Optimal pricing in a Single Server Queue

Ashok Krishnan K.S.
Inria,
Sophia Antipolis, France.

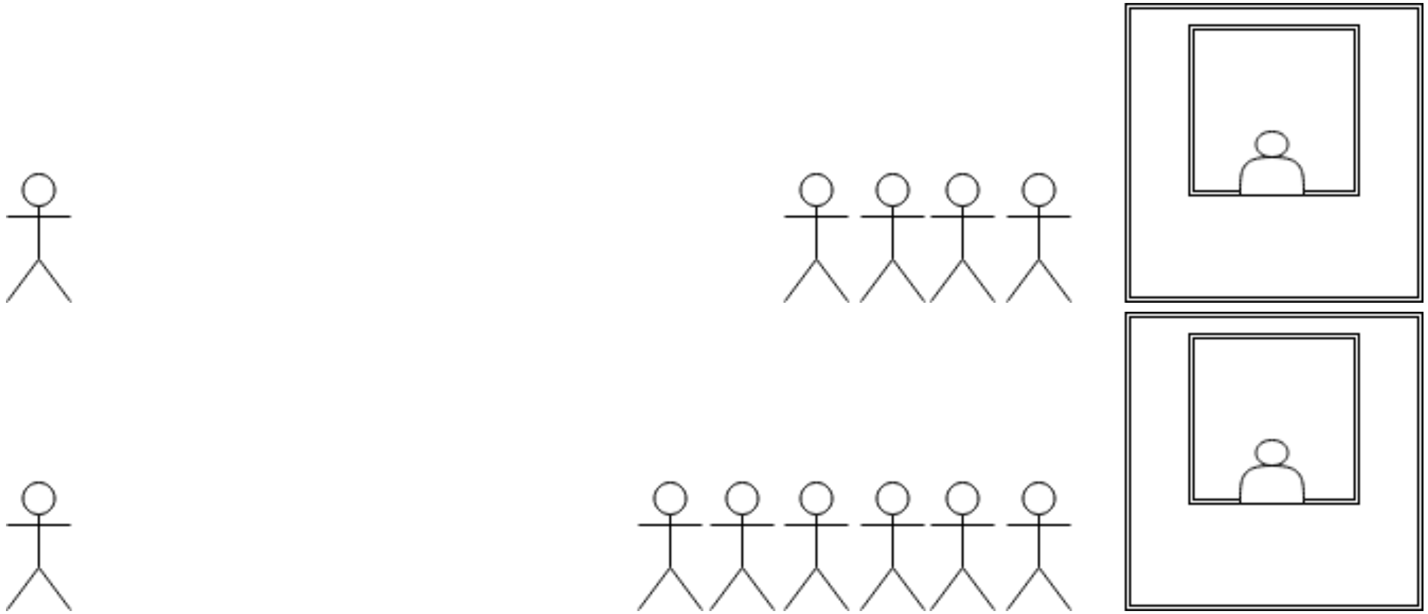
Joint work with
Chandramani Singh[1], Siva Theja Maguluri[2], Parimal Parag[1]

