Exploring Markov models for gate-limited service and their application to network-based services

Glenford Mapp and Dhawal Thakker

Networking Research Group

School of Engineering and Information Sciences (EIS)

Middlesex University, London

Outline of Talk

- Context
- Types of Service
- Exhaustive-Limited Service
- Gated-Limited Service
- An Approximate Solution
- Application to Network-Based Services
- Conclusion and Future Work

Context: Queuing Theory

- Customers being served in some way
  - Teller at bank, roving salesman, etc
- Servers do work on customers after which customers may leave the system or join another queue for further service.
- Servers may serve other queues
- Different ways of serving
  - Exhaustive
  - Gated

Types of Service

- Exhaustive – the server serves until the queue is empty. So it serves customers who arrive after service on the queue has started.
- Gated Service – the server only serves the customers in the queue at the start of service. Customers arriving after service has begun are served on subsequent server visits.

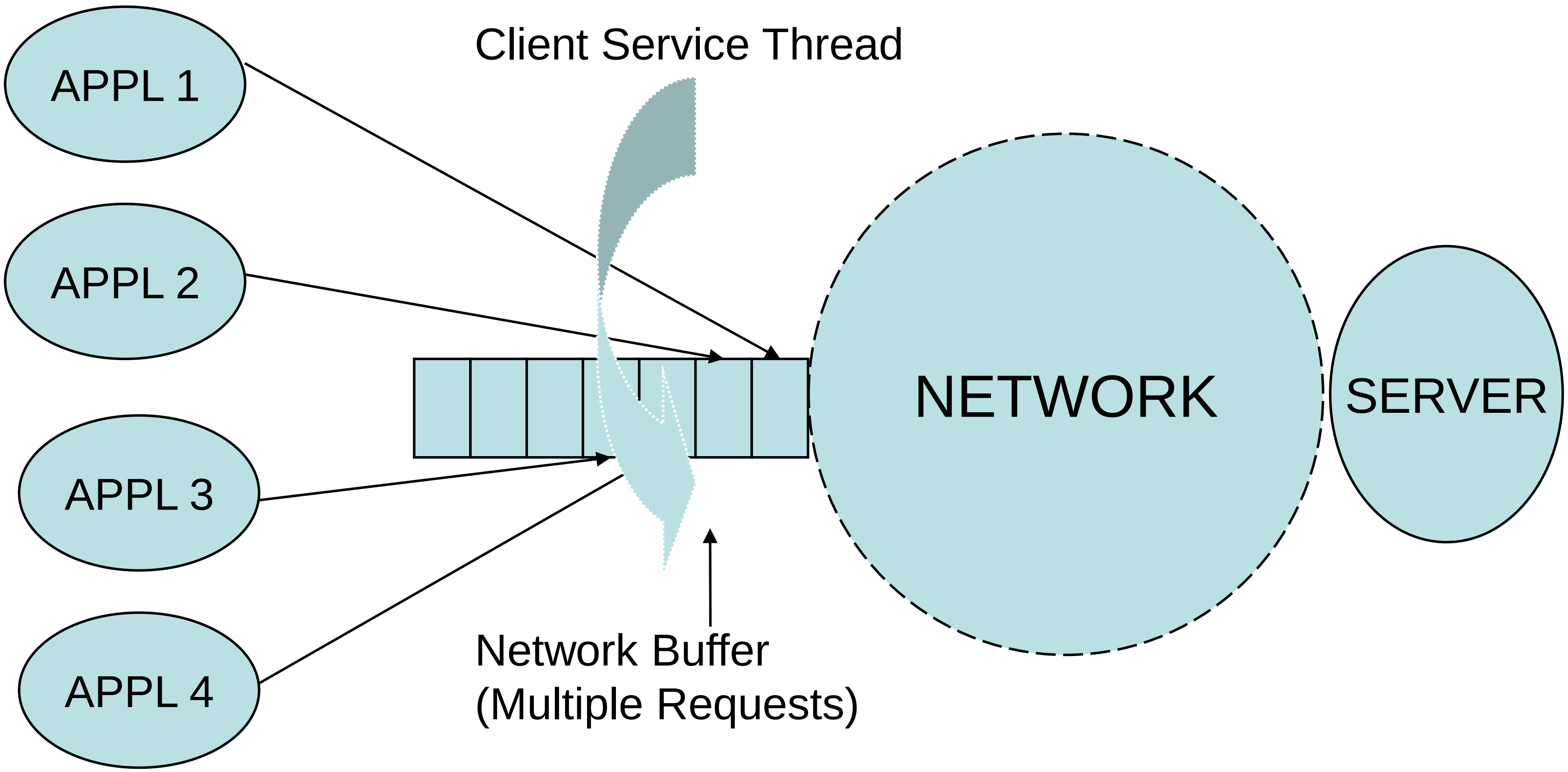
Limited Service Derivatives

- Exhaustive-Limited: The server serves a maximum of k customers. Other customers are serviced on subsequent visits.
- Gated-Limited: The server only serves a maximum of k customers that it finds in the queue when it arrives. Other customers are serviced in the same manner but on subsequent visits

Application

- Different applications may different service models
- We focus on gated-limited systems for a number of reasons
  - Transport:
    - Large systems such as buses can be modelled as gated-limited servers.
  - Network Services can also be modelled as gated-limited service

