15.09.2025

THE SERVICE POINTS’ LOCATION AND CAPACITY PROBLEM

Tal Raviv

Presentation by Francesca Craievich

THE PROBLEM

We're talking about automated parcel lockers (like Amazon Lockers) that serve last-mile delivery - the final leg from courier to customer

To build a network of these lockers, we need to decide:

1. Where to place them
2. How large they should be

OBJECTIVE: minimize costs

SOLUTION

Challenge: Real-world parcel delivery is stochastic
Random daily arrivals
Unpredictable pickup times

↓

PWL Pre-compute stochastic behavior
Create piecewise linear approximations
Capture all randomness in advance

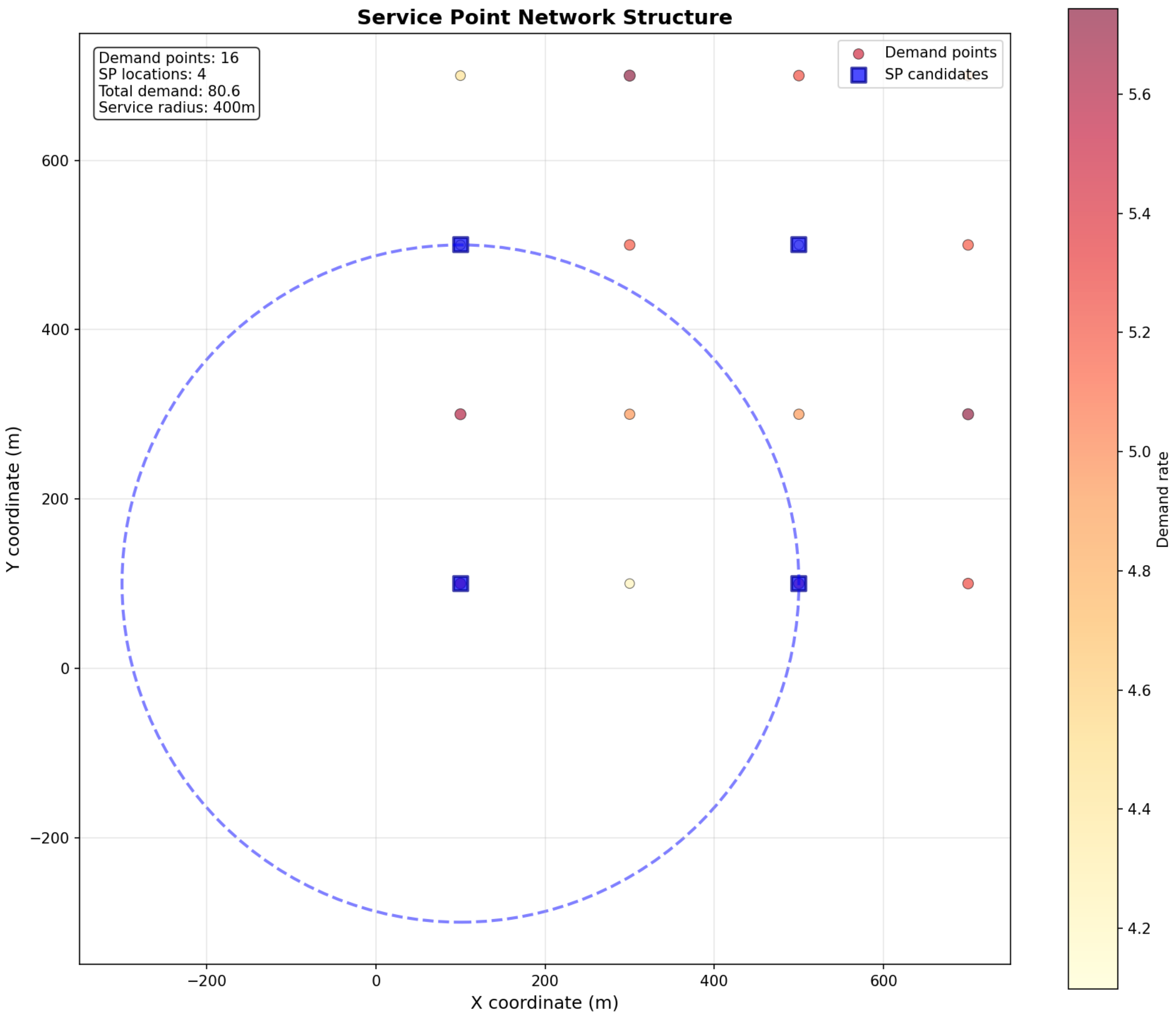
↓

Result: Tractable MILP
Transform into deterministic optimization
Linear constraints and objectives