[1] IISc Bangalore [2] Georgia Tech

Source:
Alexandre Duret-Lutz
https://www.flickr.com/photos/gadl/89650415

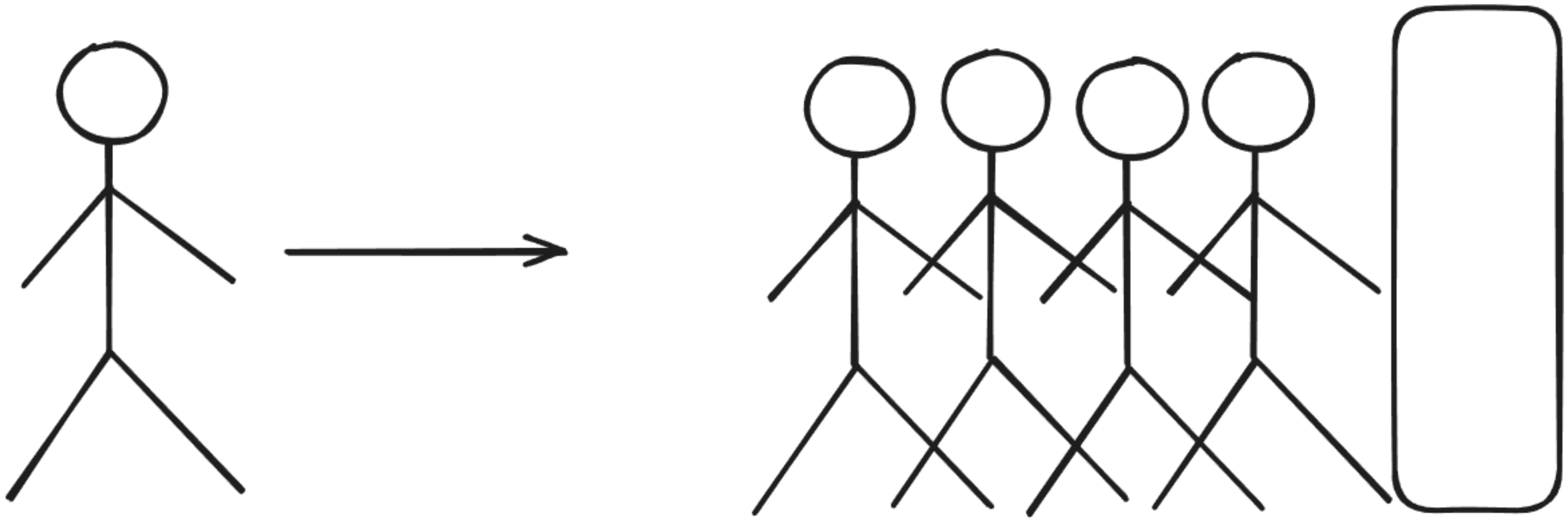
To Join or Not to Join

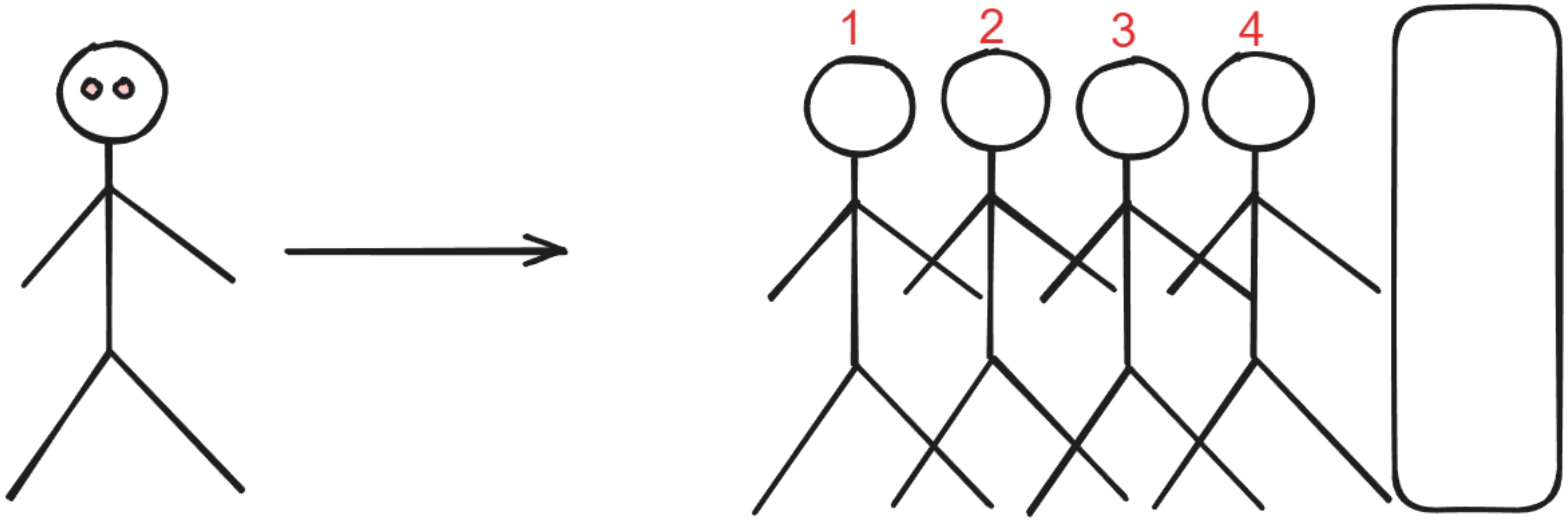
- Example

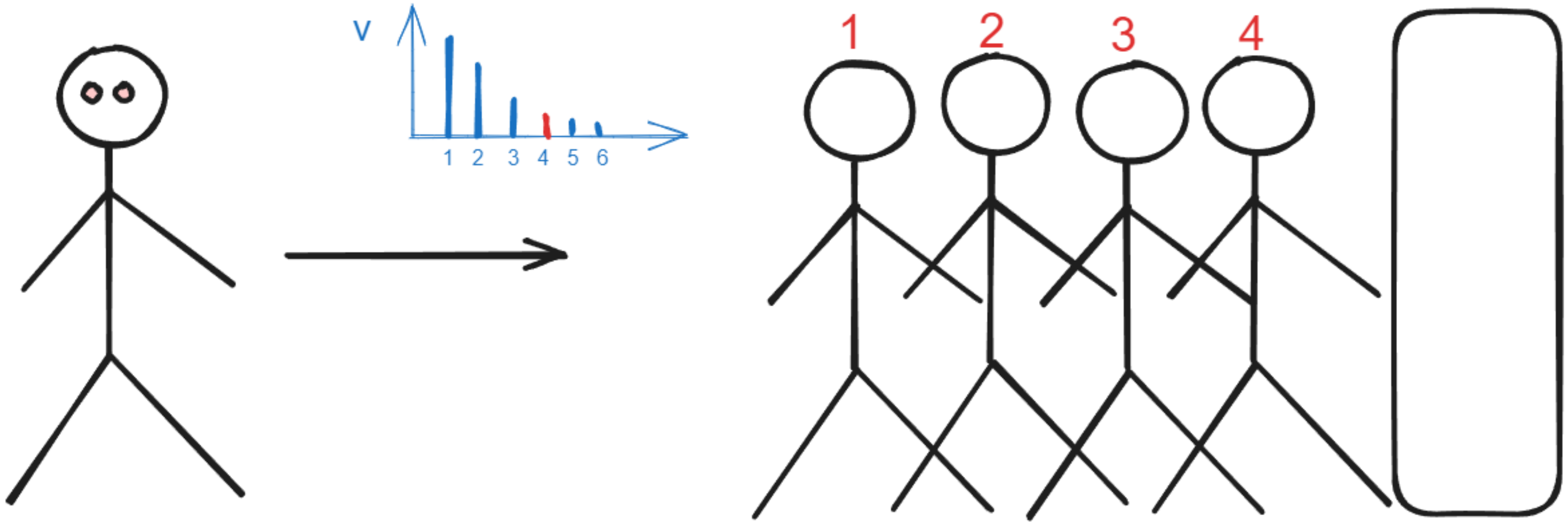


