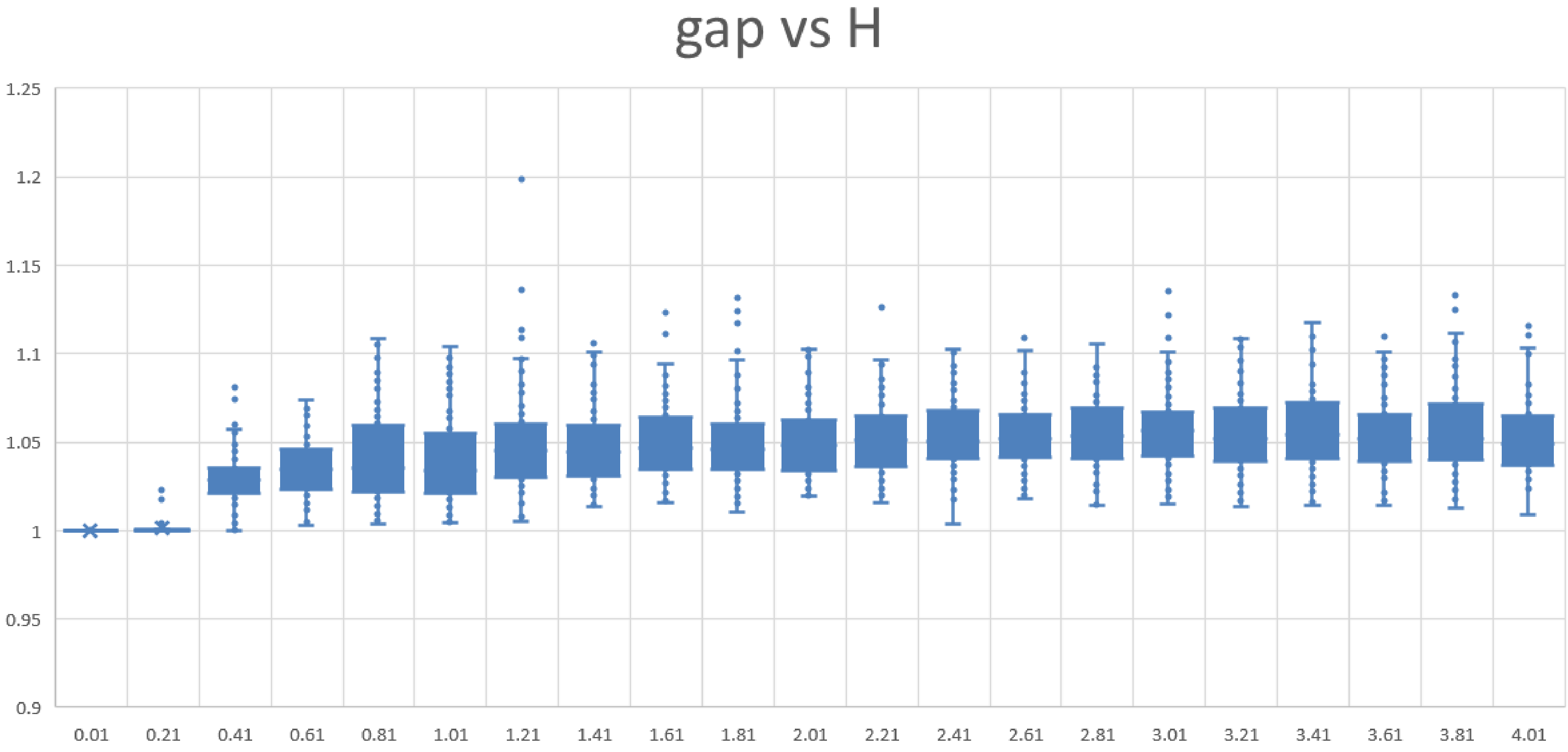
Performance of Delete



Performance of Delete



Performance of Delete-Add



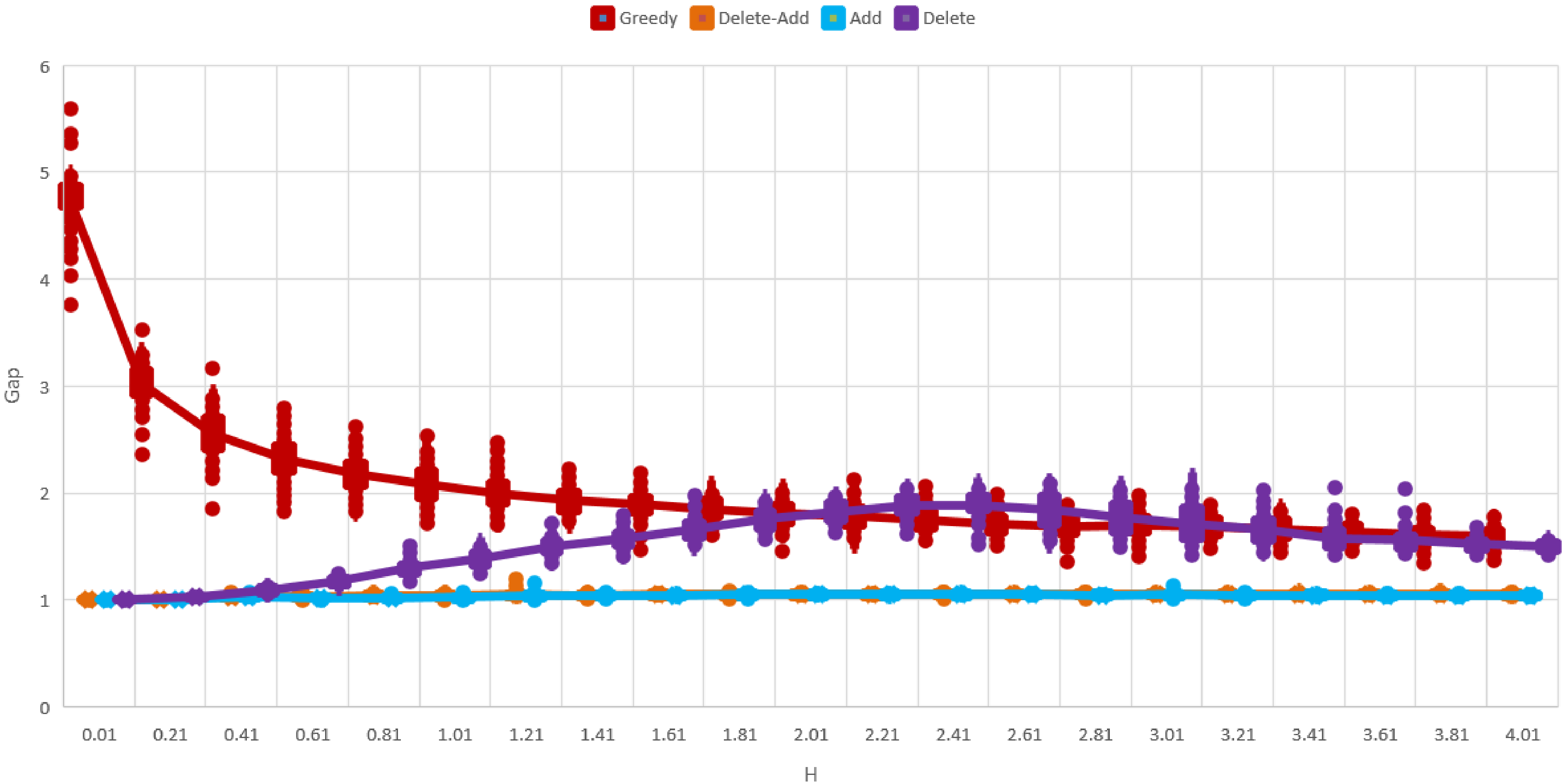
Summary

- IRP – challenging to solve optimally due to intertemporal dependencies
- PCST – an effective tool to obtain near-optimal solutions for IRP
- Best overall heuristic – ADD local search, both fast and accurate

Thank You!

