Bandwidth Allocation with Processing Constraints

April 8, 2004

Song Chong
KAIST
song@ee.kaist.ac.kr

Outline

❑ Introduction

❑ Resource Allocation Principle

❑ System Model

❑ Flow Control Algorithm

❑ Equilibrium, Fairness & Stability

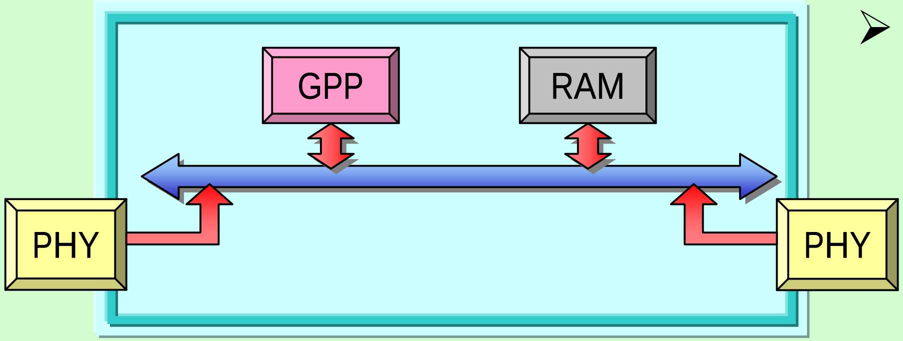
❑ IXP1200 Implementation & Performance Results

❑ Conclusion

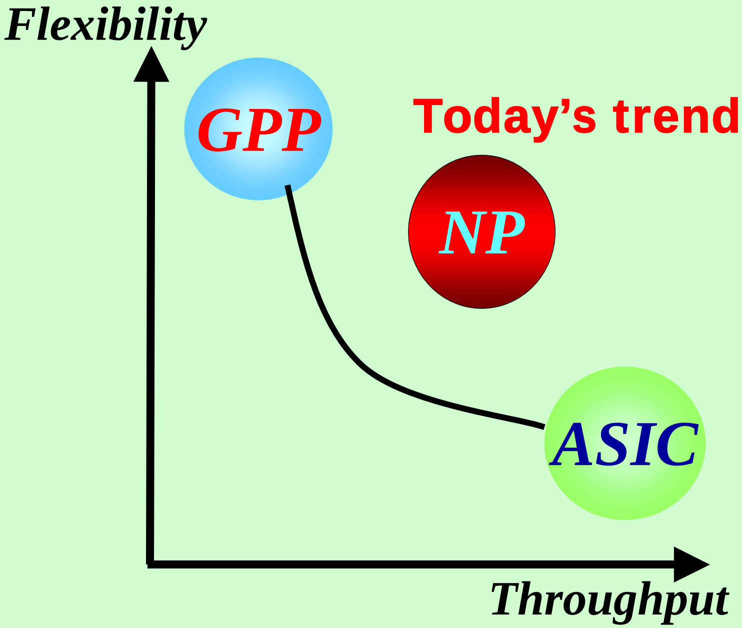
❑ References

Introduction

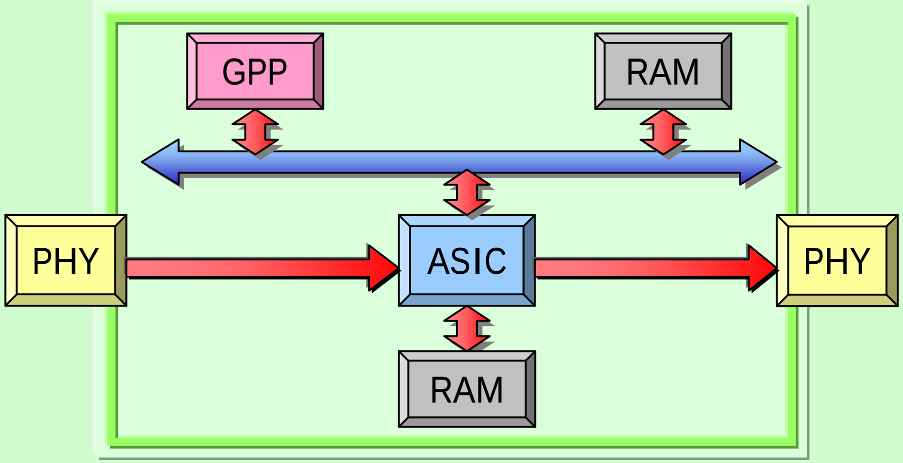
- The evolution of router architecture
  - GPP-based architecture



- All packets are processed by GPP.