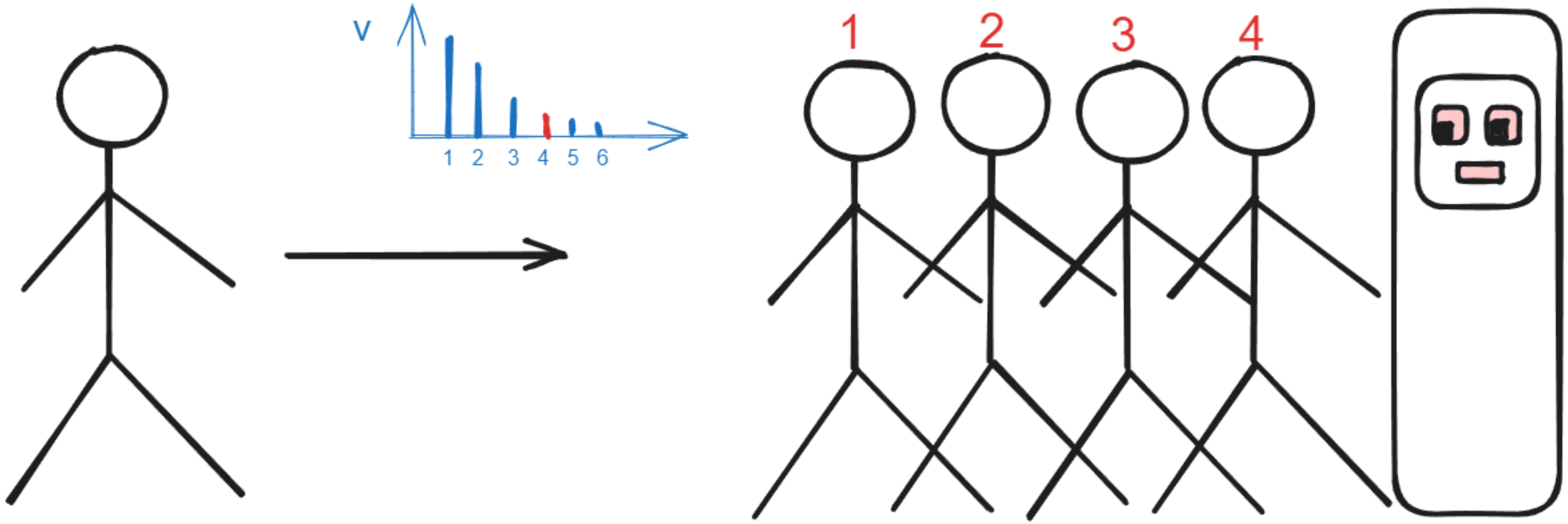
Examples

- Cloud Servers
- Uber
- Movies
- Restaurants
- And more…

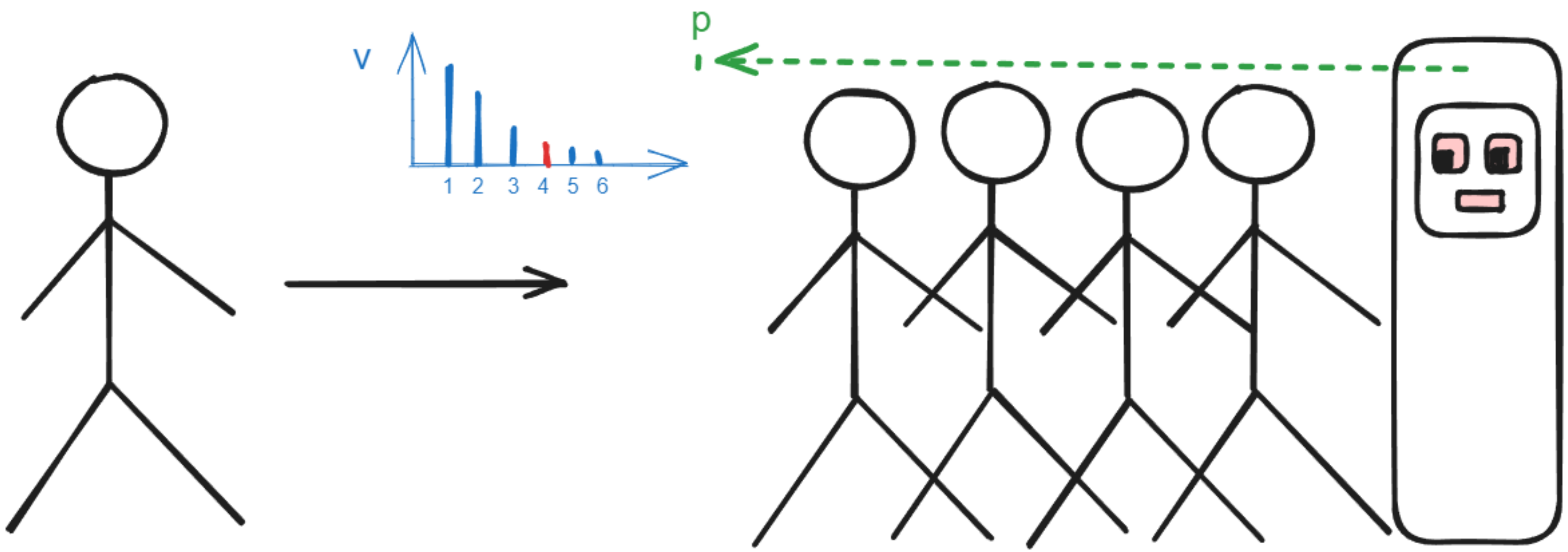



1
2
3
4

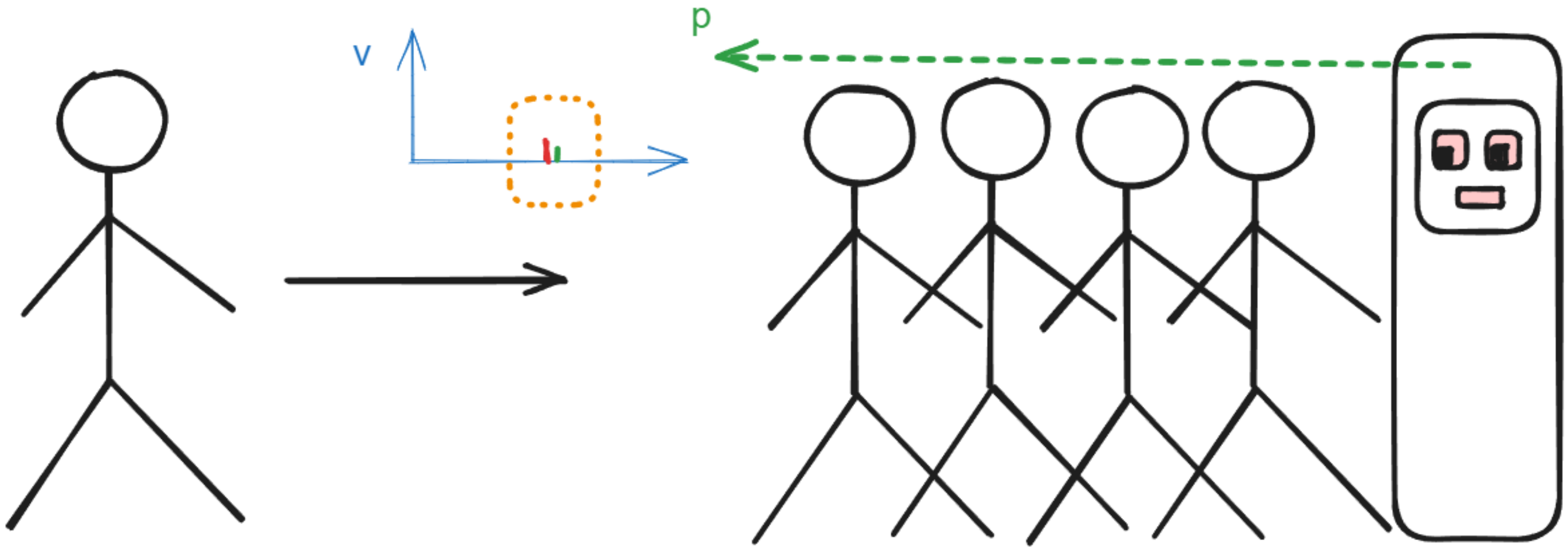

v
1 2 3 4 5 6
1
2
3
4

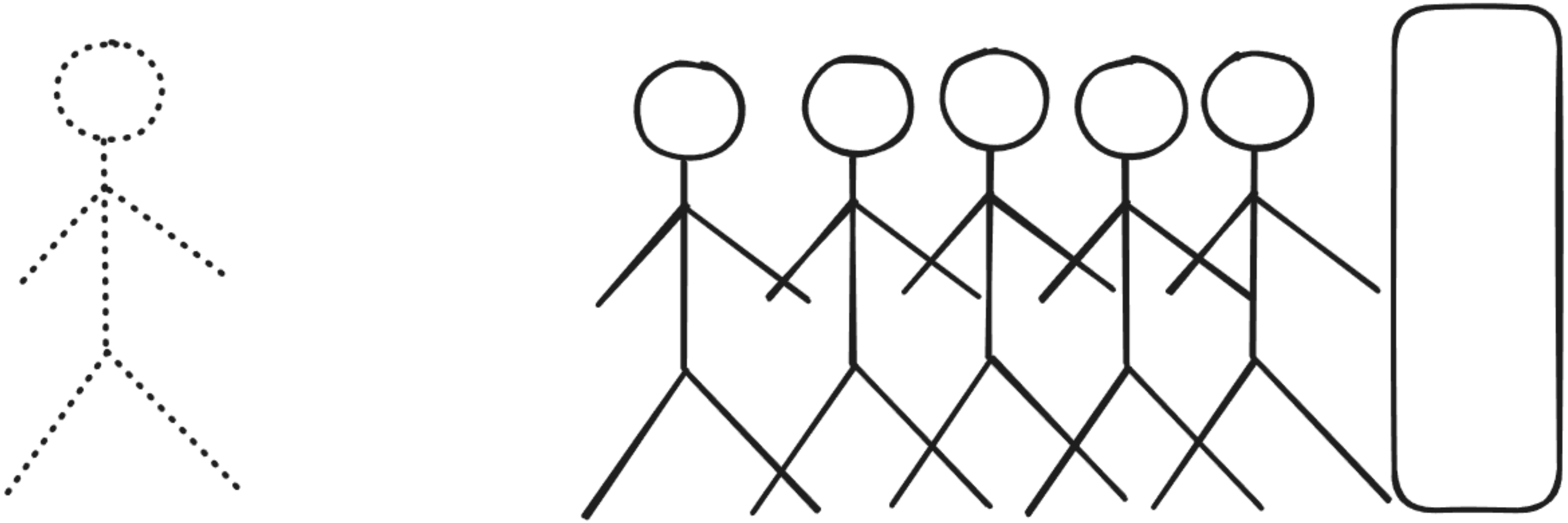


v