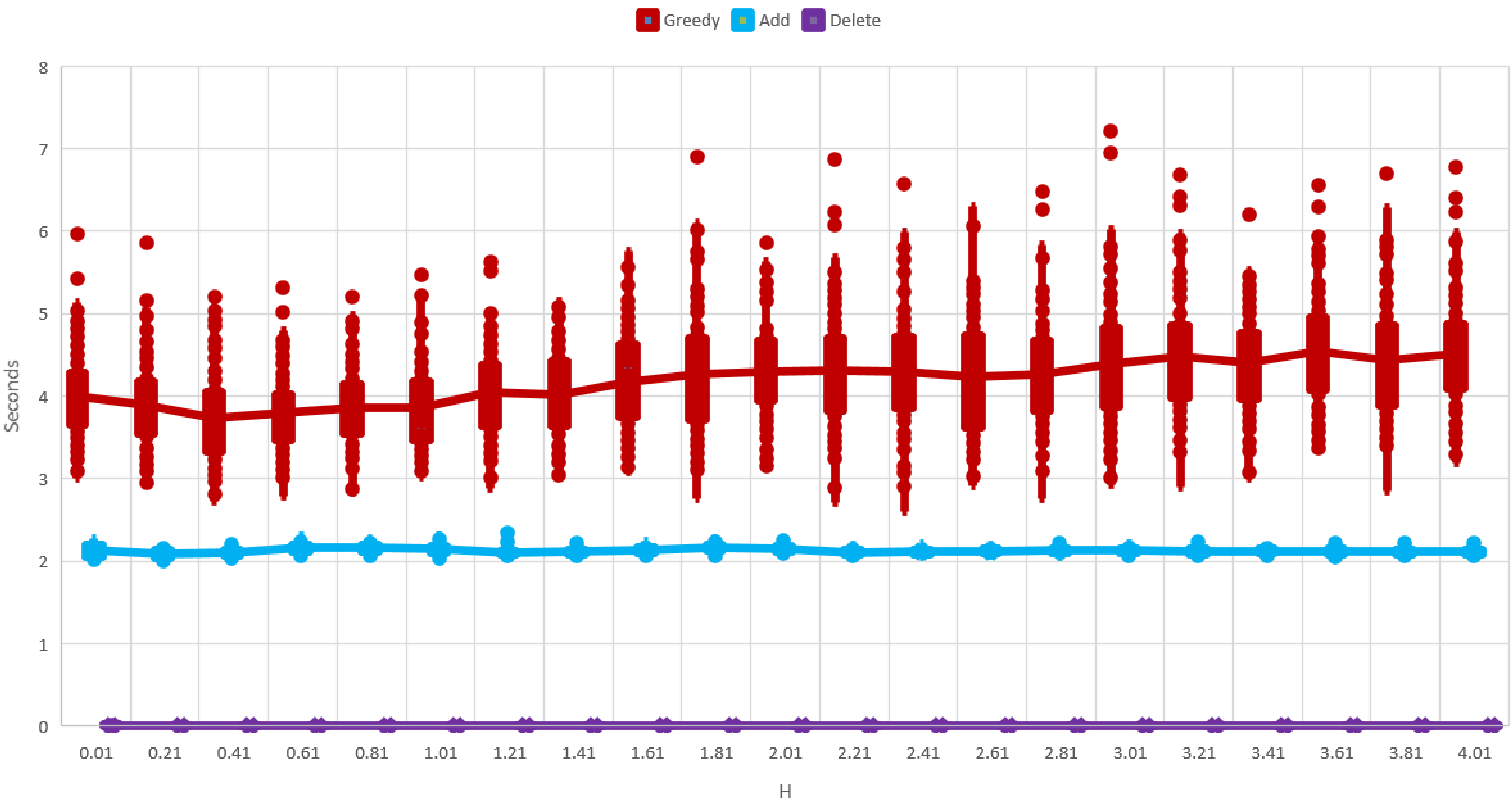
Appendix

Gap



Greedy
Delete-Add
Add
Delete
Gap
0
1
2
3
4
5
6
0.01
0.21
0.41
0.61
0.81
1.01
1.21
1.41
1.61
1.81
2.01
2.21
2.41
2.61
2.81
3.01
3.21
3.41
3.61
3.81
4.01
H

Runtime



Greedy
Add
Delete
8
7
6
5
4
3
2
1
0
Seconds
0.01
0.21
0.41
0.61
0.81
1.01
1.21
1.41
1.61
1.81
2.01
2.21
2.41
2.61
2.81
3.01
3.21
3.41
3.61
3.81
4.01
H

parameter	definition	value
N	Number of stores	10
T	Number of days	10
L	Routing cost scale	3
p	Positive demand probability	.2

parameter	definition	range	increment
N	number of locations	5 to 35	5
T	number of days	5 to 35	5
L	routing scale	0.1 to 8.1	0.5
p	positive demand probability	0.01 to 0.35	0.02