ASSUMPTIONS

ASSUMPTIONS

- City divided into grid cells (200×200m)
- Location & capacity fixed BEFORE operations
- Each (location, size) pair has unique cost
- Each point = cluster of customers (e.g., 500 families)
- Every demand point must have SP within radius r
- Daily arrivals ~ Poisson(λ)
- Parcels $\rightarrow$ nearest open SP (no choice)
- Pickup time ~ Geometric(p)
- When full: Rejection (home delivery) OR Postponement (wait for space)


Service Point Network Structure
Demand points: 16
SP locations: 4
Total demand: 80.6
Service radius: 400m
Demand points
SP candidates
Y coordinate (m)
X coordinate (m)
Demand rate
600
400
200
0
−200
0
200
400
600
5.6
5.4
5.2
5.0
4.8
4.6
4.4
4.2

UNCERTAINTY

- Parcel arrivals at each Service Point: RANDOM
- Customer pickups between cycles: RANDOM

→ SP can be seen as a queue with

limited capacity:

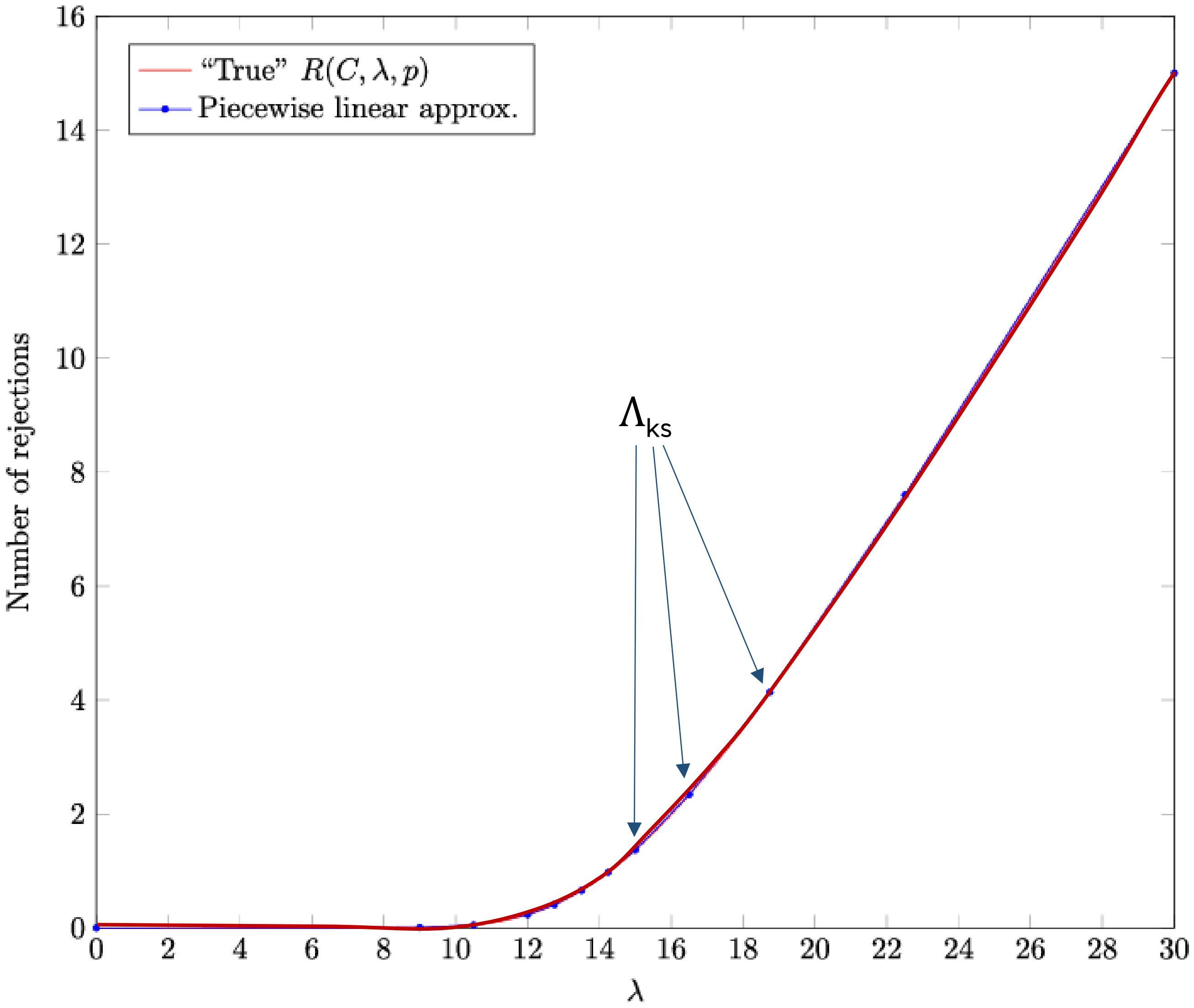
parcels will arrive (supply), be collected (pickup), but if the queue is full, we need to handle overflow.