Simple Example of a Network-Based System



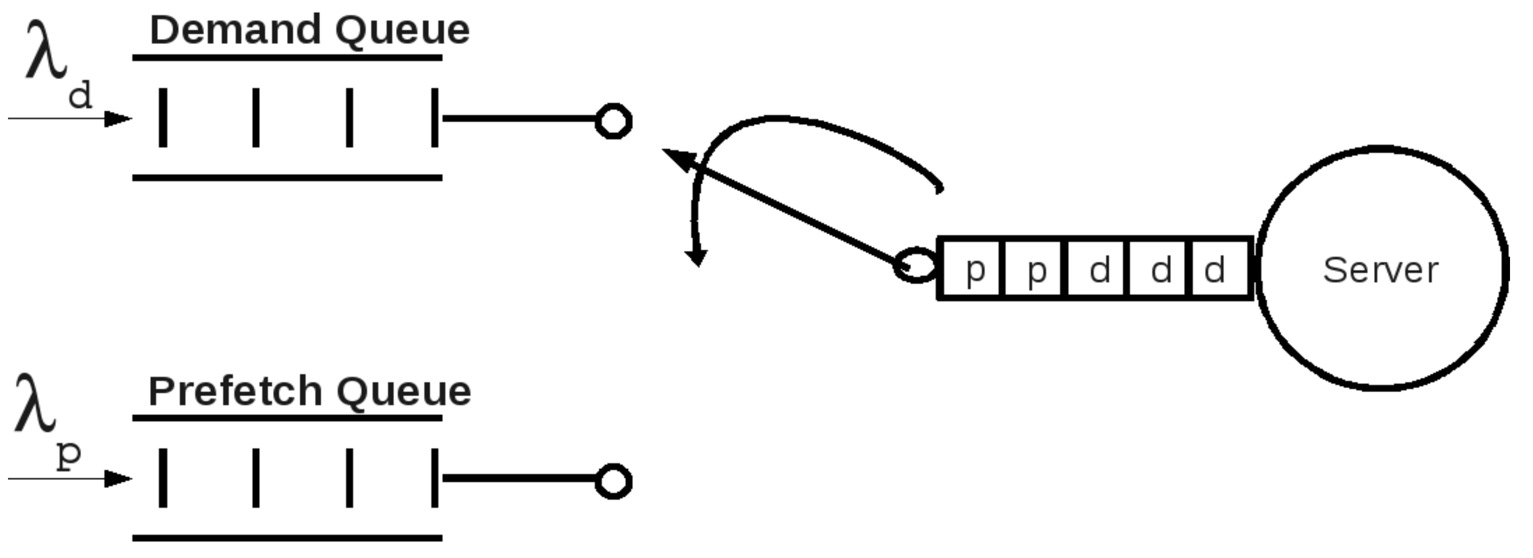
Key Observations

- Once the network buffer is sent off to the server, applications must wait before putting new requests in the buffer
- The buffer is finite, so there is a maximum number of requests, K, that can be serviced at the same time
- Gated-Limited Service

Motivation

- Everything is going to be network-based
  – The network is the computer
- New streaming applications are being used that require better than best-effort service.
    - You-Tube; BBC iPlayer, etc
- Can use pre-fetching techniques
  – but we have to see about demand misses

Network-Based Storage Service to support pre-feteching



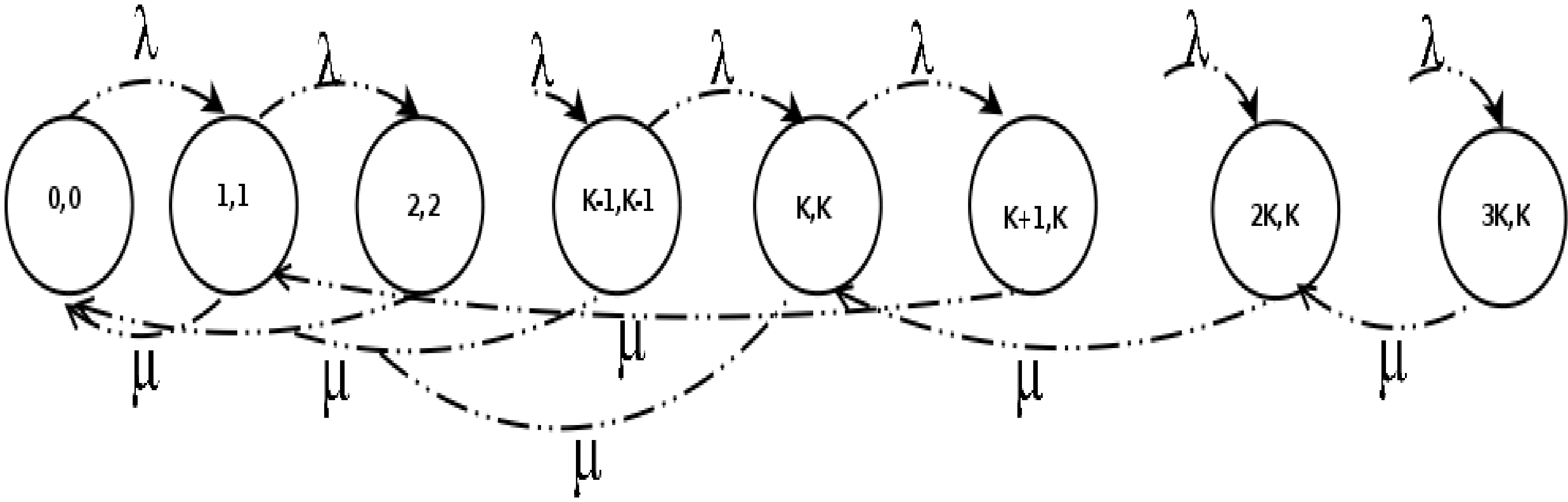
What’s already been done

- If you look at the literature, a lot of work has been done on exhaustive or exhaustive-limited systems
- Gated service also studied
- Gate-limited solutions exist but are generally too hard to calculate
  - Not back-of-the envelope stuff

Standard Solution: The Partial Bulk Service Model (PBM)

- Found in standard text-books:
- Each Markov state is defined by 2 parameters:
    - n = the total number of customers in the system
    - s = the number of customers currently being served
- Note: the maximum number of customer that can be served at the same time is given by K.

PBM Con't



The PBM is exhaustive-limited service, but we need to understand the solution. If K = 1, we have normal MM1. The general solution is quite similar to the MM1 solution

The balance equations for PBM:

$$0 = -(\lambda+\mu)p_n+\mu p_{n+k}+\lambda p_{n-1}$$

$$0 = -\lambda p_0+\mu p_1+\mu p_2+\mu p_{K-1}+\mu p_K$$

For $k = 1$;

$$0 = -(\lambda+\mu)p_n+\mu p_{n+1}+\lambda p_{n-1}$$

$$0 = -\lambda p_0 + \mu p_1$$

This are the basic equations for the

MM1 queue!

So we can say solution for PBM is the same as the MM1 so:

$$p_n = p_0 r^n$$

Where r between 0 and 1. For the MM1 queue; $r = \rho$ The mean queue length (L):

$$L = \frac{r}{1-r}$$

The average waiting time (W):