Primal Dual vs DP Performance

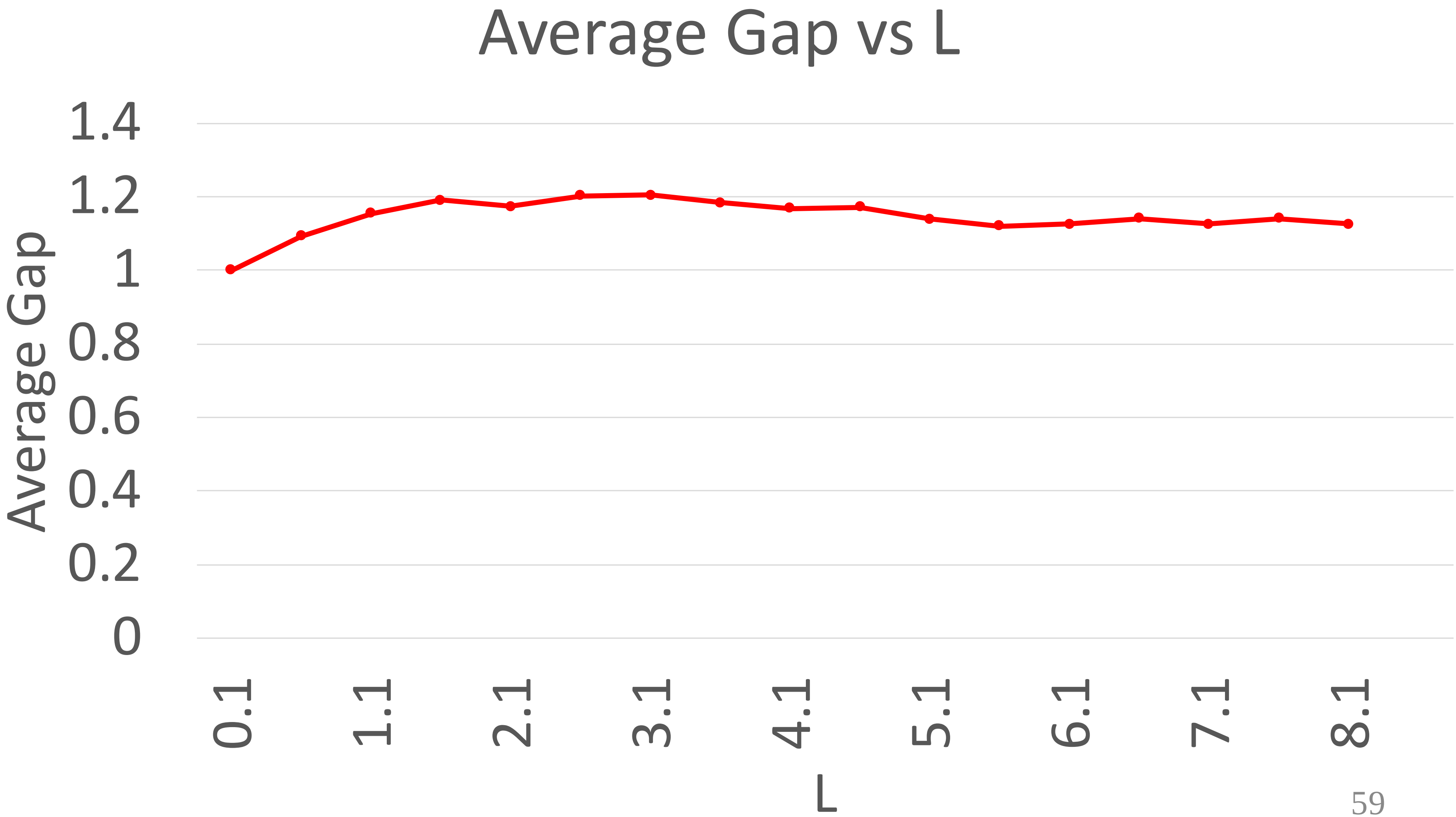
Data Generation

Demands: with probability p: $NB(r, \frac{\alpha}{\alpha+1})$; with probability $1-p$: 0

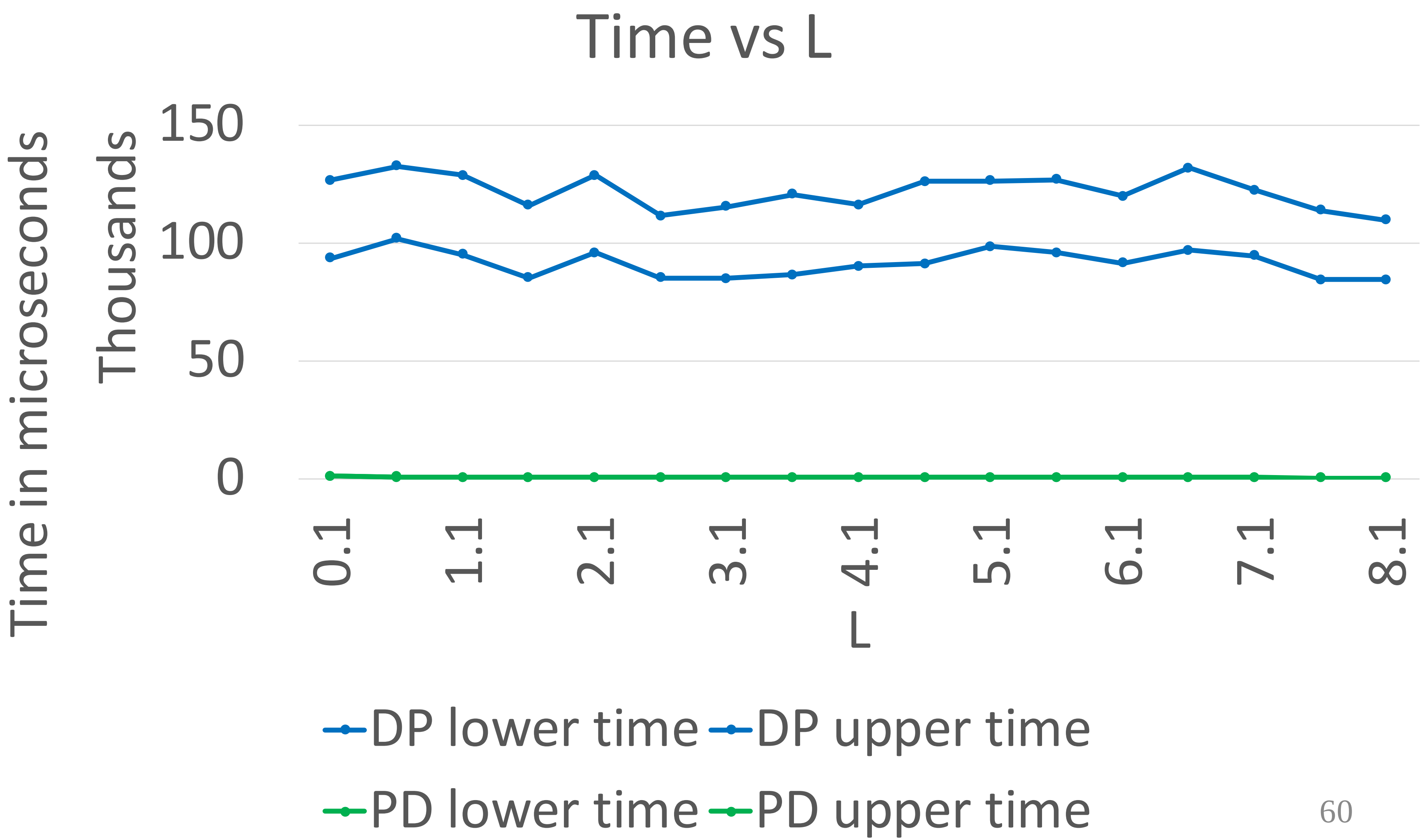
Location distances ~ $\exp(\lambda)$

parameter	definition	value
N	Number of stores	10
T	Number of days	10
L	Routing cost scale	0.1, 0.6, … , 8.1
p	Positive demand probability	.2