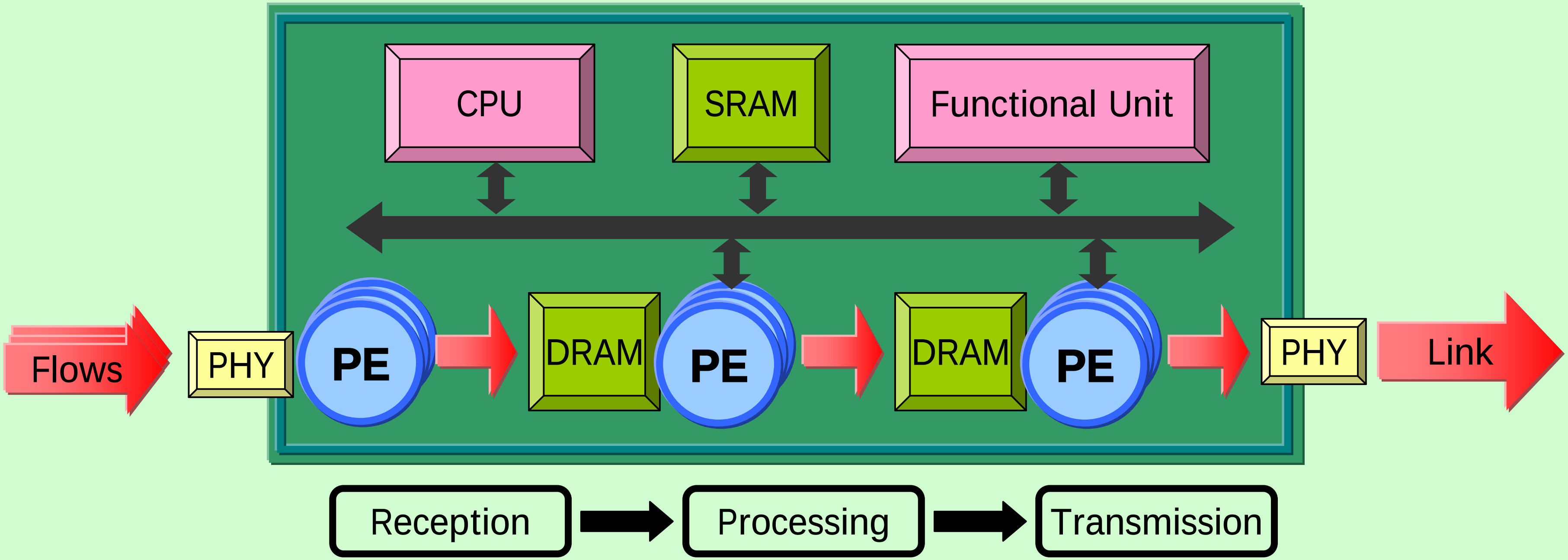
  - ASIC-based architecture



- ASIC processes the majority of packets and GPP processes unusual packets.

Introduction

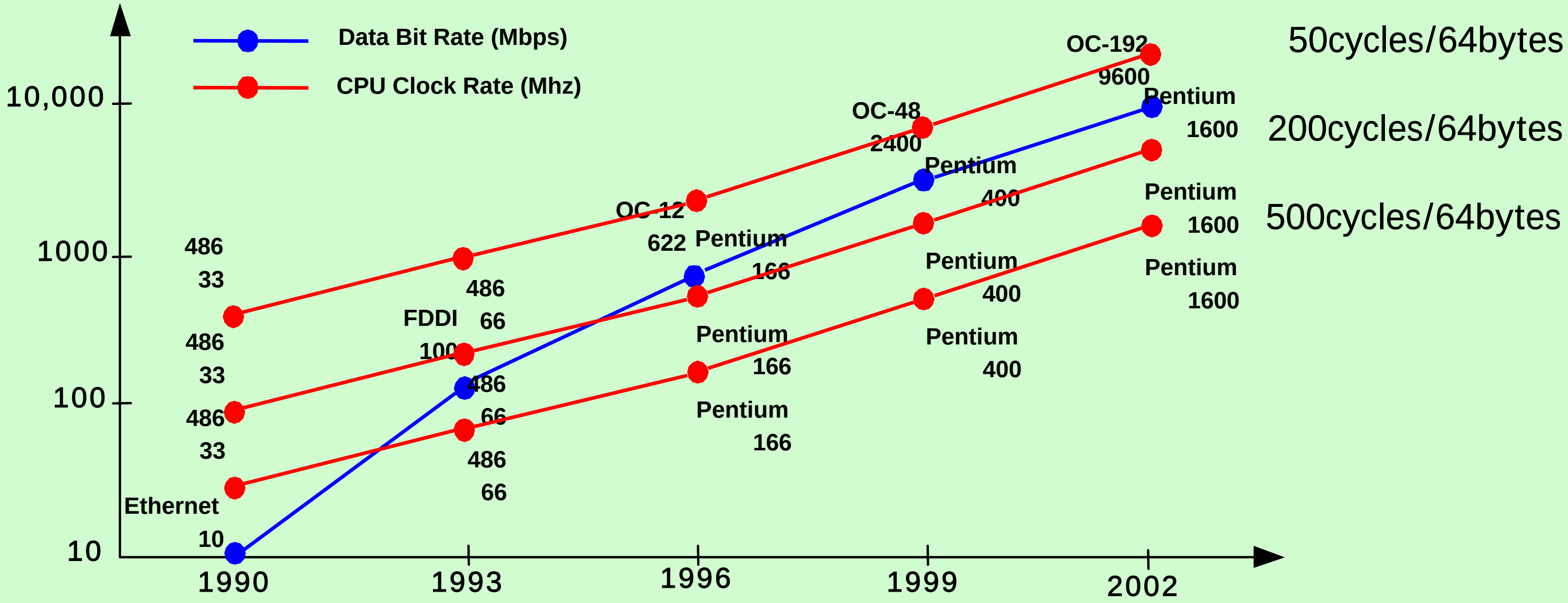
❑ Logical architecture of a general NP-based node [Johnson 2002]



❖ A network processor is mainly used for packet processing which includes CRC check, routing table look-up, header modification, and other extra processes for various network services and algorithms.

Introduction

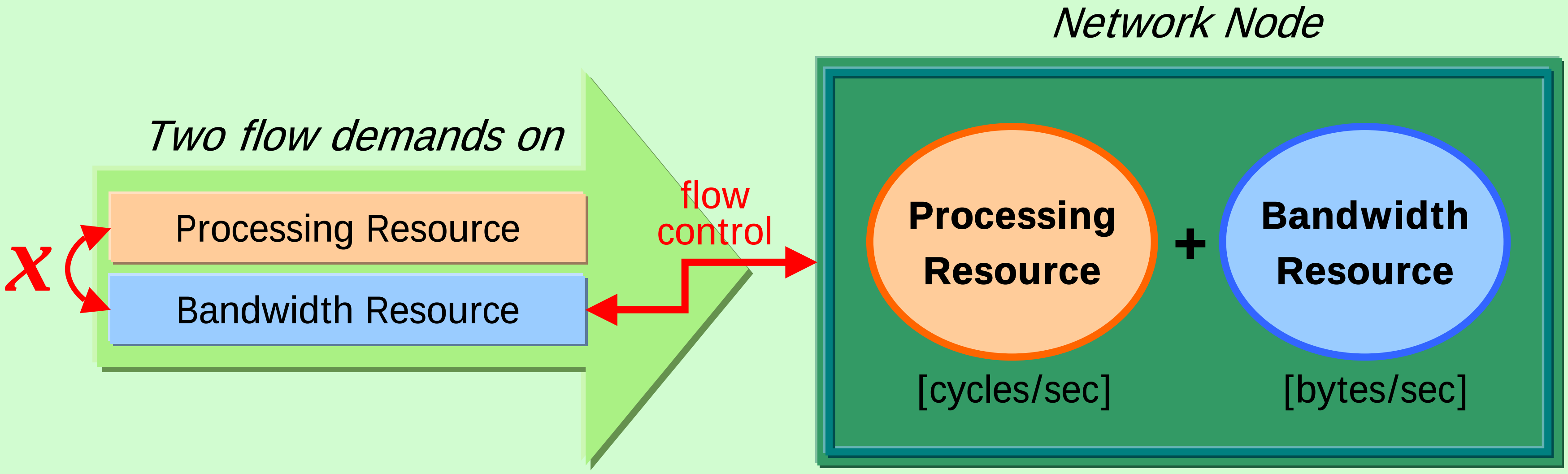
- ❑ Network BW & silicon speed growth [Herity 2001]
  - ❖ CPU speed : X 2.7 / 3year (by Moore's Law)
  - ❖ Link capacity : X 4 / 3year



Hard to predict which resource to be bottlenecked !!