$$W = \frac{r}{\lambda(1-r)}$$

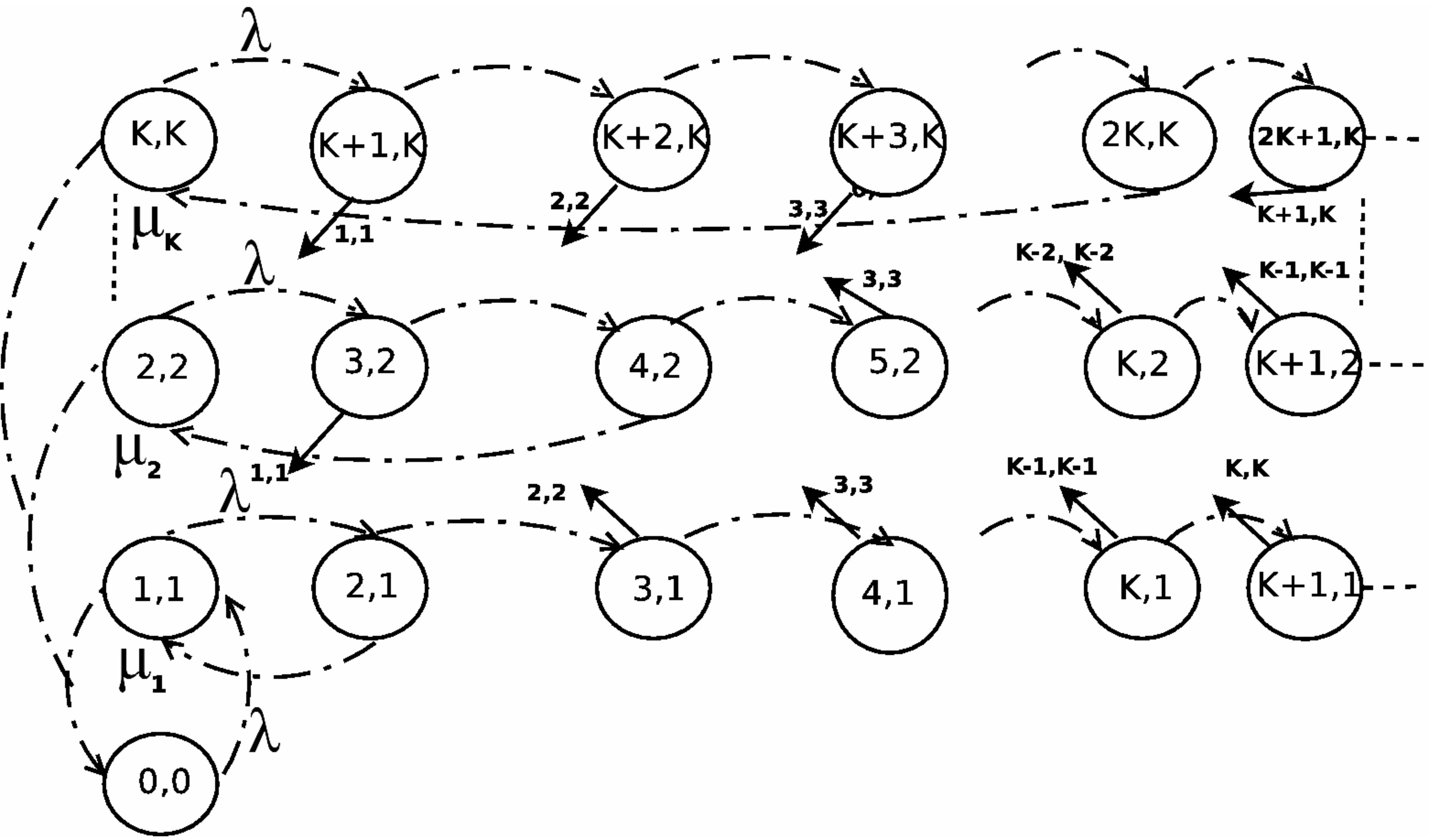
Evaluation of PBM

- Exhaustive-limited not gated-limited
- However, at extremely high loads, the gated-limited waiting times will approach the exhaustive waiting times because there will almost always be K or more customers in the queue.
- We need to remember this!

Our approach

- Use Markov Models
  - Previous approaches have been very mathematical from the word go!
- Look at arrival and departure moments
- For the gated-limited model, the number of customers served in the next cycle is evaluated at the end of the current cycle.

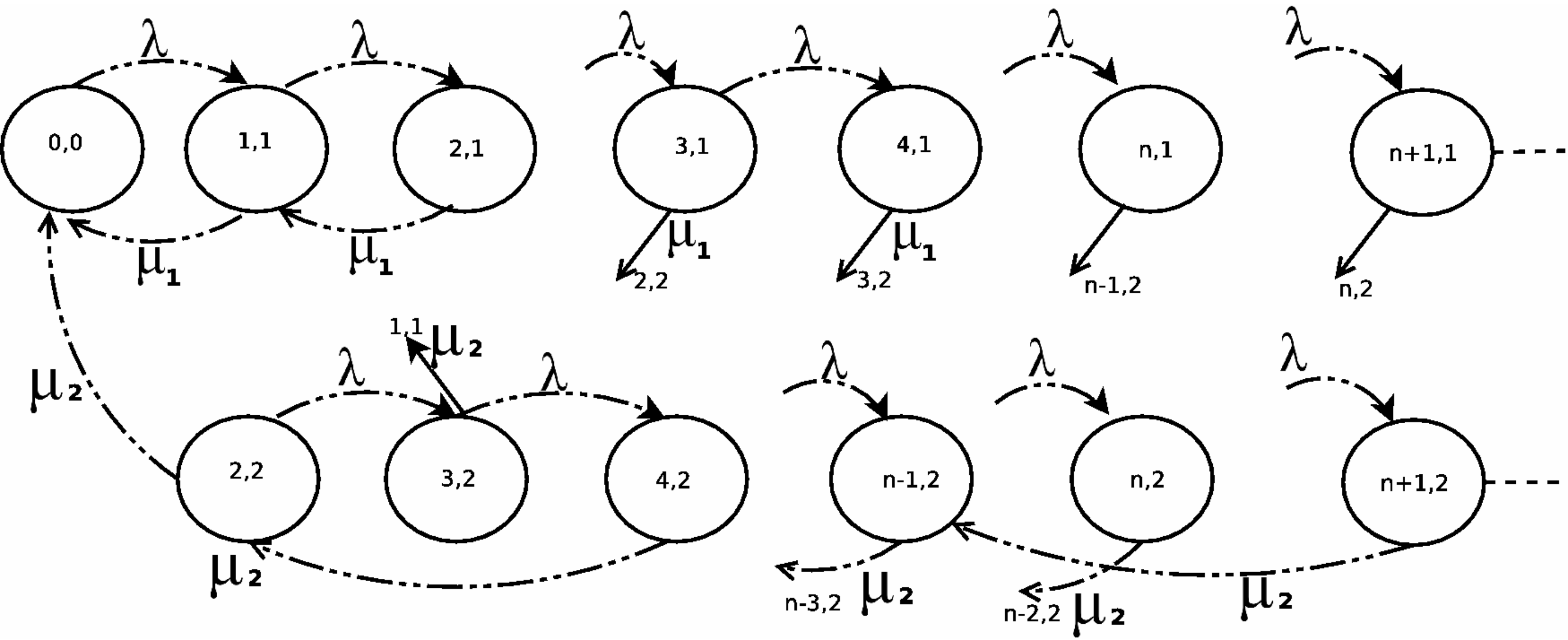
Our Markov Model for Gate-Limited



Model is complicated because..