HANDLE OVERFLOW

REJECTION $R(C,\lambda,p)$

- Use a Discrete-Time Markov Chain (**DTMC**)
- State = number of parcels in the locker at time t
- At each cycle:
  - Parcels arrive ~ **Poisson**(λ)
  - Some are picked up by customers ~ **Geometric**(p)
  - If capacity exceeded → **rejection**

PWL APPROXIMATION WITH NON-UNIFORM BREAKPOINTS



Key properties:

Convex & monotonically increasing

PARAMETERS

PARAMETERS

Geographic Parameters

- F = Set of candidate SP locations
- D = Set of demand points
- t_{df} = Distance/time matrix between each (d,f) pair
- r = Maximum service radius

Capacity Parameters

- $C_1, C_2, \ldots, C_{\bar{s}}$ = Available sizes (30, 60, 90 compartments)
- Λ_{ks} = Discretization points for PWL approximation

PARAMETERS

Demand Parameters

- μ_d = Average daily demand at point d ~ Poisson(μ_d)
- p = Daily pickup probability ~ Geometric(p)

Economic Parameters

- h_{fs} = Total setup cost for size s at location f
- α = Cost per rejection

DECISION VARIABLES

$y_{fs} \in \{0,1\}$ = Open SP of size s at location f?

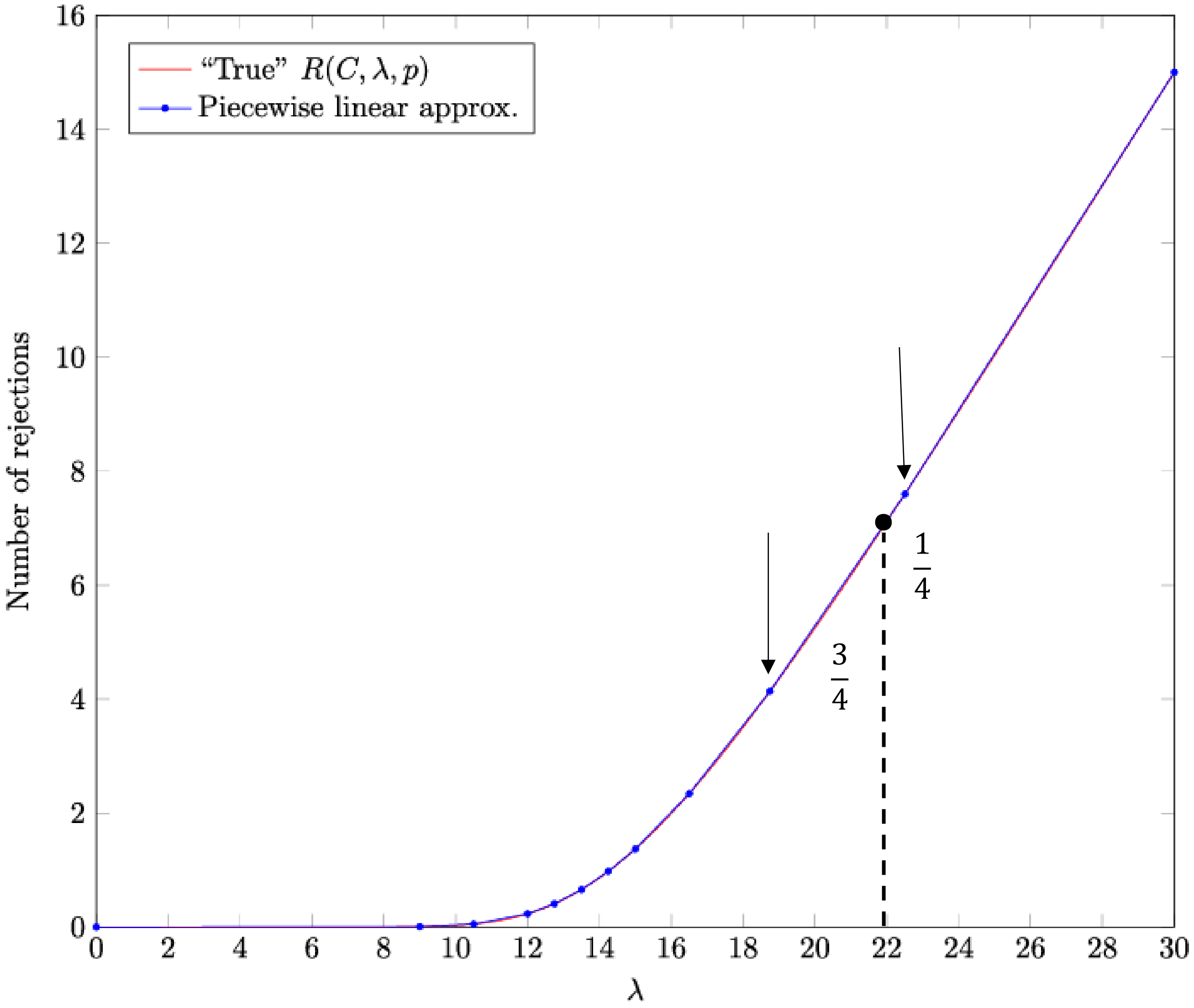
- Binary decision for each (location, capacity) pair