Introduction

- ❑ Two-dimensional paradigm in flow control
  - ❖ Should consider both processing resource and bandwidth resource.
  - ❖ Hard to know the flow demand on processing resource until the processing completes.
  - ❖ Can control flows only in terms of "bytes/sec".



Packet Processing

- CommBench [9]: A telecommunications benchmark for network processors
- Header processing applications (HPA)
  - RTR: lookup on tree data structures
  - FRAG: header modifications and checksum
  - DRR: packet scheduling
  - TCP: pattern matching on header fields
- Payload processing applications (PPA)
  - CAST: encryption/decryption
  - ZIP: data compression
  - REED: forward error correction (FEC)
  - JPEG: media transcoding (DCT, Huffmann coding)

HPA	64	576	1536
TCP	10.3	1.2	.4
FRAG	7.7	.9	.3
DRR	4.1	.5	.2
TCP	2.1	.2	.1

(bytes)

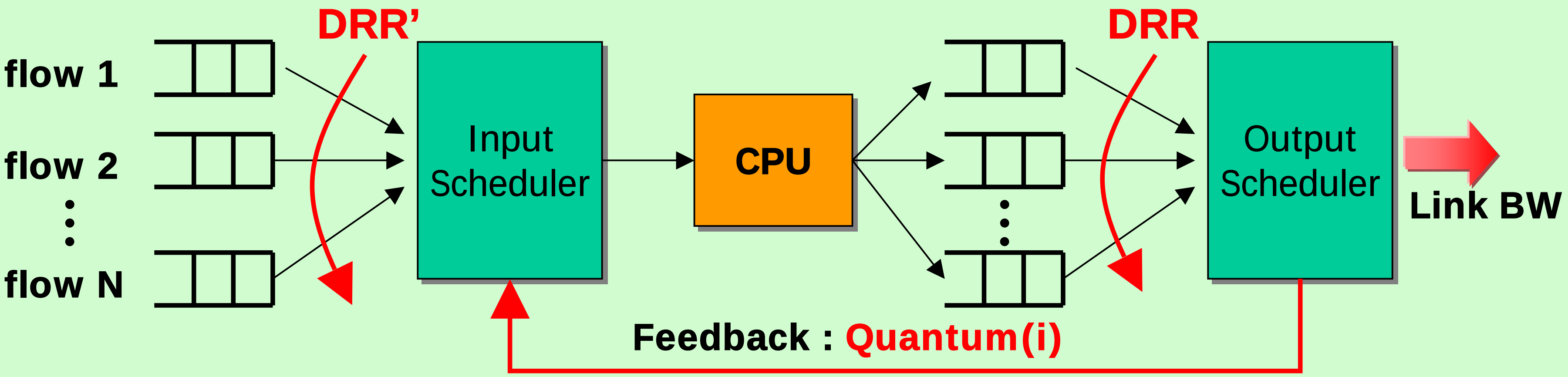
(instructions per byte)

PPA	enc	dec	
REED	603	1052	
ZIP	226	35	
CAST	104	104	
JPEG	81	60	

(instructions per byte)

Previous Work

❑ "Fair Resource Allocation in Active Networks" [Ramachandra 2000]



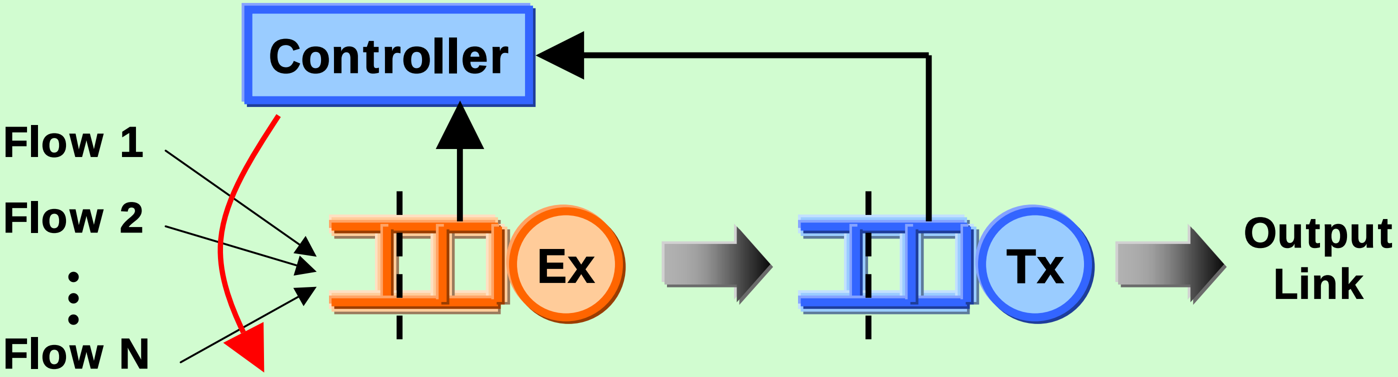
❖ **Fairness:**

- ➢ "CPU_allocation(i) + BW_allocation(i)" is to be equalized.