- There are several chains where each chain represents the number of customers currently being served.
  - So Chain 1, represents 1 customer being served, etc.
- You can jump over several chains in one go depending on how many people are left in your queue at the end of the current service time.

Gated Limited Model for K=2



How do you solve this model?

- There are K chains; so if K is 2; there are 2 chains
- Our approach
    - Try to express the probability of being in each state of each chain in terms of the probabilty of lowest element in the chain. For K = 2; we need to express Chain 1 in terms of P_{11} and P_{22} for Chain 2.
  - Then we concentrate on solving each chain

For CHAIN 1:

$$\lambda p_{n-1,1} = (\lambda + \mu_1) p_{n,1}$$

For any $n > 1$, in Chain 1:

$$p_{n,1} = \frac{\lambda}{\lambda + \mu_1} p_{n-1,1}$$

So we can write:

$$p_{n,1} = \left(\frac{\lambda}{\lambda + \mu_1}\right)^{n-1} p_{1,1}$$

And for $n = s = 1$, we have:

$$(\lambda+\mu_1) p_{1,1} = \lambda p_{0,0} + \mu_1 p_{2,1} + \mu_2 p_{3,2}$$

Finally, for $n = s = 0$:

$$\lambda p_{0,0} = \mu_1 p_{1,1} + \mu_2 p_{2,2}$$

For CHAIN 2:

$$(\lambda+\mu_2)p_{n,2} \quad = \quad \lambda p_{n-1,2}+\mu_2 p_{n+2,2}+\mu_1 p_{n+1,1}$$

And for $n = s$:

$$(\lambda + \mu_2)p_{2,2} \quad = \quad \mu_2 p_{4,2} + \mu_1 p_{3,1}$$

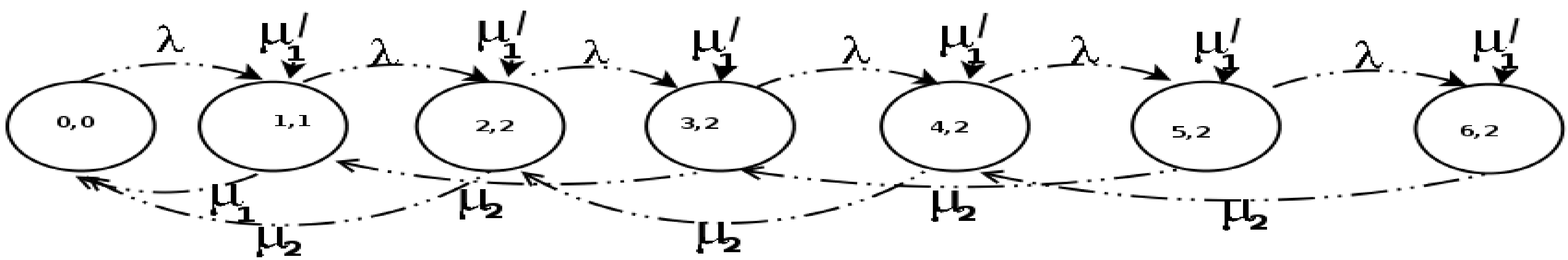
$$(\lambda+\mu_2)p_{3,2} \quad = \quad \lambda p_{2,2}+\mu_2 p_{5,2}+\mu_1 p_{4,1}$$

$$\mu_1 p_{4,1} = (\lambda p_{0,0}-\mu_2 p_{2,2})(\frac{\lambda}{\lambda + \mu_1})^3$$

$$\begin{aligned} 0 \quad = \quad & -(\lambda+\mu_2)p_{3,2}+\lambda p_{2,2}+\mu_2 p_{5,2} \\ & +(\lambda p_{0,0} - \mu_2 p_{2,2})(\frac{\lambda}{\lambda + \mu_1})^3 \end{aligned}$$

The Effect of this Approach

By only expressing Markov state equations for Chain 2 in terms of other Markov states for Chain 2, we are saying that we can imagine that the states for Chain 2 in the gated-limited model to be part of an IMAGINARY PBM chain So we can solve for root r, between 0 and 1, just like in the PBM approach



$$p_{n,2} = p_{2,2} r^{n-2}$$

So we can write: $p_{3,2} = r p_{2,2}$

$$p_{1,1} = C_{1,1}\, p_{2,2}; \qquad p_{0,0} = C_{0,0}\, p_{2,2}$$

$$C_{1,1} = \mu_2 (1 + r)(\frac{\lambda + \mu_1}{\lambda^2})$$

$$C_{0,0} = \frac{\mu_1 C_{1,1} + \mu_2}{\lambda}$$

$$S_1 = \sum_{n=1}^{\infty} p_{n,1}; \quad S_2 = \sum_{n=2}^{\infty} p_{n,2}$$

Then we can say that:

$$p_{0,0} + S_1 + S_2 = 1$$

1

$$p_{2,2} = \frac{1}{C_{0,0} + \frac{\lambda+\mu_1}{\mu_1} C_{1,1} + \frac{1}{1-r}}$$