$x_{df} \geq 0$ = Daily flow from demand point d to SP at f

- Continuous variable (parcels/day)

$z_{fks} \geq 0$ = PWL interpolation weights

PWL APPROXIMATION WITH NON-UNIFORM BREAKPOINTS



λ = 22

- Between 19 and 23
- Sets z_{19} = 0.25 and z_{23} = 0.75 (all other z = 0)
- Estimates rejections: 0.25×R(19) + 0.75×R(23)

POSTPONEMENT $D(C,\lambda,\eta)$

- Postponed delivery → parcels accumulate in queue
- Modeled as **M/G/k** queueing system:
  - **M**: Markovian arrivals with rate λ
  - **G**: General pickup time distribution η
  - **k**: Number of compartments C
- $\boldsymbol{\eta}$ captures the full pickup behavior (not just daily probability p)

If arrival rate > service rate → queue explodes!

OBJECTIVE FUNCTION & CONSTRAINTS

THE OBJECTIVE FUNCTION – REJECTION

$$\boxed{\min \sum_{s=1}^{\bar{s}} \sum_{f \in F} \left[h_{fs} y_{fs} + \alpha \sum_{k=1}^{\bar{k}} z_{fks} R(C_s, \Lambda_{ks}, \eta) \right]}$$

CONSTRAINTS

$$\sum_{s=1}^{\bar{s}} y_{fs} \leq 1 \qquad \forall f \in F \tag{1}$$

$$\sum_{f \in F : t_{df} < r} x_{df} = \mu_d \qquad \forall d \in D \tag{2}$$

$$\sum_{f' \in F : t_{df'} > t_{df}} x_{df'} \leq \mu_d \left(1 - \sum_{s=1}^{\bar{s}} y_{fs} \right) \quad \forall d \in D, f \in F : t_{df} < r \tag{3}$$

$$\sum_{k=1, s=1}^{\bar{k}, \bar{s}} z_{fks} \Lambda_{ks} = \sum_{d \in D : t_{df} < r} x_{df} \qquad \forall f \in F \tag{4}$$

$$\sum_{k=1}^{\bar{k}} z_{fks} = y_{fs} \qquad \forall f \in F, s = 1, \ldots, \bar{s} \tag{5}$$

CONSTRAINT

$$\sum_{k=1}^{\bar{k}} z_{fks} = y_{fs} \qquad \forall f \in F, s = 1, \ldots, \bar{s}$$

$$\min f(z) = \sum z_k \times R(C(\Lambda_k, P))$$

$$g_1(z) = \sum z_k \times \Lambda_k - \lambda_{\text{target}} = 0$$

$$g_2(z) = \sum z_k - 1 = 0$$

$$z_k \geq 0 \quad \forall k$$

Carathéodory's theorem—If $x \in \mathrm{Cone}(S) \subset \mathbb{R}^d$, then x is the nonnegative sum of at most d points of S.

If $x \in \mathrm{Conv}(S) \subset \mathbb{R}^d$, then x is the convex sum of at most $d+1$ points of S.

Lemma—If $q_1, \ldots, q_N \in \mathbb{R}^d$ then $\forall x \in \mathrm{Cone}(\{q_1, \ldots, q_N\})$, there exist $w_1, \ldots, w_N \geq 0$ such that $x = \sum_n w_n q_n$, and at most d of them are nonzero.