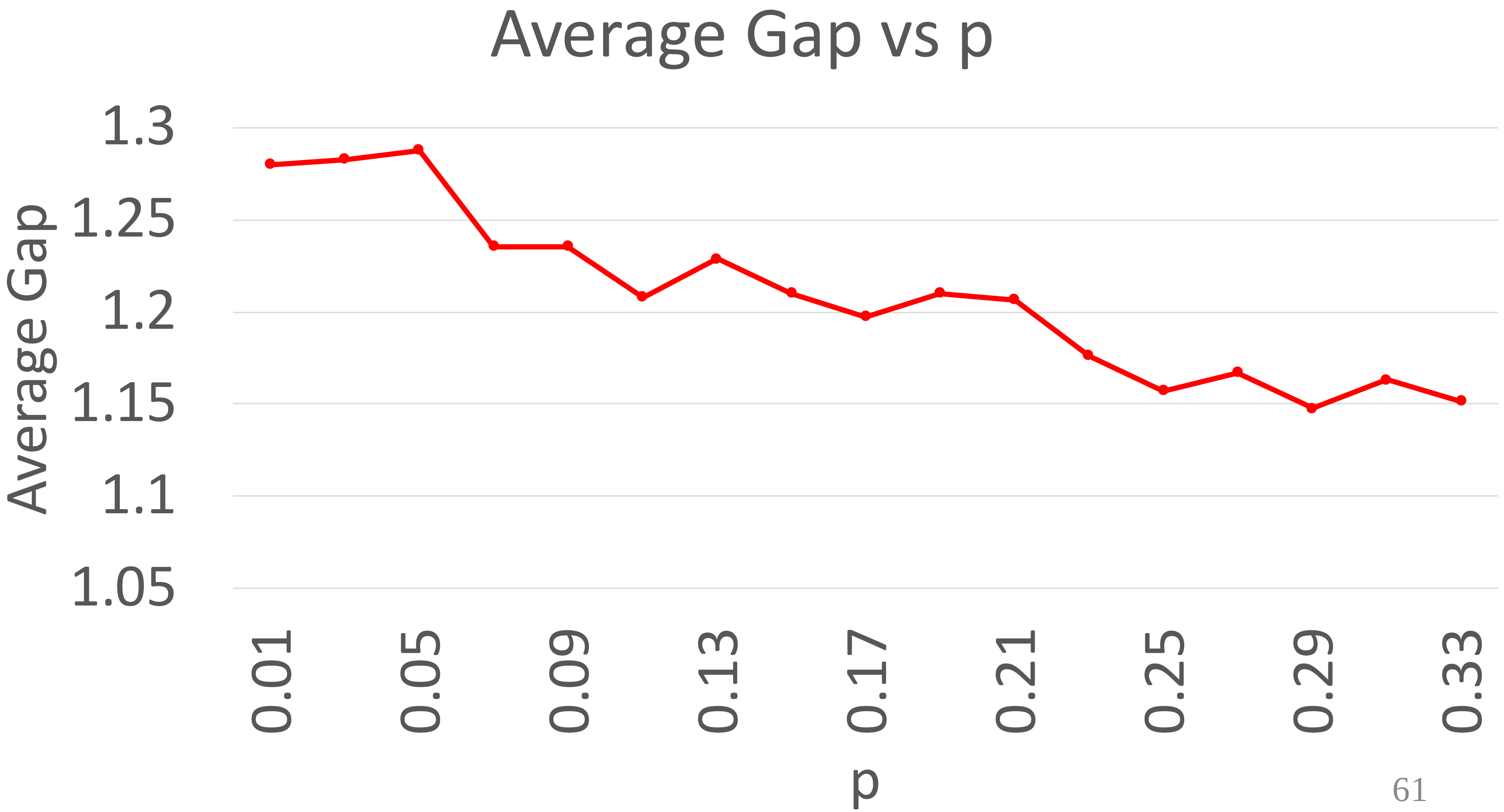
Primal Dual vs DP Performance



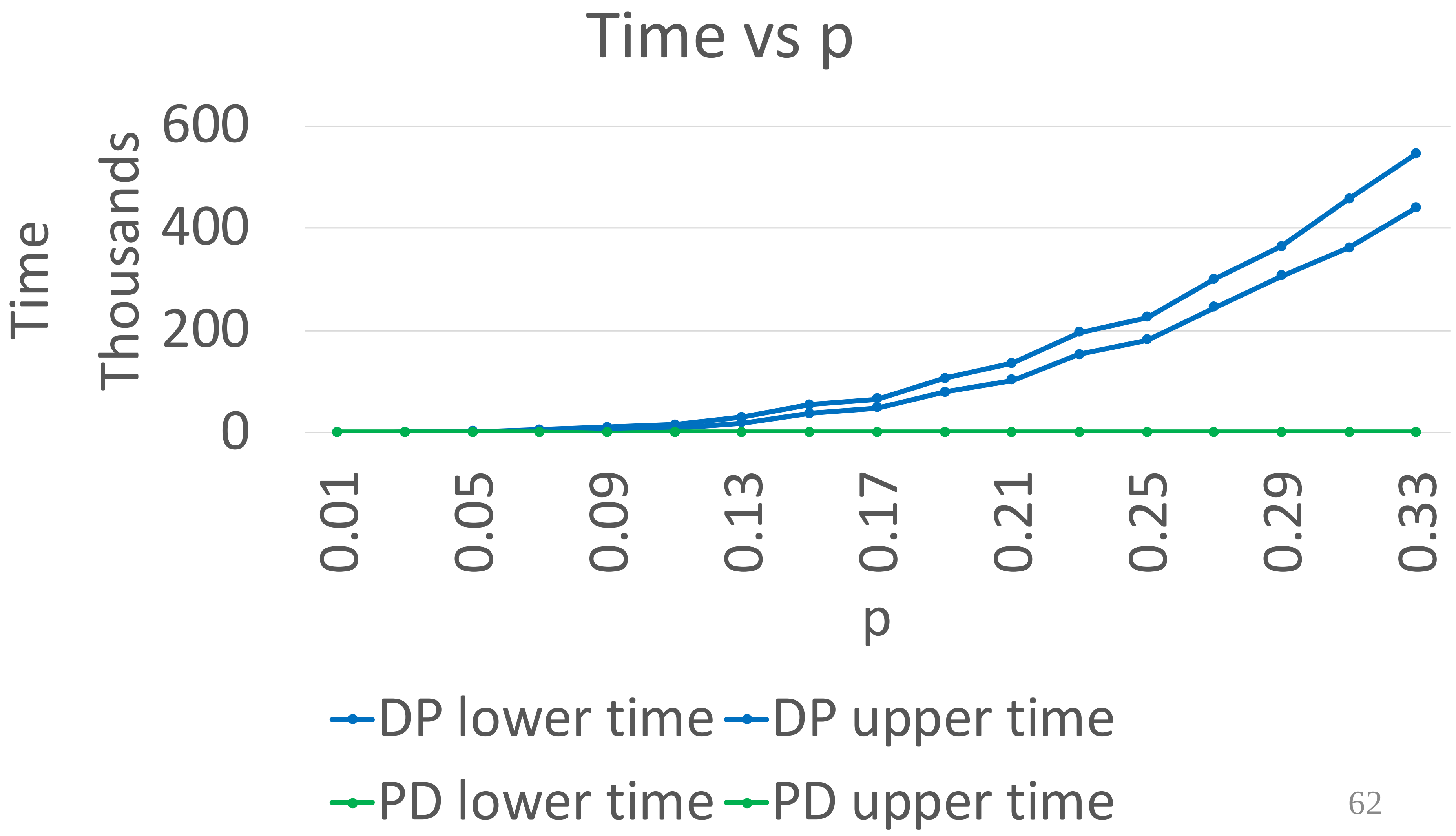