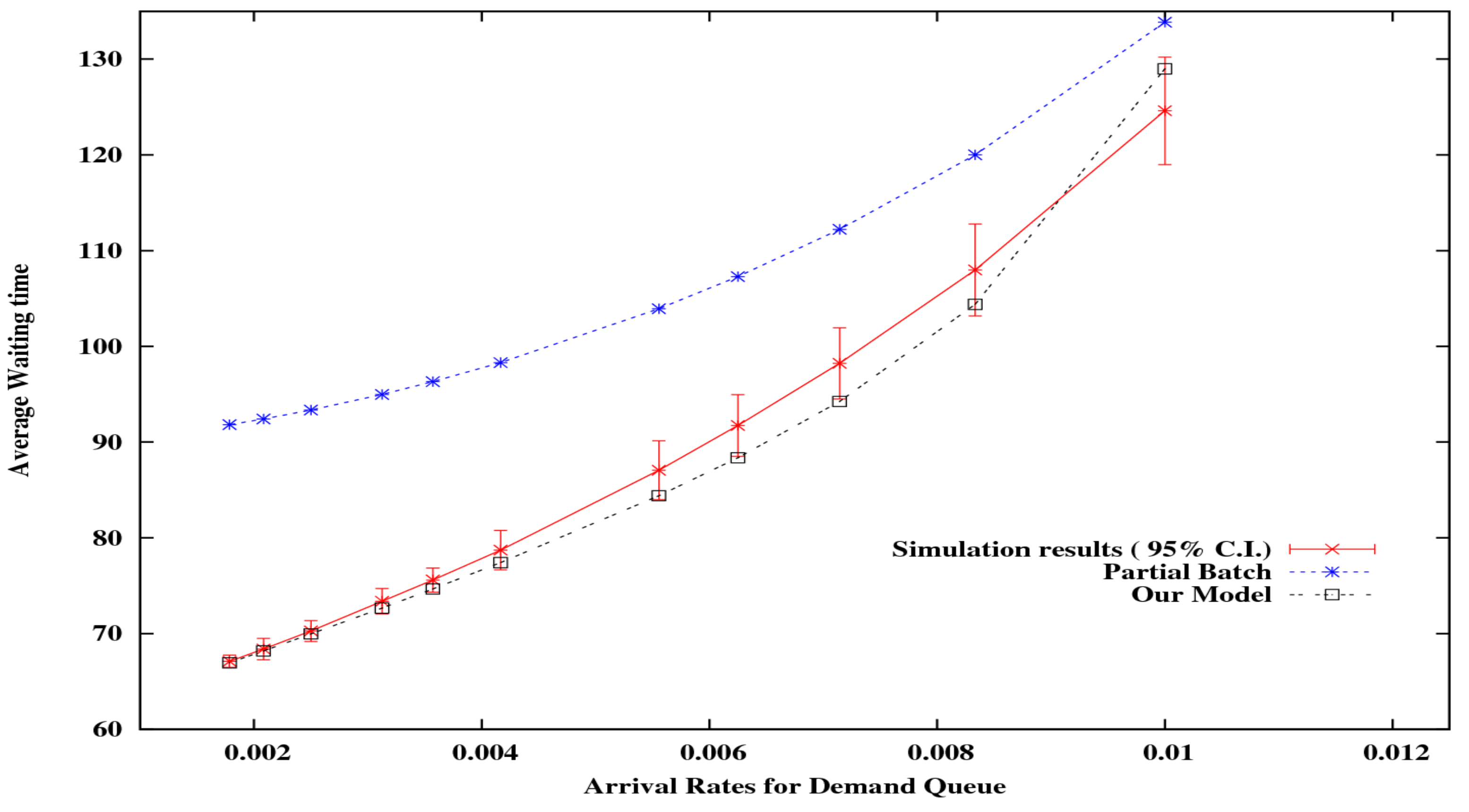
The average queue length, L is:

$$L = \sum_{n=1}^{\infty} n \left(\frac{\lambda}{(\lambda+\mu_1)}\right)^{n-1} p_{1,1} + \sum_{n=2}^{\infty} n r^{n-2} p_{2,2}$$

$$L = \left(\frac{\lambda+\mu_1}{\mu_1}\right)^2 p_{1,1} + \frac{2-r}{(1-r)^2} p_{2,2}$$

Average waiting time: $W_d = \frac{L}{\lambda_d}$

Results for K = 2



Towards a general solution

1

If we express L for each chain in terms of its first element and then sum we get:

$$L = \sum_{n=1}^{k} \frac{n-(n-1)r_n}{(1-r_n)^2} p_{n,n}$$

For $n < k$,

$$r_n = \frac{\lambda}{\lambda + \mu_n}$$

For $n = k$, we use the imaginary PBM technique

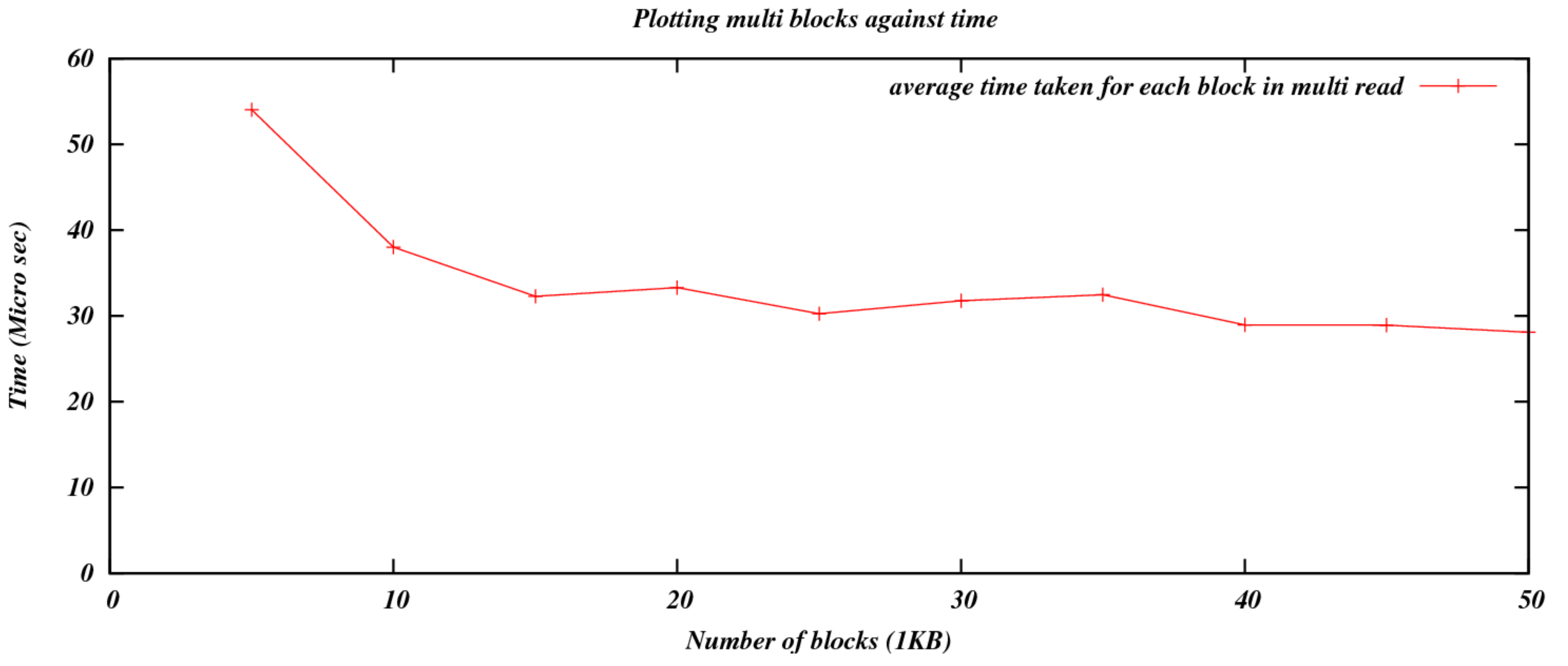
Towards a general solution

- The most difficult part is to calculate $p_{n,n}$
- Use the equations for $p_{n,n}$ and express these variables in terms of $p_{k,k}$.
- Mathematics is complicated
  - Need to look at using matrix techniques to solve these equations.
- Still a lot of work to be done but the approach looks promising.