1 2 3 4 5 6
1
2
3
4

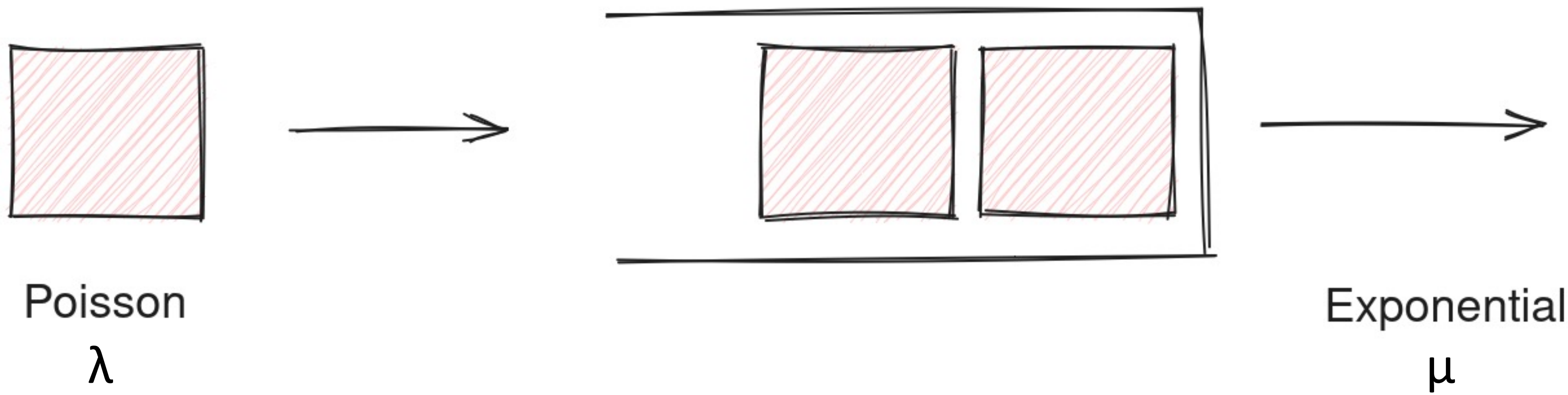


v
1 2 3 4 5 6
p


v
p

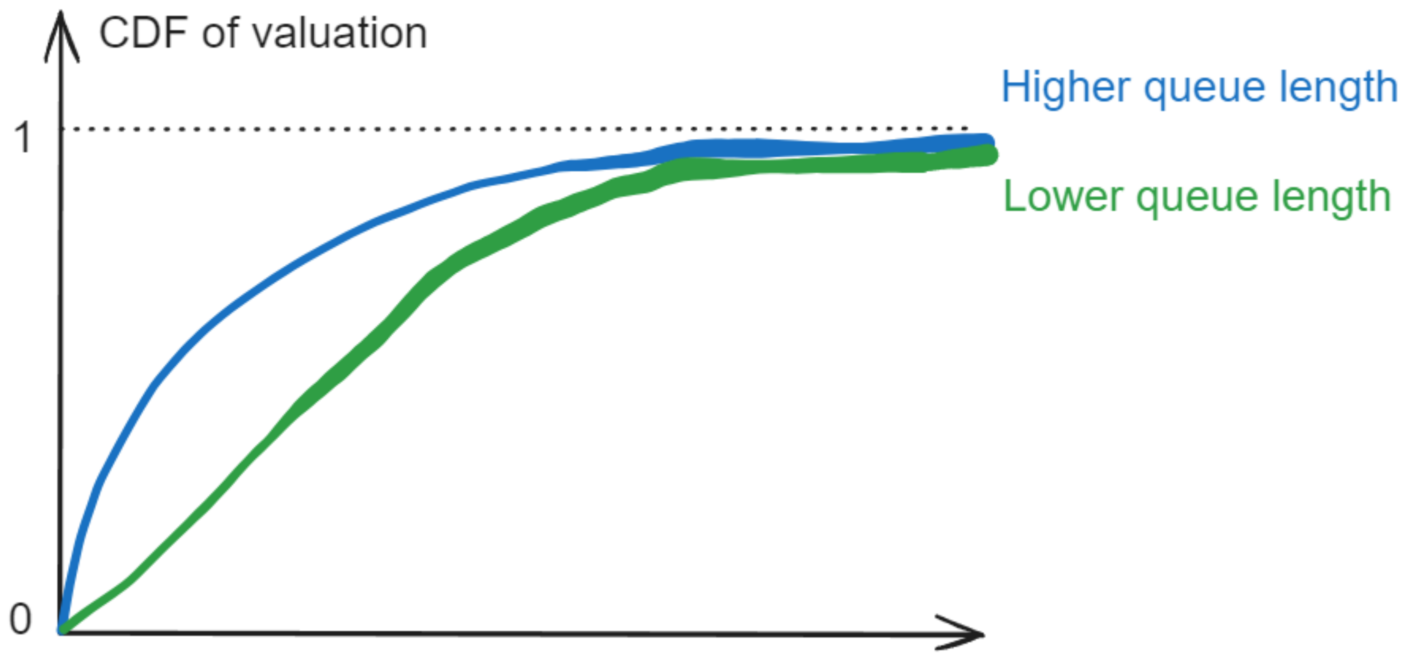


Model

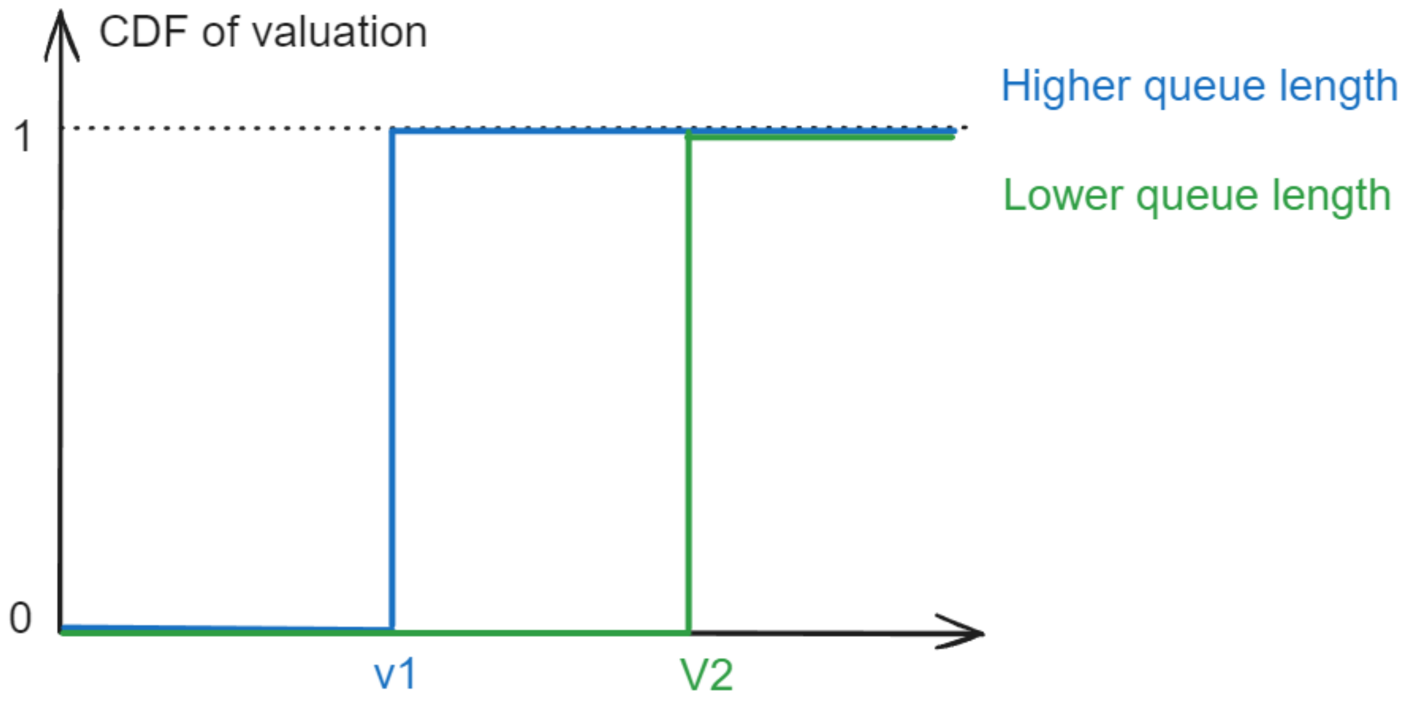


Model

- Random valuation for each user
- Valuation depends on queue length