Primal Dual vs DP Performance



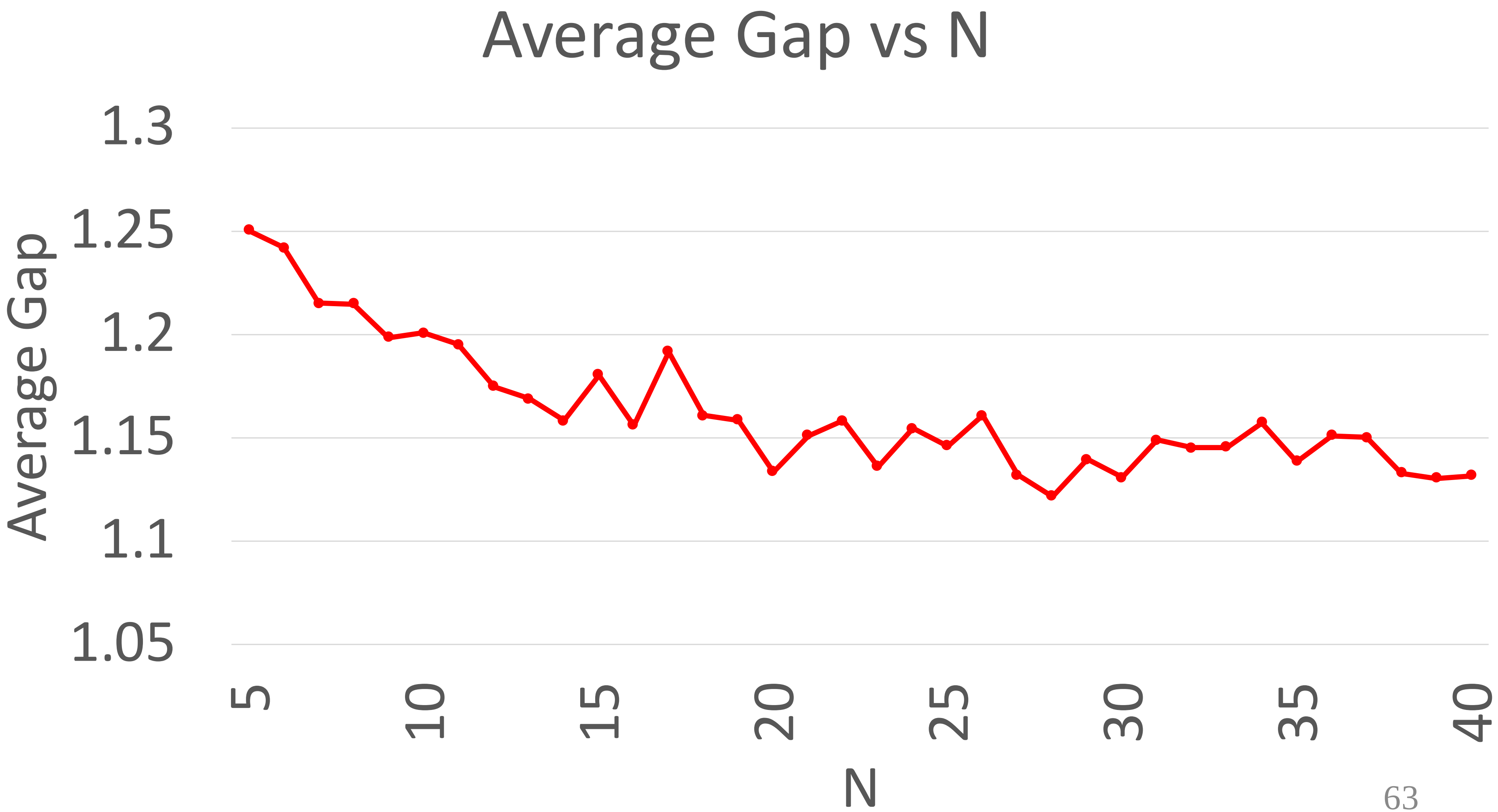
Primal Dual vs DP Performance