❖ **Weakness:**

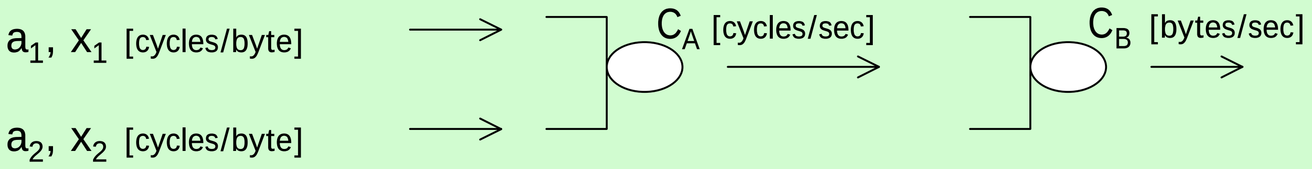
- ➢ Two-stage DRR scheduling → Complex
- ➢ Per-flow queueing → Not scalable
- ➢ No buffer control → Internal loss

Our Approach



- ❑ Fairness
  - ❖ If BW > CPU, allocate CPU MAX-MIN fair rate.
  - ❖ If BW < CPU, allocate BW MAX-MIN fair rate.
  - ❖ Otherwise, allocate the weighted average of above two rates.
- ❑ Control
  - ❖ FIFO queueing → **Simple**
  - ❖ No per-flow queueing → **Scalable**
  - ❖ Joined control of rate & queue length → **No internal loss**

Resource Allocation Principle



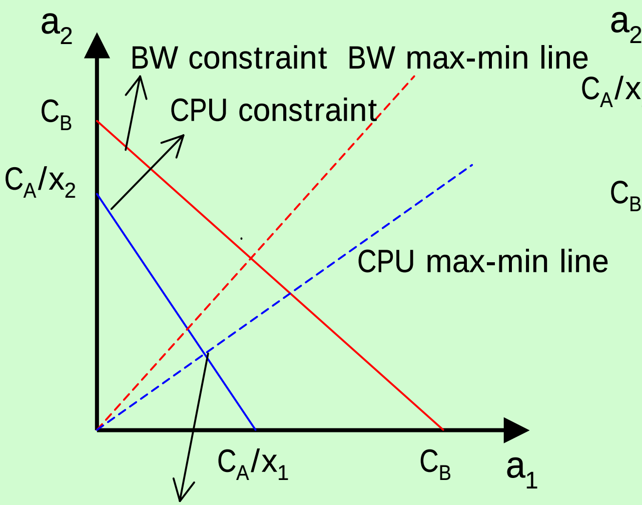
a_1, x_1 [cycles/byte]

a_2, x_2 [cycles/byte]

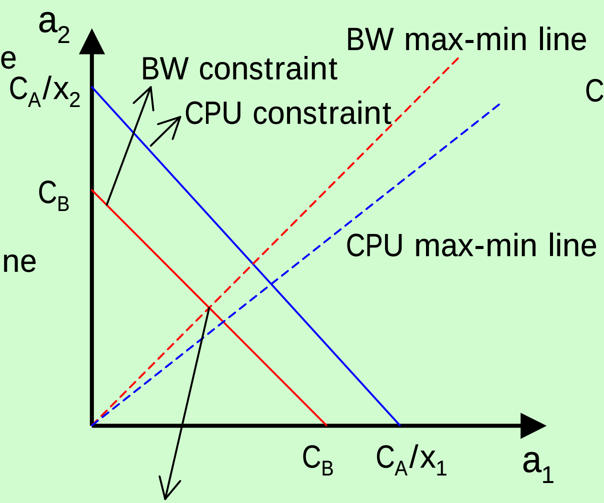
C_A [cycles/sec]

C_B [bytes/sec]

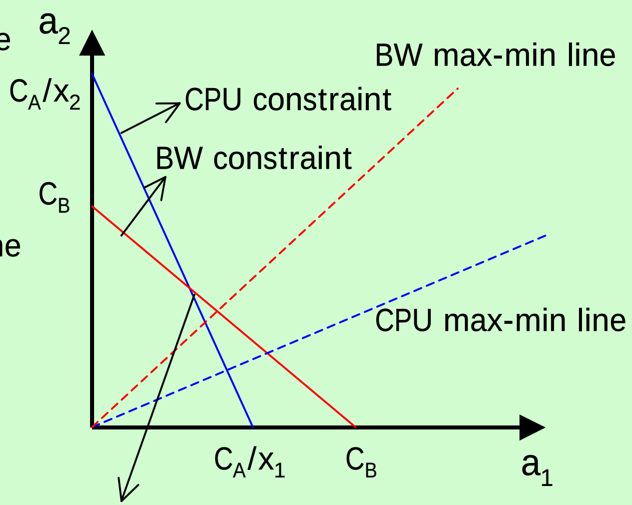
$$\begin{cases} a_1x_1 + a_2x_2 \leq C_A & : \text{ CPU constraint} \\ a_1 + a_2 \leq C_B & : \text{ BW constraint} \end{cases}$$