- Valuation decreases stochastically as queue length increases



CDF of valuation
Higher queue length
1
Lower queue length
0


CDF of valuation
Higher queue length
1
Lower queue length
0
v1
V2

Optimization

- Choose admission prices at each queue length
- Maximize long run average revenue
- A simple solution: Myopic policy

Myopic Policy: For each queue length, charge the price that is optimal for its valuation.

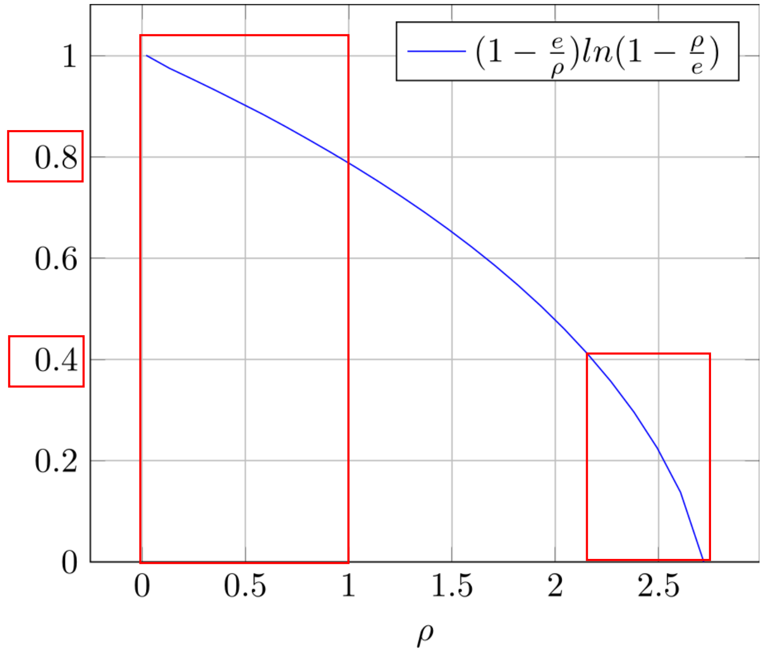
Theorem[KSMP]

The myopic policy is guaranteed to provide at least a fixed fraction of the optimal revenue.