Application – Global Storage Server Project

- Aim
  - To develop a high performance globally accessible storage server
    - Eliminate dependency on local storage
      - More green: less power, noise
  - Network Memory Server (NMS)
    - Uses the memory of another machine
    - Appears as a hard-disk to the client OS

Clustering in the NMS



- Time to Fetch p blocks = L + Cp

Approach

- To allow stream to run without jitter
  - Time to fetch < Time to process
    - $L + Cp < Tcpu * p$
- Average waiting time expereince to satisfy Demand misses < the average waiting time on disk (Tdisk)
    - $L + (d * C) + Twait < Tdisk$

Looking at Conservative Pre-fetching (PonD)

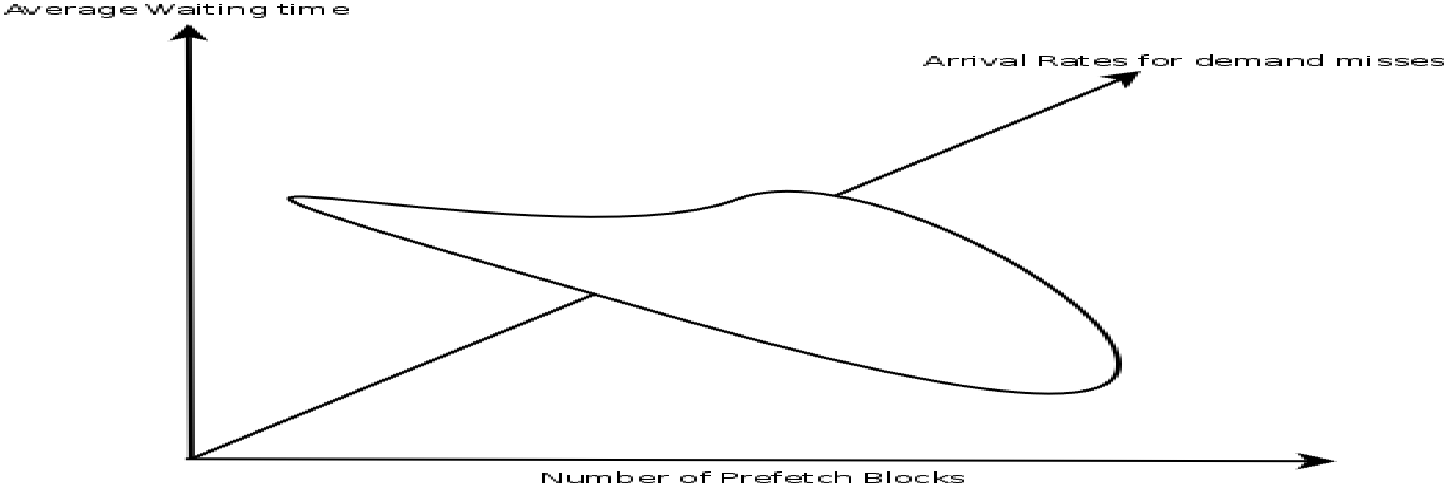
- Prefetch only when there is a demand miss

- Therefore, using PonD:
  - $L + (p+d) * C < (T_{cpu} * p)$

- For demand misses, the waiting time < Tdisk:
  - Time to fetch + Twait < Tdisk
    - $L + (p+d) * C + T_{wait} < T_{disk}$