CONSTRAINT

$$\sum_{k=1}^{\bar{k}} z_{fks} = y_{fs} \qquad \forall f \in F, s = 1, \ldots, \bar{s}$$

KKT CONDITIONS

The frontier is the convex hull of discrete points (Λ_k, R_k)
Convex combination of two points lies on the frontier
The optimum can only combine adjacent points

Stazionarietà

$$\nabla f(z) - \mu \cdot \nabla\left(\sum_k z_k - 1\right) - \nu \cdot \nabla\left(\sum_k z_k \Lambda_k - \lambda_f\right) - u = 0$$

, per ogni k:

$$R_k - \mu - \nu\Lambda_k - u_k = 0$$

$z_k > 0, \quad u_k = 0$ **Convexity of R(Λ)**

$R = \mu + \nu\Lambda$ **Monotonicity of R(Λ)**

Ammissibilità primal

$$z_k \geq 0, \quad \sum_k z_k = 1, \quad \sum_k z_k \Lambda_k = \lambda_f$$

Λ increases, R also increases.
The frontier is ordered with no crossings or oscillations.
Intermediate value λ_f lies within a segment [Λ_i, Λ_{i+1}]
→ identification two consecutive points that bound λ_f.

Ammissibilità dual

$$u_k \geq 0 \quad \forall k$$

Complementary slackness

$$z_k \cdot u_k = 0 \quad \forall k$$

THE OBJECTIVE FUNCTION – POSTPONEMENT

$$\min \sum_{s=1}^{\bar{s}} \sum_{f \in F} \left[h_{fs} y_{fs} + \alpha \sum_{k=1}^{\bar{k}} z_{fks} D(C_s, \Lambda_{ks}, \eta) \right]$$

CONSTRAINTS

$$\sum_{d \in D : t_{df} < r} x_{df} \leq \frac{1-\epsilon}{E(\eta)} \sum_{s=1}^{\bar{s}} y_{fs} C_s \qquad \forall f \in F.$$

Service rate

- On average, a locker is freed once every E(η) periods.
- So the effective service rate per locker = 1/E(η)

Slack factor 1−ε

- To guarantee stability (it can grow unbounded), we require **utilization < 1**.
- Multiplying by 1−ε forces the system to run slightly below full capacity.

CONSTRAINTS

$$x_{df} \geq 0 \qquad \forall f \in F, d \in D : t_{df} < r \tag{1}$$

$$y_{fs} \in \{0, 1\} \qquad \forall f \in F, s = 1, \ldots, \bar{s} \tag{2}$$

$$z_{fsk} \geq 0 \qquad \forall f \in F, s = 1, \ldots, \bar{s}, k = 1, \ldots, \bar{k} \tag{3}$$

ALTERNATIVE MODELS

DETERMINISTIC MODEL

- No randomness considered

- Add safety margin β (arbitrary)
- R(overflow) = max(0, demand - capacity)
- New variable: q_f = deterministic rejections

Why it Fails:

- Underestimates rejections by 70-80%
- Pushes utilization to 95% (dangerous zone)

