

# LoGPC: Modeling Network Contention in Message-Passing Programs

---

Csaba Andras Moritz  
Matthew I. Frank  
Laboratory for Computer Science  
Massachusetts Institute of Technology  
{andras,mfrank}@lcs.mit.edu

© Andras 1998

SIGMETRICS98 1

---

---

---

---

---

---

---

---

## Introduction

---

- ◆ LogP, LogGP are great models to capture first order system costs
- ◆ Our new model LoGPC extends LogP and LogGP capturing pipelining and network contention
- ◆ Results preview: 3 applications, 50-76% contention found

© Andras 1998

SIGMETRICS98 2

---

---

---

---

---

---

---

---

## Motivation - why do we care?

---

- ◆ Regular, tightly synchronized communication patterns successfully modeled with LogP, LogGP.
- ◆ Important classes of applications have irregular comm patterns, are not tightly synchronized or using large messages.

© Andras 1998

SIGMETRICS98 3

---

---

---

---

---

---

---

---

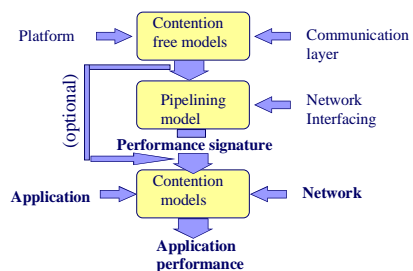
## Outline of presentation

- ◆ LoGPC methodology
- ◆ Contention-free models: LogP, LogGP
- ◆ Pipelining model
- ◆ Network contention model
- ◆ Applications

© Andras 1998

SIGMETRICS98 4

## LoGPC framework



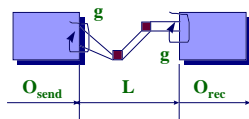
© Andras 1998

SIGMETRICS98 5

## Short messages: LogP (Culler *et al*)

4 parameters:

- ◆  $L$  = Latency
- ◆  $O$  = Overheads
- ◆  $g$  = gap minimum time interval consecutive sends and receives
- ◆  $P$  = Processors



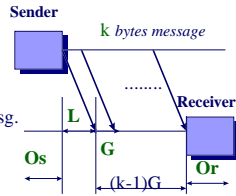
© Andras 1998

SIGMETRICS98 6

## Long messages: LogGP (Alexandrov *et al*)

A new parameter introduced:

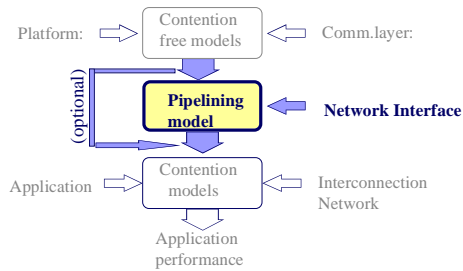
- ◆  $G$  = Gap per byte for long messg.



© Andras 1998

SIGMETRICS98 7

## LoGPC framework

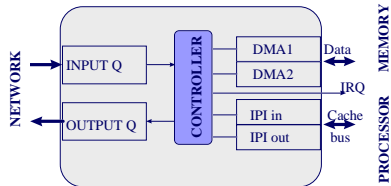


© Andras 1998

SIGMETRICS98 8

## Pipelining model

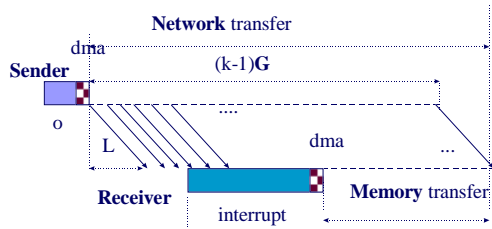
- Network Interface - Alewife



© Andras 1998

SIGMETRICS98 9

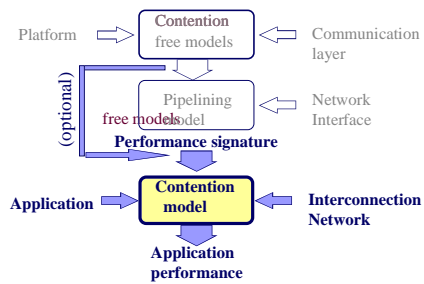
## Pipelining model



© Andras 1998

SIGMETRICS98 10

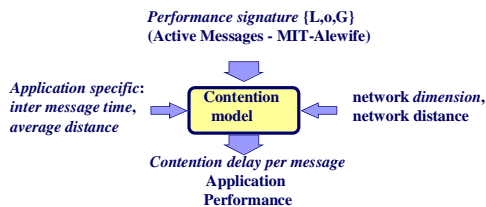
## LoGPC framework



© Andras 1998

SIGMETRICS98 11

## Network contention model



© Andras 1998

SIGMETRICS98 12

## Contention per message: $C_n$

- ◆ Start with open network model by Agarwal for expressing contention per message:



$$C_n = f(\text{application}, \text{machine})$$

$C_n$  = network contention  
 $L$  = network latency

© Andras 1998

SIGMETRICS98 13

---

---

---

---

---

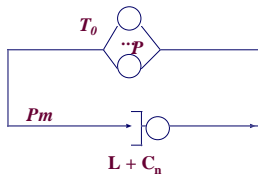
---

---

---

## Contention delay per message: $C_n$

- ◆ Close the model,  $P$  customer system



$T_0$ : inter-message time  
 $P$ : processors  
 $m$ : message rate  
 $C_n$ : contention delay  
 $L$ : network latency

- ◆ Apply Little's equation, solve for  $m$

$$Pm = \frac{P}{T_0 + (C_n + L)}$$

© Andras 1998

SIGMETRICS98 14

---

---

---

---

---

---

---

---

## LoGPC step-by-step

- Extract com. signature  $\{L, o, G\}$
- Estimate inter message time(s) based on application comm pattern(s)  $\{T_0\}$ .
- Estimate application locality (= average message distance)
- Use contention-model for contention delay per message  $\{C_n\}$ .
- Estimate runtime based on critical path

© Andras 1998

SIGMETRICS98 15

---

---

---

---

---

---

---

---



## Summary

---

- ◆ We found network contention significant
  - » all-to-all remap: 50%
  - » Diamond DAG: up to 56%
  - » EM3D: up to 70% (overall performance 20%)
- ◆ LoGPC: Simple way to evaluate how much locality matters for an application.
- ◆ LoGPC: Simple way to evaluate if network contention is significant for an application.

---

---

---

---

---

---

---

---