Myopic Policy

- Can perform well when variation of valuation with load is small
- Performs badly at high loads in some cases

Ratio of Myopic revenue To Optimal Revenue for Exponential valuation



Queue without waiting cost

- All packets have same valuation distribution G
- Price p independent of queue length
- Arrival joins if valuation greater than price, i.e, with probability $1 - G(p)$
- Revenue equals $\max_{p} \lambda p(1 - G(p))$

Queue with waiting cost

- Valuation G_i will depend on queue length i
- Modelled as MDP
- Obtain Bellman's equations

$$m_i(\Delta_i) = \frac{\theta - \mu\Delta_{i-1}}{\lambda}, \qquad i = 0,1,2,3,\ldots$$

$$\Delta_{-1} = 0$$

- Congestion loss Δ_i increasing in queue length i
- Optimal prices correspond to $\max_p \lambda(p - \Delta(i))(1 - G(p))$

Solving the equations: Outline

- Truncate to K equations
- Solve to get optimal price vector $u(K)$

Theorem[KSMP]

As K becomes larger, $Revenue\big(u(K)\big) \rightarrow optimal\ Revenue$

Theorem[KSMP]

As K becomes larger, $u(K) \rightarrow optimal\ price\ vector^{*}$

*conditions apply

Solving the equations: Outline

- The K equations yield a scalar fixed point equation in the optimal revenue rate θ

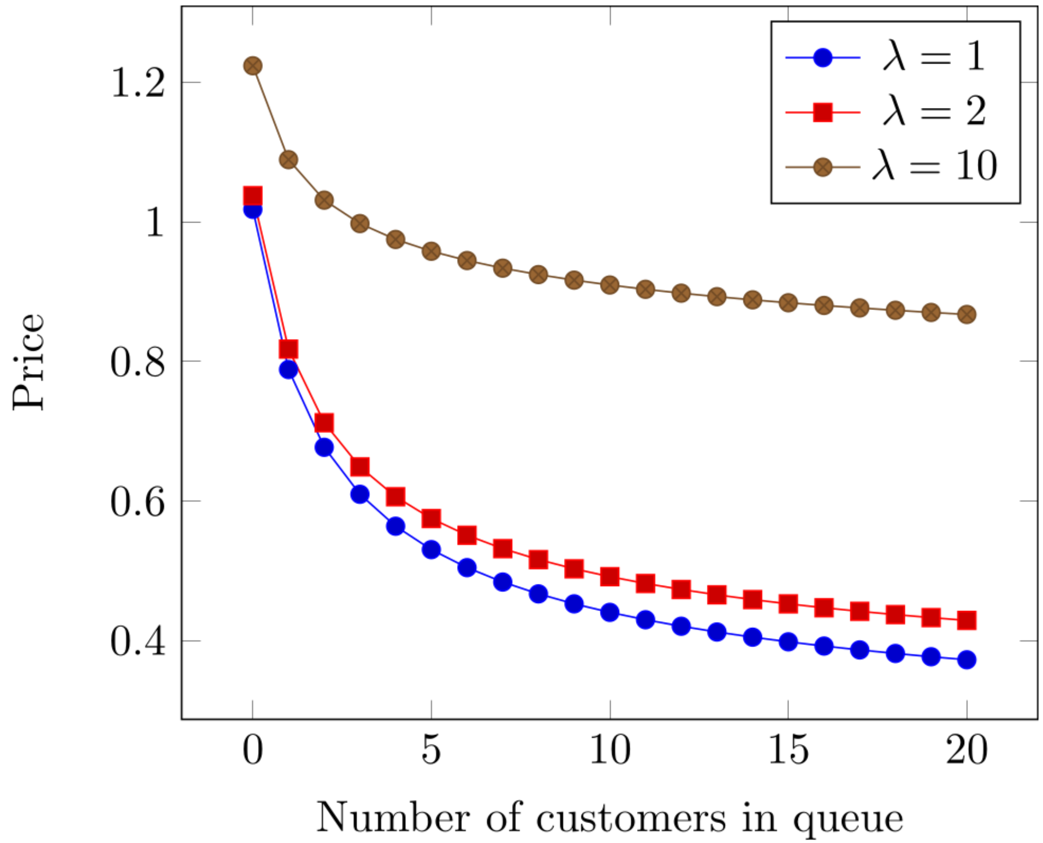
$$\theta = \psi(\theta)$$

- Solve this iteratively and obtain the revenue rate θ
- Plug this back in the equations to obtain the optimal prices
- Iterative algorithm gives optimal revenues and prices