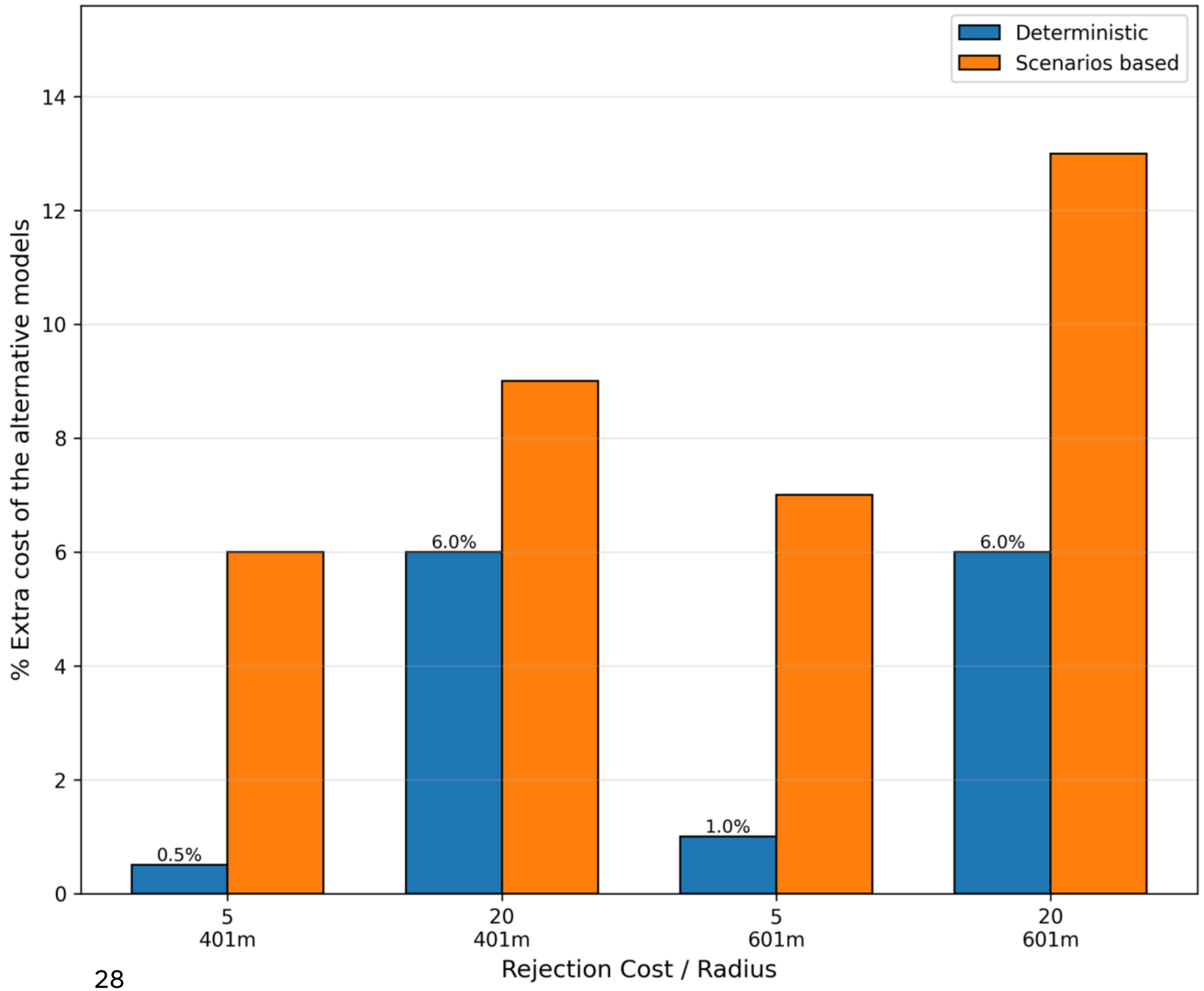
SCENARIO-BASED MODEL

- Generate 50-100 demand scenarios ($\mathcal{K}$)
- μ_{dk} demand at point d in scenario k (vs. single μ_d in PWL)
- π_k the probability of scenario k

Problems:

- Computational explosion
- 50× more variables
- Still not accurate enough
- Can't solve large instances