CPU max-min allocation



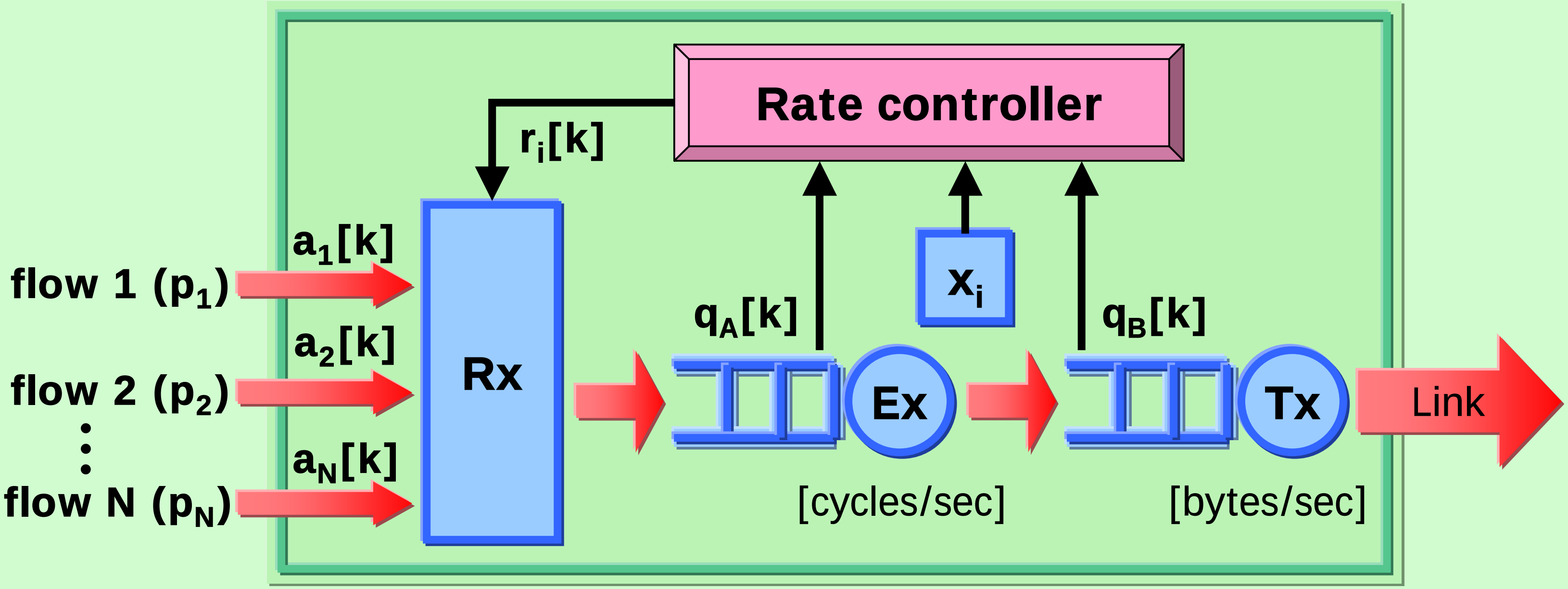
BW max-min allocation



efficiency allocation

System Model

❑ System model of a NP-based node employing the proposed approach



❖ Injected flow rate : $\mathbf{a_i[k] = \min(p_i, r_i[k])}$

❖ Processing density : $x_i = \dfrac{\text{demand for processing resource}}{\text{demand for bandwidth resource}}$ [cycles/byte]

(the only per-flow state)

Flow Control Algorithm

- Processing density update : EWMA Filter

$$\textbf{for every packet,}\quad x_i = (1-\alpha)x_i + \alpha\left(\frac{\textbf{processing time}}{\textbf{packet length}}\right) \qquad (0 < \alpha < 1)$$

- Fair rate (FR) is computed in two steps
  - Intermediate FR computation is based on PI control (**BW MAX-MIN**)

$$r[k] = \left[r[k-1] - \frac{A}{|Q|}\left(q[k-1] - q[k-2]\right) - \frac{BT}{|Q|}\left(q[k-1] - q_T\right) \right]^+$$

q[k] : total queue length ($q_A[k] + q_B[b]$)

q_T : target length of total queue

A,B : control gain

T : control period

Q : set of bottlenecked input flows (|Q| is the cardinality of Q)

Flow Control Algorithm

- ❖ Final FR computation (**CPU MAX-MIN & BW MAX-MIN**)

$$r_i\left[k\right] = \left(\frac{q_T - q_B\left[k-1\right]}{q_T}\right)\frac{\dfrac{1}{x_i}}{\displaystyle\sum_{i\in Q}\frac{1}{x_i}}\left|Q\right|r\left[k\right] + \left(\frac{q_B\left[k-1\right]}{q_T}\right)r\left[k\right], \qquad \forall i \in N$$

❑ Intelligent behavior of the algorithm

- ❖ When processing resource is the bottleneck, allocate CPU MAX-MIN rate.
- ❖ When bandwidth resource is the bottleneck, allocate BW MAX-MIN rate.
- ❖ When both resources are bottlenecked together, determine the degree of bottleneck intensity of each resource and allocate rate as a convex combination of CPU MAX-MIN rate and BW MAX-MIN rate.

Steady State Solutions

- Three steady state solutions according to C_A, C_B, and the following two averages of x_i.

$$\bar{x}_n = \frac{\sum_{i \in Q} x_i}{|Q|} \quad (\text{numerical avg.}), \qquad \bar{x}_h = \left(\frac{\sum_{i \in Q} \frac{1}{x_i}}{|Q|} \right)^{-1} \quad (\text{harmonic avg.})$$

  - If $C_A^* \geq \bar{x}_n C_B^*$ (bandwidth resource is the bottleneck),

$$q_A^* = 0, \quad q_B^* = q_T, \quad a_i^* = \frac{C_B}{N}$$

  - If $C_A^* \leq \bar{x}_h C_B^*$ (processing resource is the bottleneck),

$$q_A^* = q_T, \quad q_B^* = 0, \quad a_i^* = \frac{C_A}{N x_i}$$

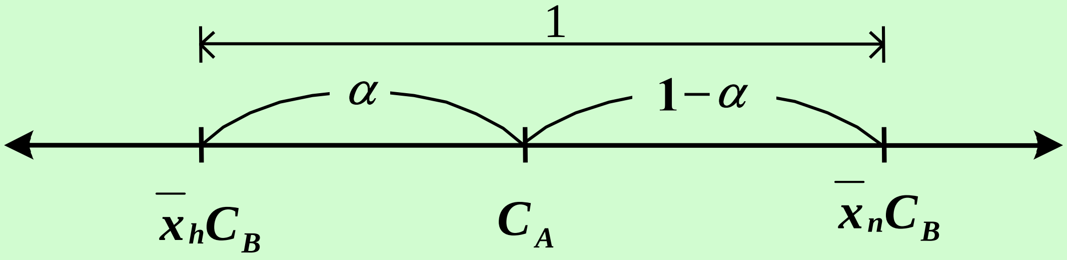
Steady State Solutions

- If $\overline{x}_h C_B^* < C_A^* < \overline{x}_n C_B^*$ (both resources are bottlenecked together),

$$q_A^* = (1-\alpha) q_T, \quad q_B^* = \alpha\, q_T, \quad a_i^* = (1-\alpha)\frac{\frac{1}{x_i}}{\sum_{i\in N}\frac{1}{x_i}} C_B + \alpha\frac{C_B}{N}$$

where, $$\alpha = \frac{C_A - \overline{x}_h C_B}{\overline{x}_n C_B - \overline{x}_h C_B}$$