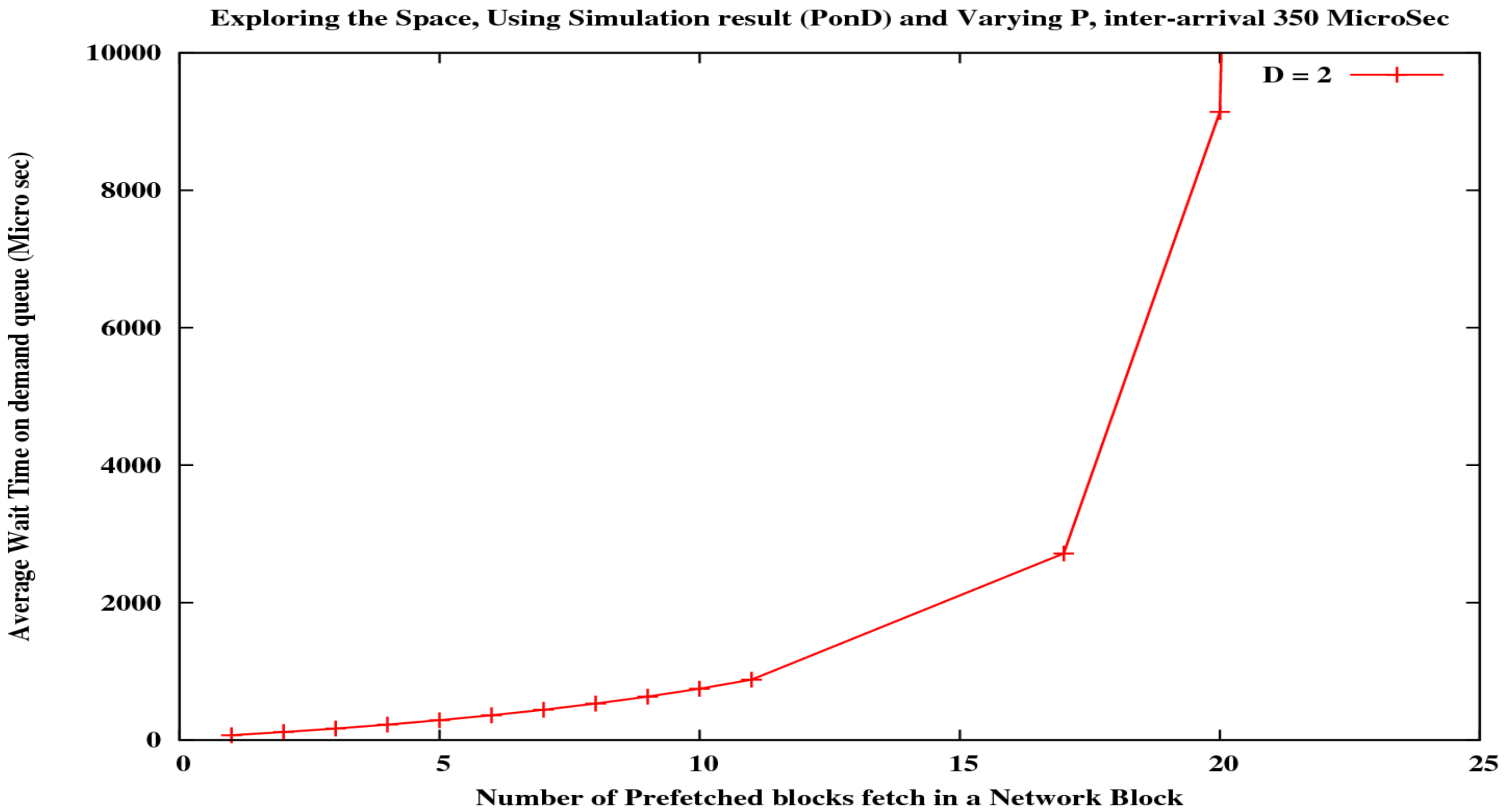
Developing an Operational Space

- Define an operational space consisting of three axes

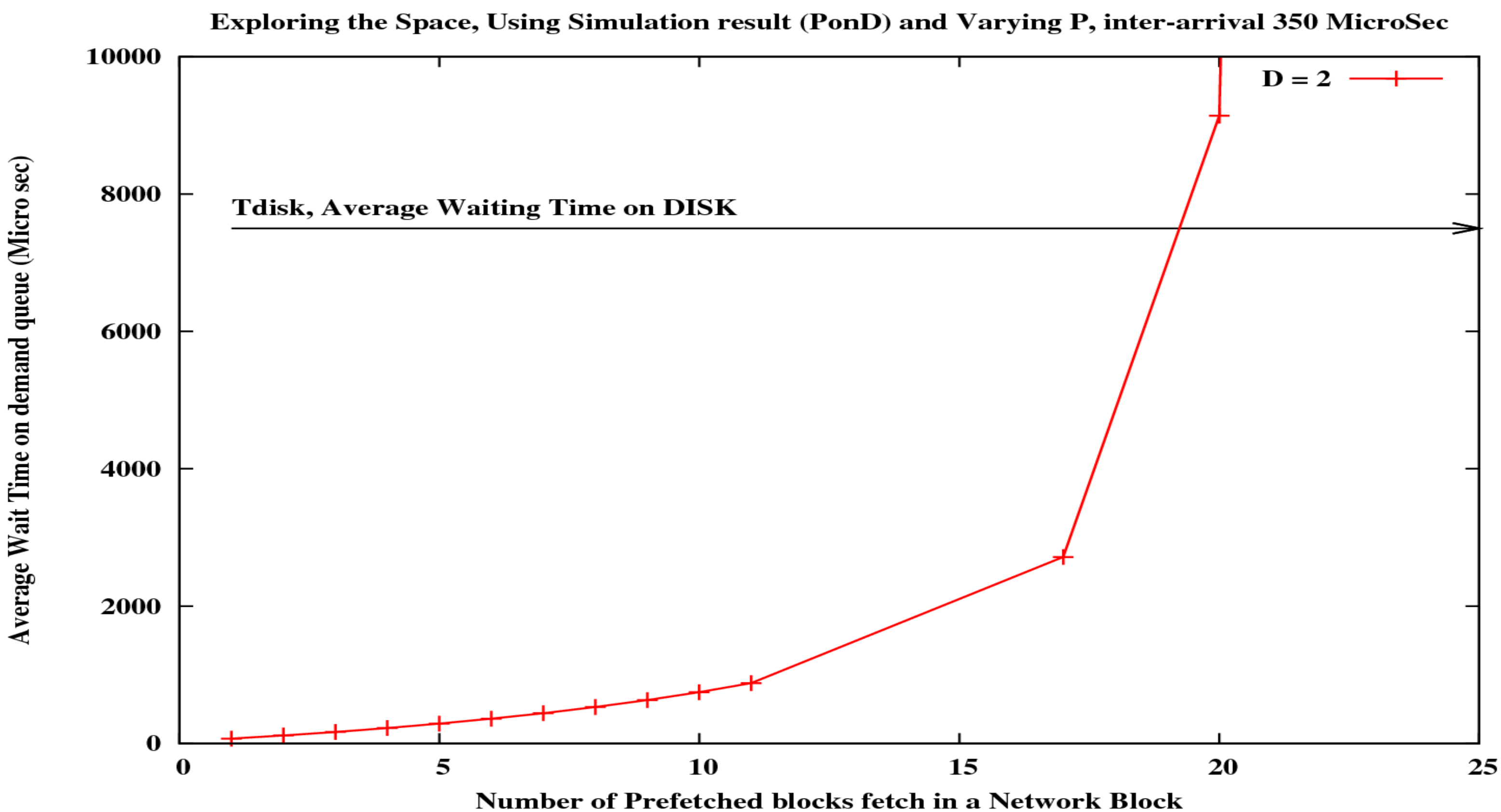


- Twait, demand misses < Tdisk
- Tnet(p+d) < P* Tcpu

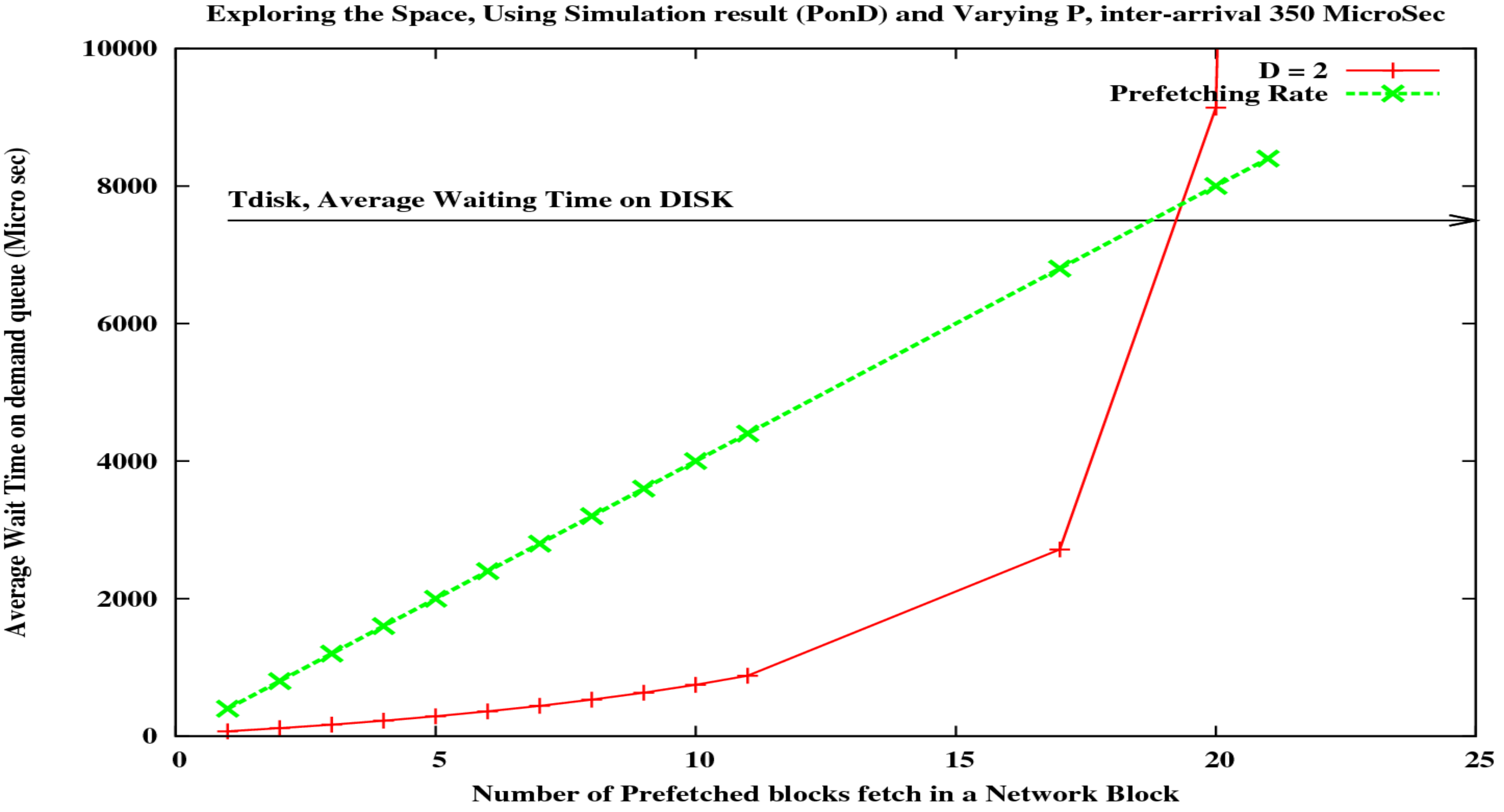
Exploring the Space for a given inter-arrival time, 350 microseconds



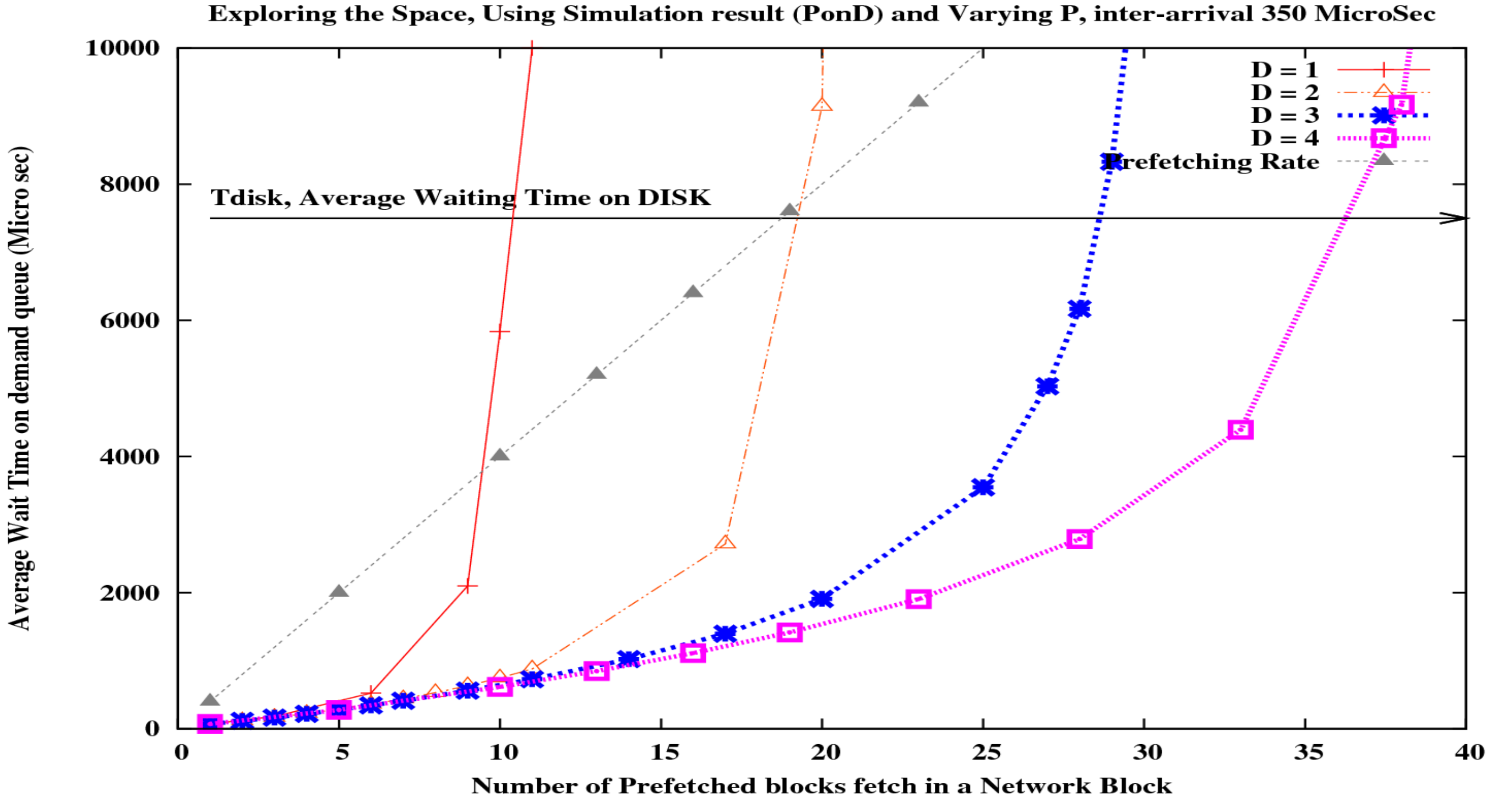
Exploring the Space, 350 microseconds



Exploring the Space, 350 microseconds



Exploring the Space, 350 microseconds



Future Work

- Further explore analytical model
- Investigate the effect of different network loads
- Develop a practical algorithm that can be part of an autonomous system that can balance pre-fetching and demand paging.
- Develop a file server that uses the algorithm and measure its performance.

Thank You

QUESTIONS?