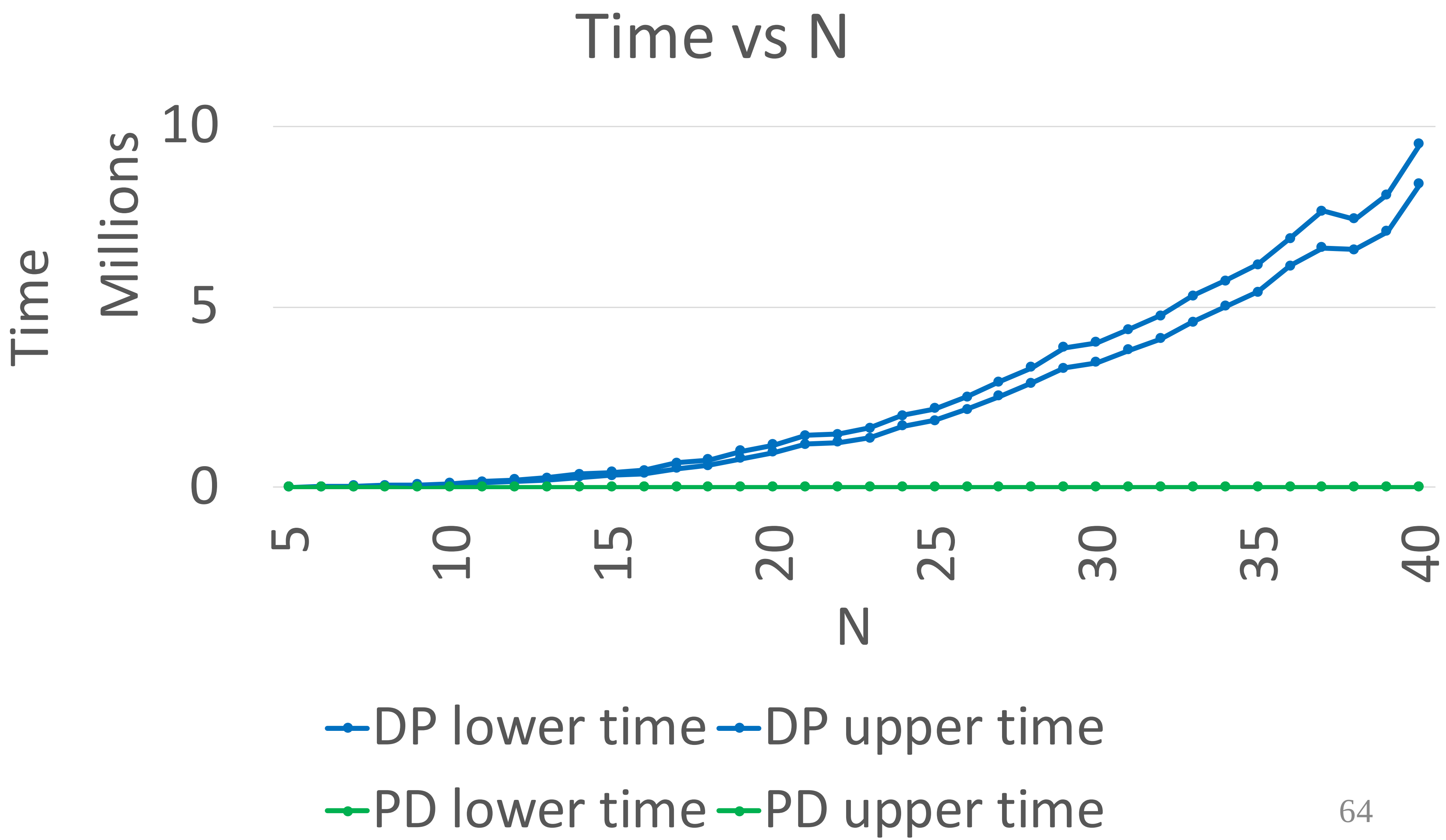
Primal Dual vs DP Performance



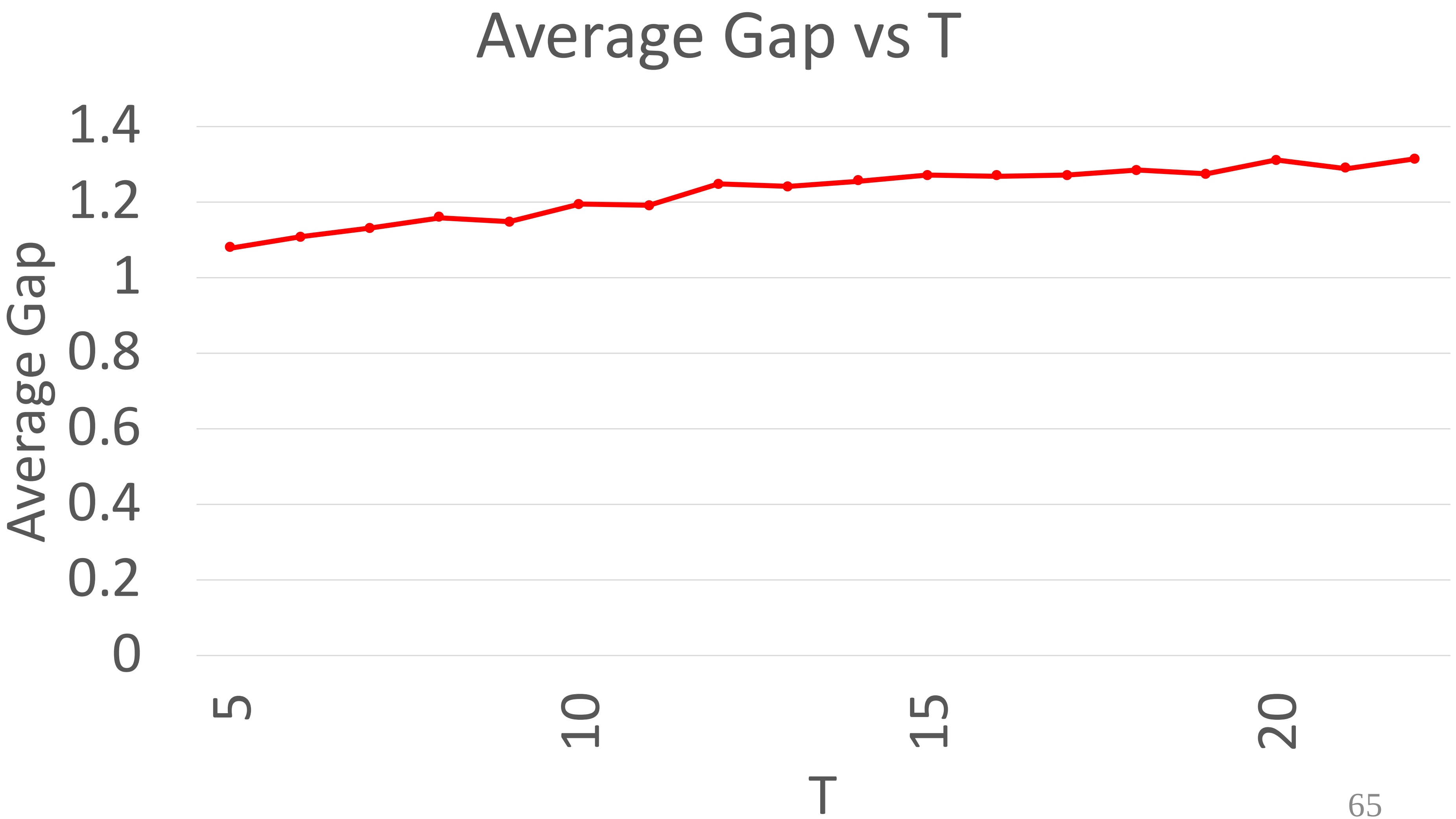
Primal Dual vs DP Performance



Primal Dual vs DP Performance



Primal Dual vs DP Performance



Primal Dual vs DP Performance