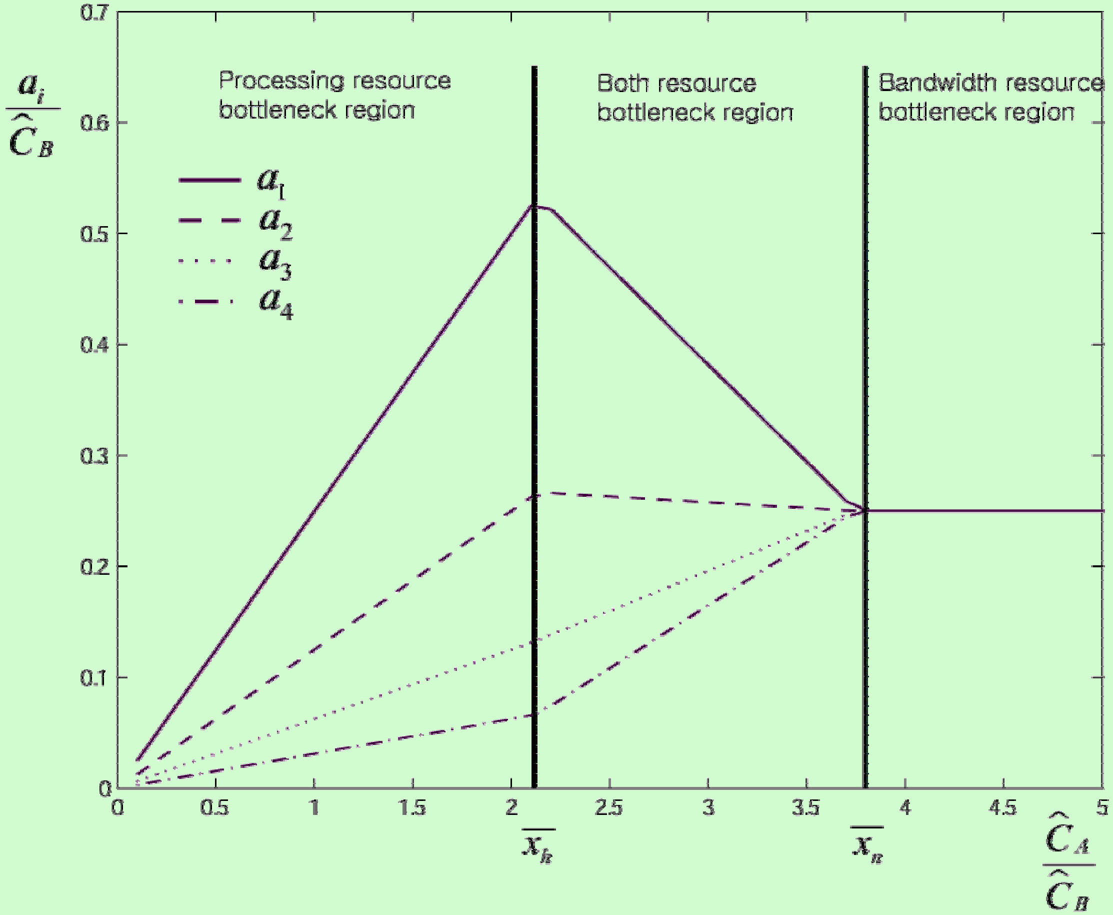
***α* determines the degree of bottleneck intensity.**



Example

4 flows with different x_i s.

$(x_1, x_2, x_3, x_4) = (1, 2, 4, 8) \qquad \overline{x}_h = 2.13, \ \overline{x}_n = 3.75$



Asymptotic Stability

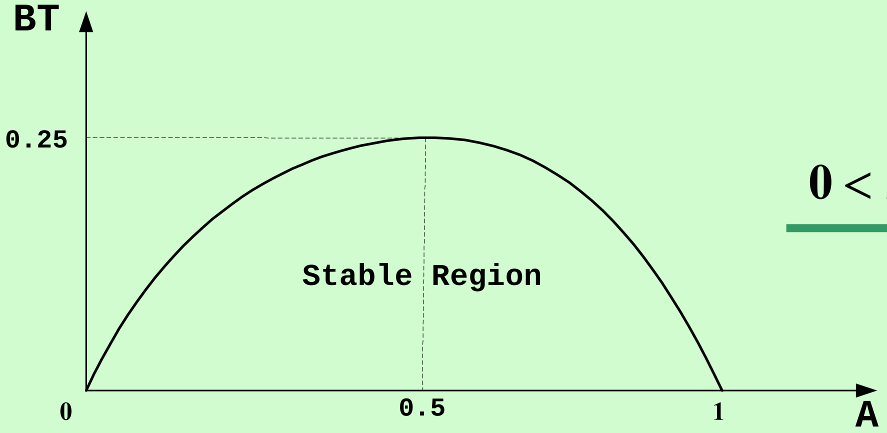
- ❑ Error function : $e[k] = q[k] - q_T$

- ❑ Closed-loop difference equation of **e[k]**:

$$e[k+1] - 2e[k] + (1 + A + BT)e[k-1] - Ae[k-2] = 0$$

- ❑ Characteristic equation : $z^3 - 2z^2 + (1 + A + BT)z - A = 0$

- ❑ By **Schűr-Cohn Stability Criteria** [7], the closed-loop equation is asymptotically stable if and only if