Special Case: Non Random valuations

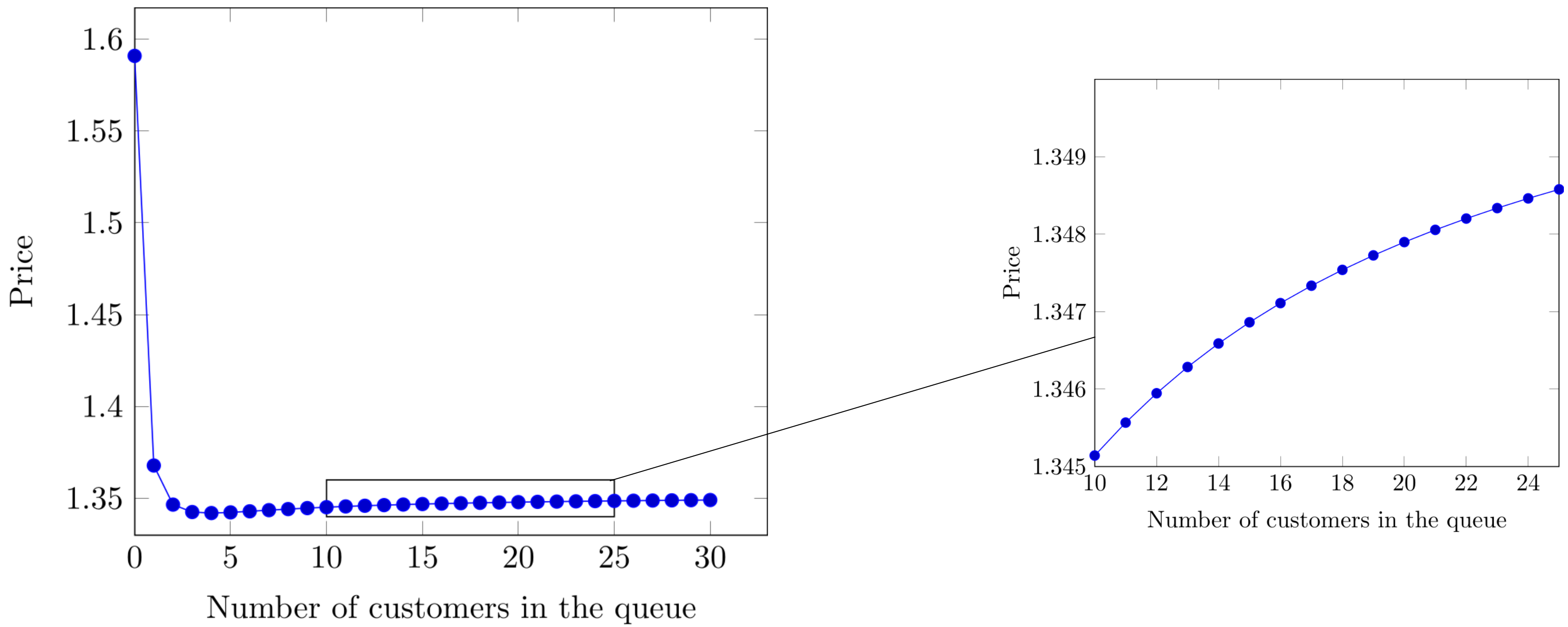
- Valuation is non random: when queue length is i, valuation is v_i
- $v_i \to v$
- Case 1: There some queue length beyond which no one is accepted
    - Finite system of equations
- Case 2: All are accepted always
    - Infinite system of equations

Numerical Results



Valuation $G(u) = 1 - \exp(-\log(i + e)\, u)$

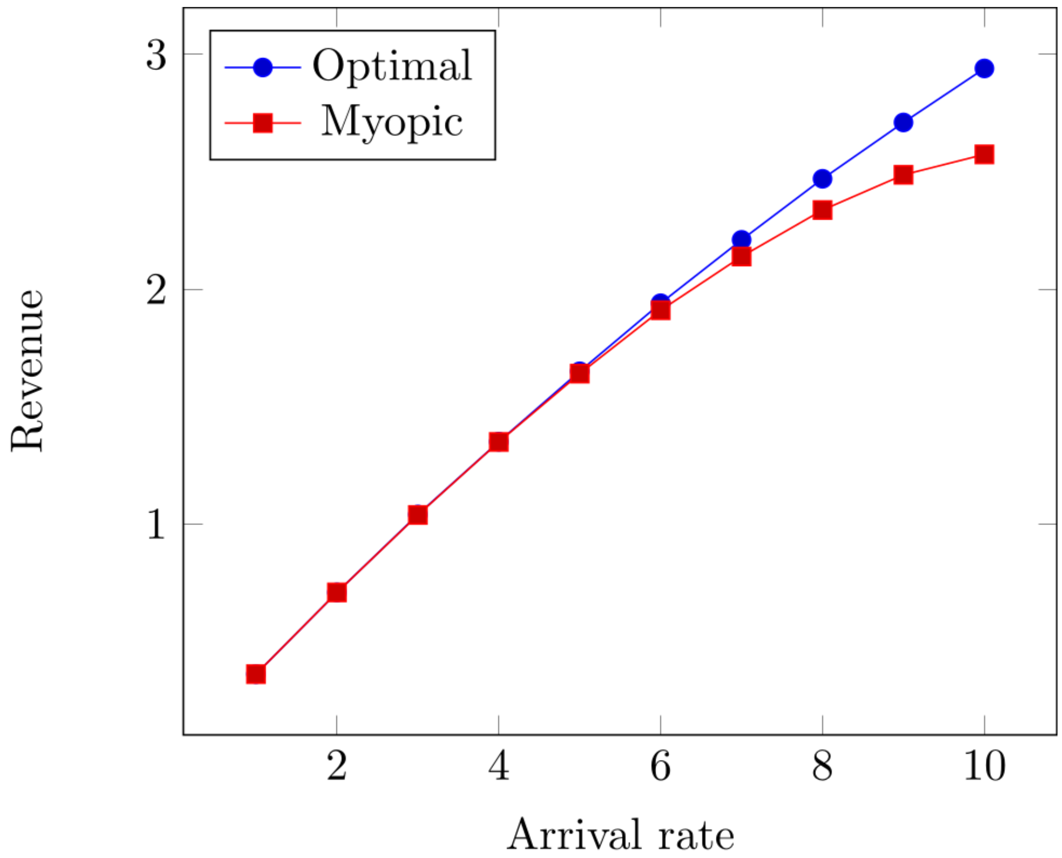
Numerical Results



Not decreasing!

Valuation $G(u) = 1 - \exp(-\left(2 - \frac{1}{1+i}\right) u)$

Numerical Results



Valuation $G(u) = 1 - \exp(-\log(i + e)\,u)$