COMPARISON PWL WITH THE ALTERNATIVE MODELS

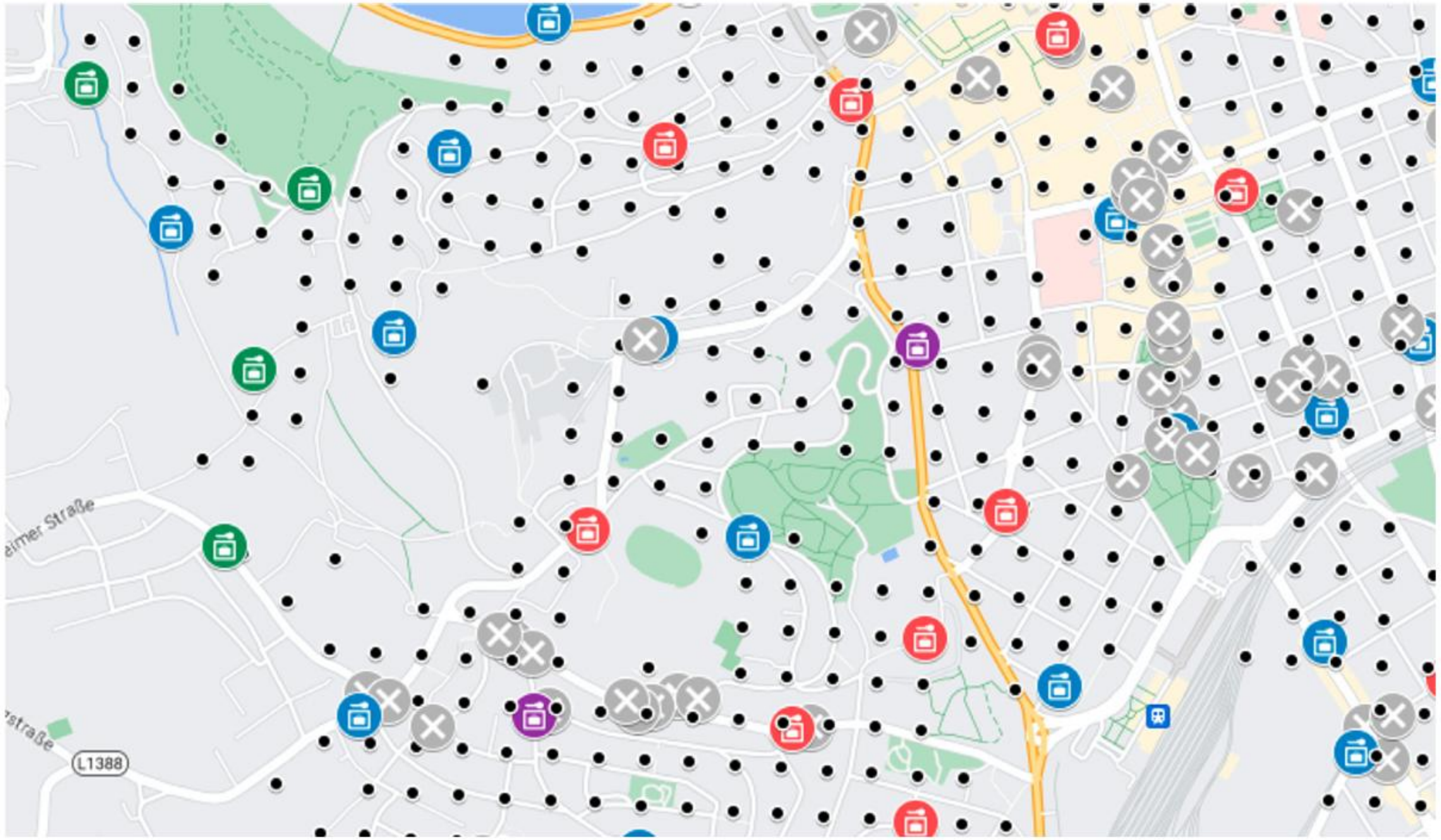


- PWL = baseline
- Bars = how much MORE the other models cost

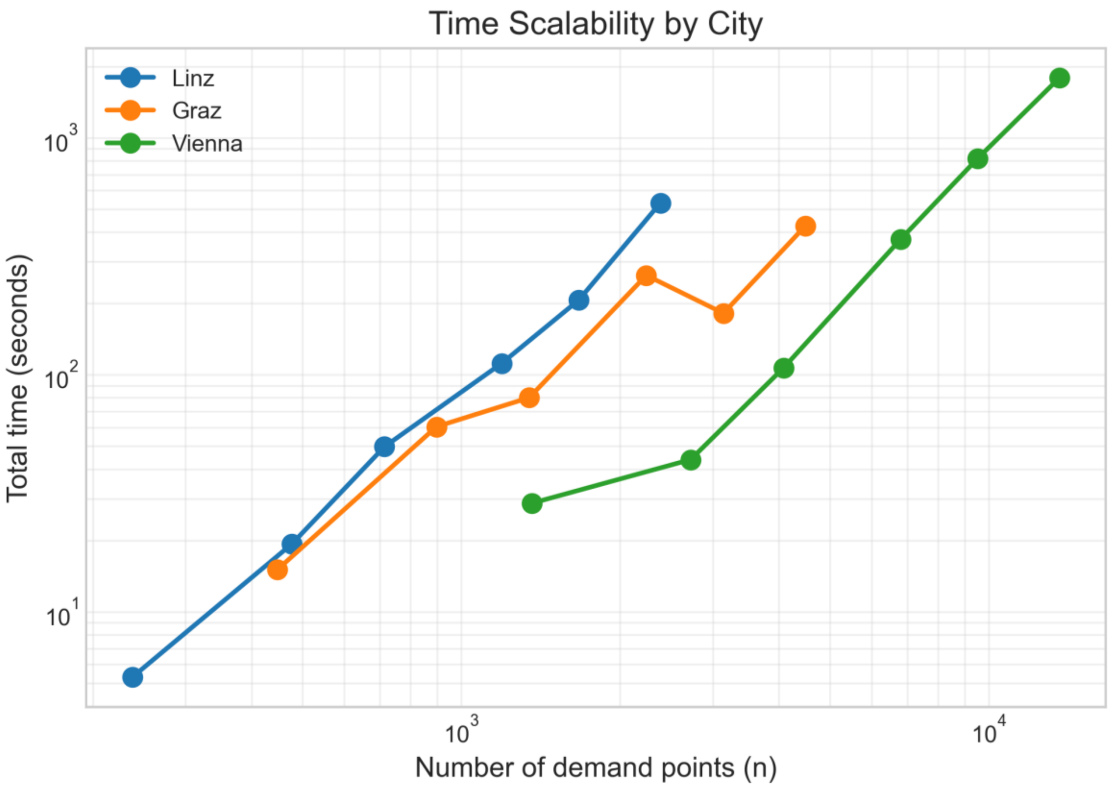
REAL-WORLD TEST: AUSTRIAN CITIES

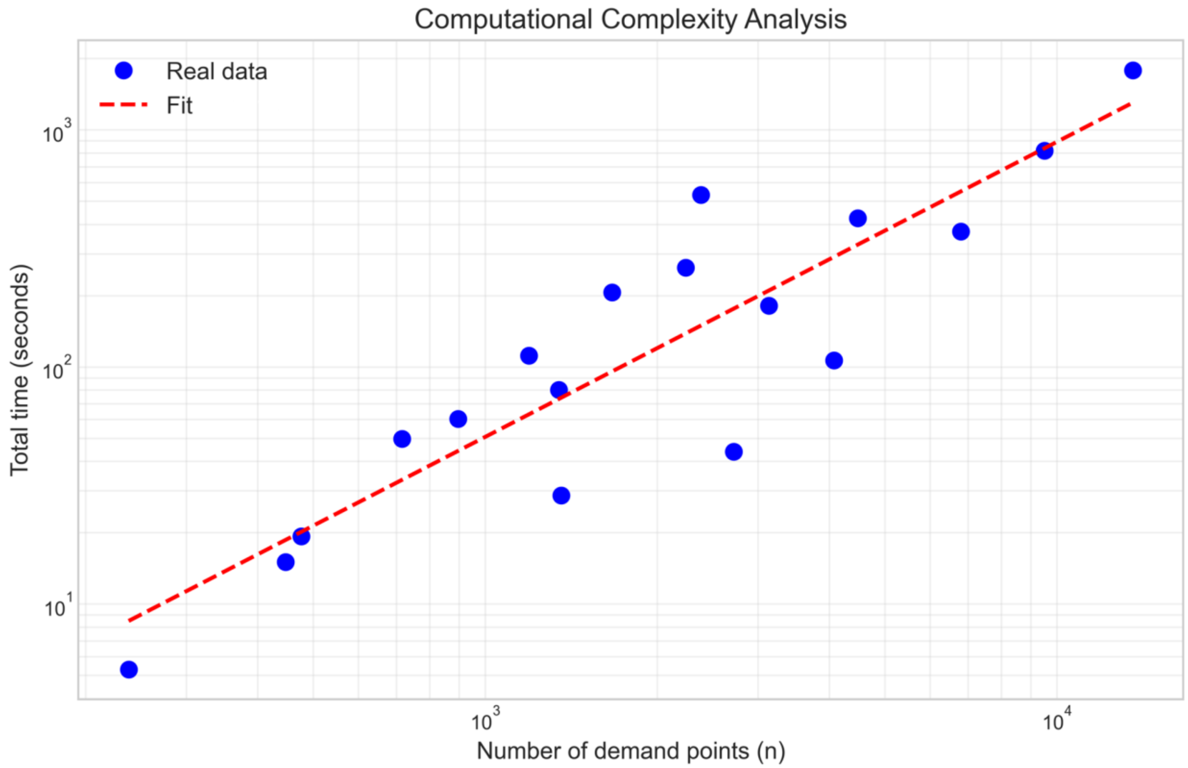
GEOGRAPHICAL REPRESENTATION OF THE SOLUTION FOR LINZ

- green: 30 lockers
- blue: 60 lockers
- red: 100 lockers
- purple: 150 lockers



SCALABILITY & COMPLEXITY





COMPUTATIONAL RESULTS ON REAL URBAN NETWORKS

	Linz	Graz	Vienna
Demand Points	2,385	4,485	13,600
SPs	542	1,111	3,310
Time(minutes)	9	7	30
Total cost(M€)	5.5	4.9	5.6
Population	207,000	291,000	1,900,000

CONCLUSIONS

✓ **Challenge**: Real systems are stochastic → intractable

✓ **Solution**: Pre-compute stochasticity → optimize deterministically

✓ **Result**:

- Maintains accuracy of stochastic models
- Achieves efficiency of deterministic optimization

Outperforms alternatives: when rejection costs are high
Case study (Austrian cities): confirms practical effectiveness

THANK YOU