$$0 < A < 1, \quad 0 < BT < A(1-A)$$

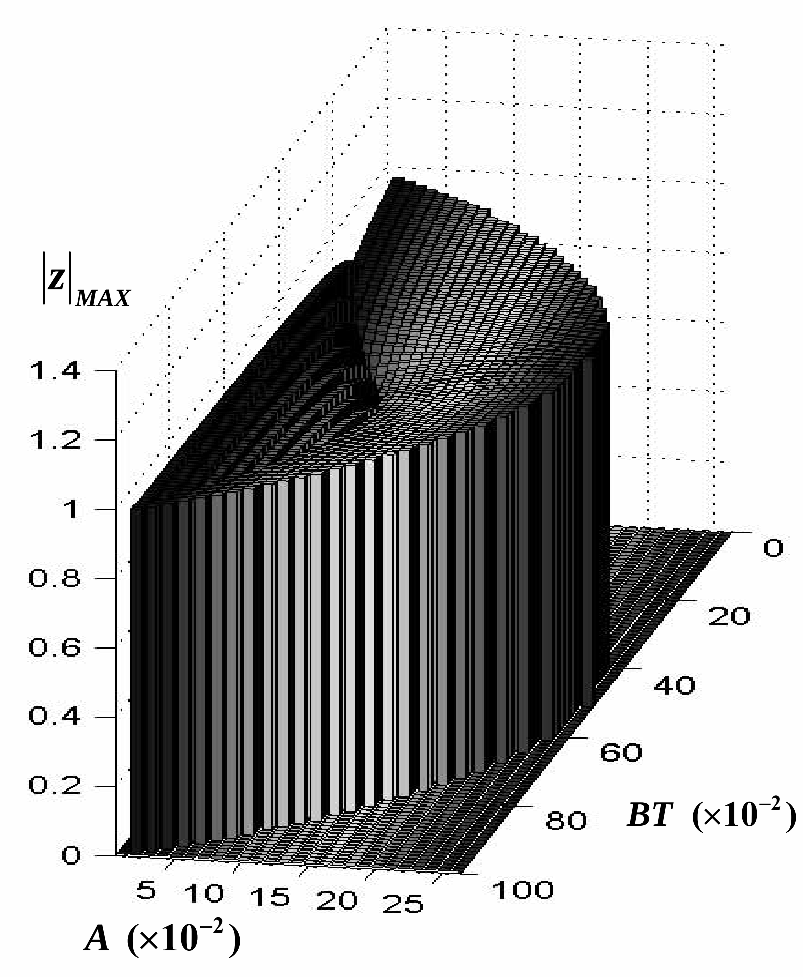
Asymptotic Decay Rate

- Optimal gains
  - Using numerical analysis,

$$A = 0.32$$
$$BT = 0.05$$

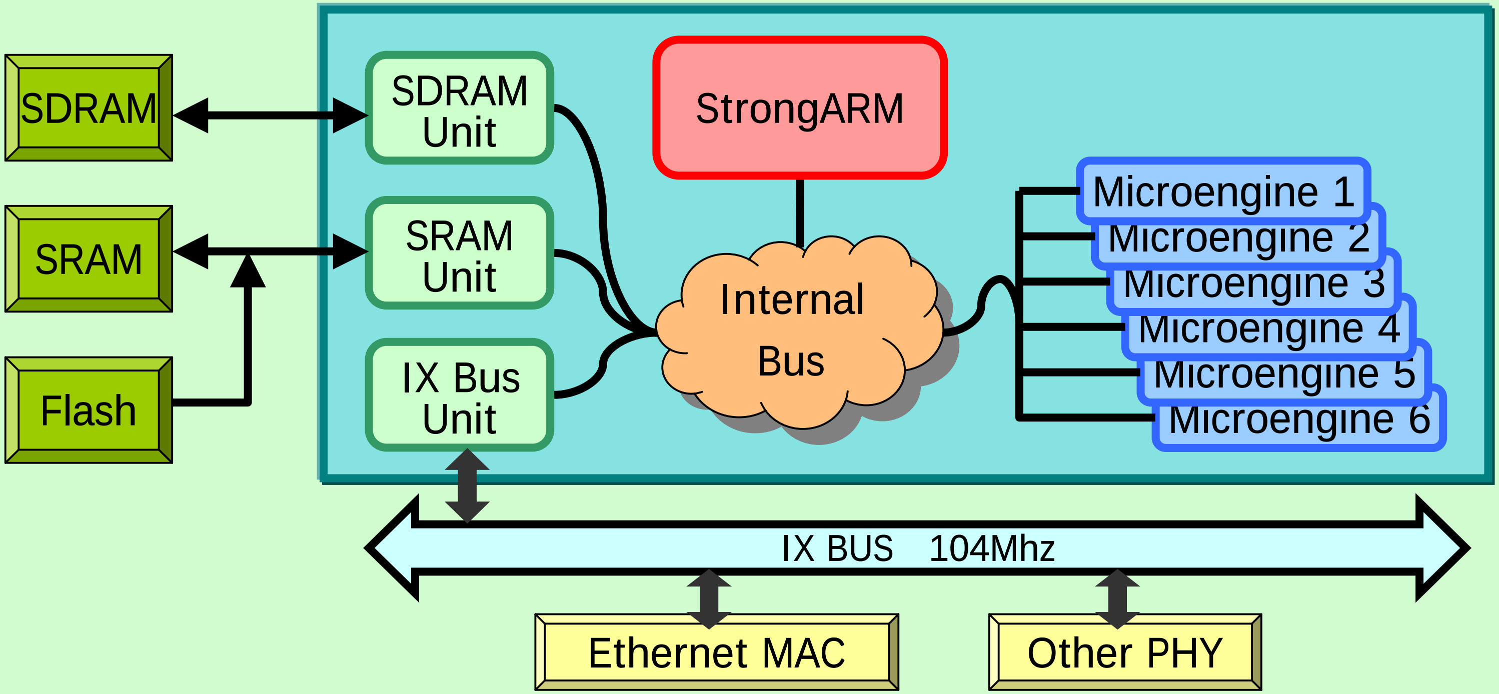
  - Asymptotic decay rate :

$$0.704 \; / \; T$$



Simulation Environment

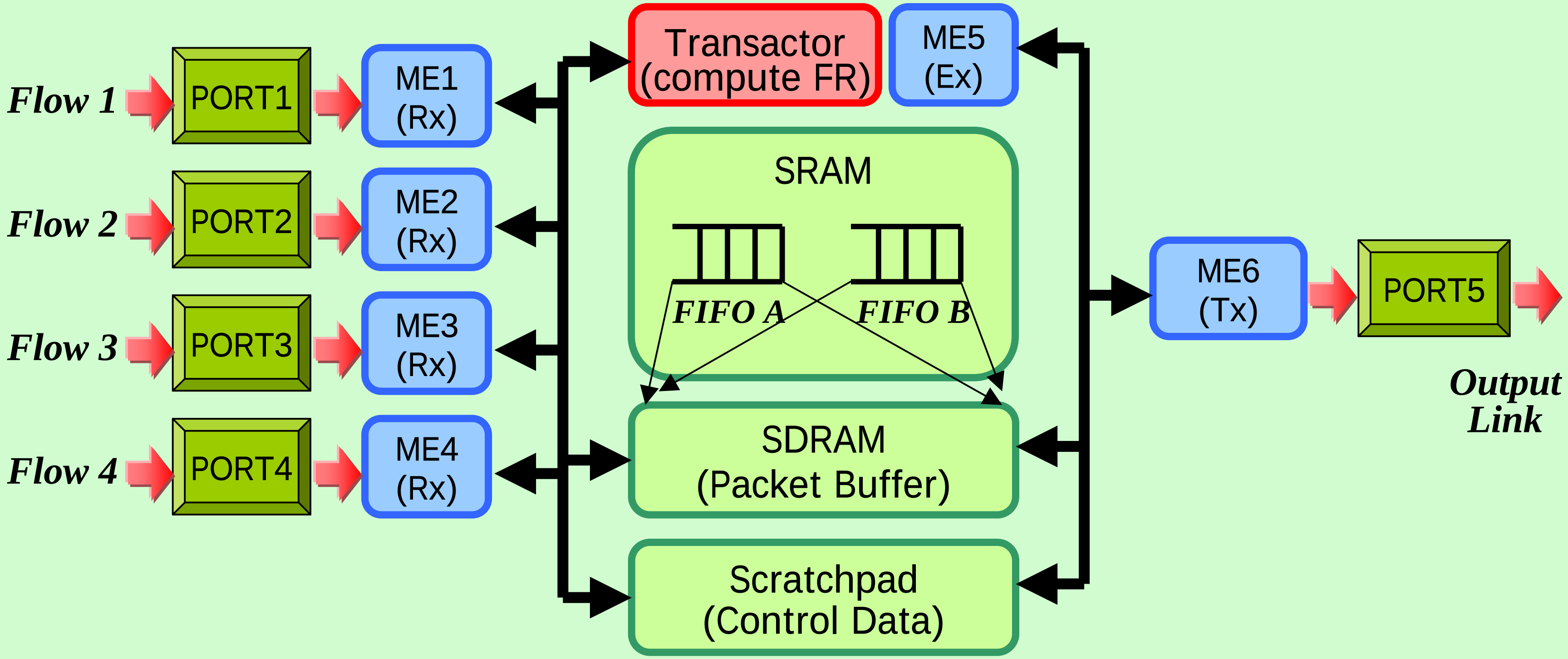
- Intel IXP1200 Evaluation Platform [9]



- IXP1200 Network Processor
  - Consists of one StrongARM Core(166Mhz) and six Microengines(166Mhz).
  - Provides four hardware contexts with zero context switching overhead in each six microengine.

Simulation Environment

- ❑ Implemented IPv4 (RFC 1812) forwarding engine on ME5.



Simulation Results

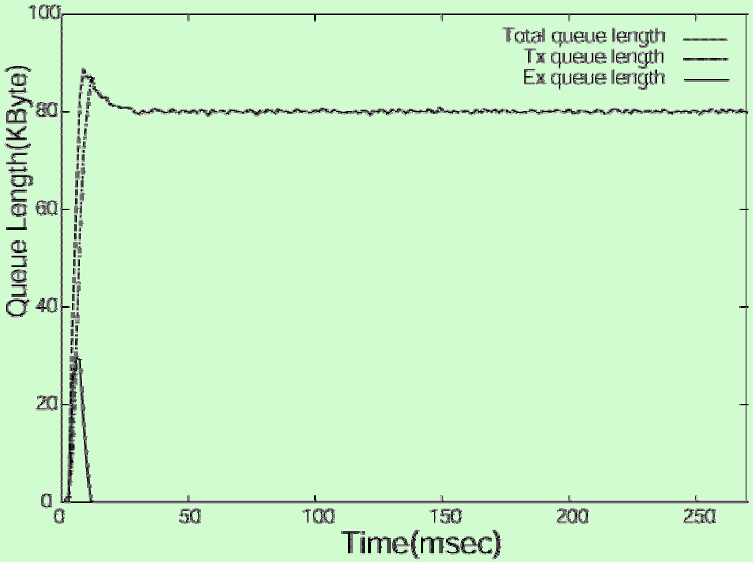
Bandwidth Resource Bottleneck Case

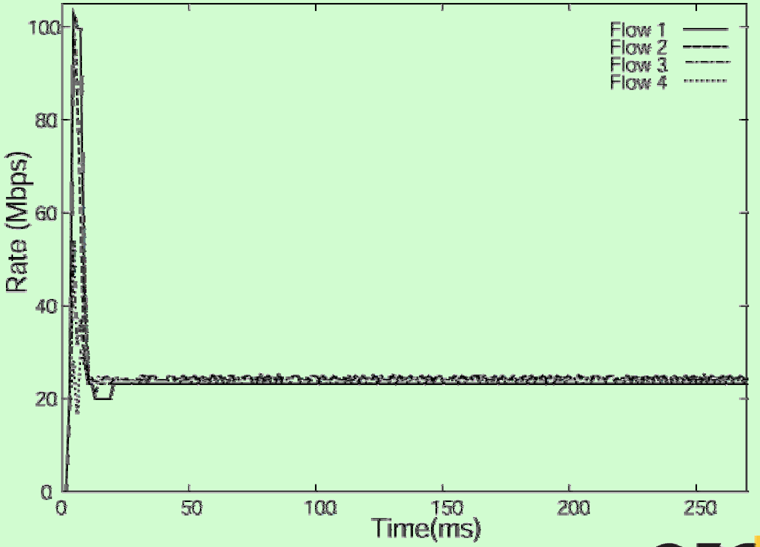
Flow No.	x_i	Fair Rate (Mbps)	Actual Rate (Mbps)
0	2.5	25.0	23.2
1	5.0	25.0	23.8
2	10.0	25.0	24.4
3	20.0	25.0	24.7

C_B = 100 Mbps

Tx queue length : 80015 bytes

Ex queue length : 0 bytes



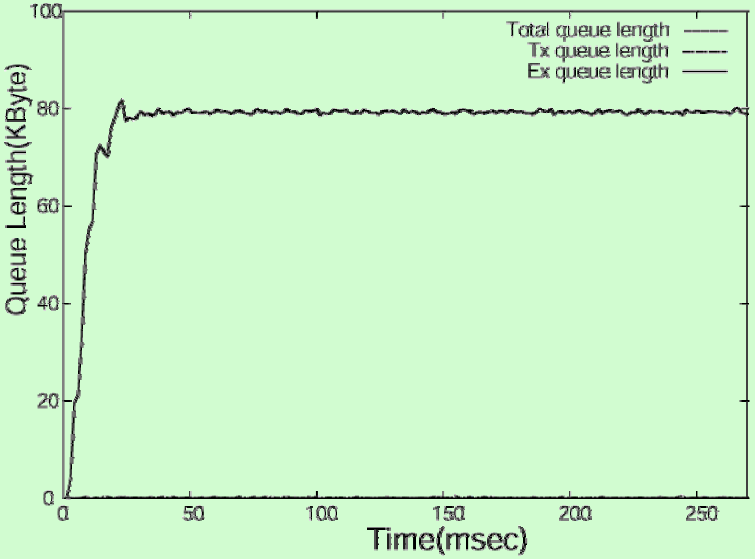


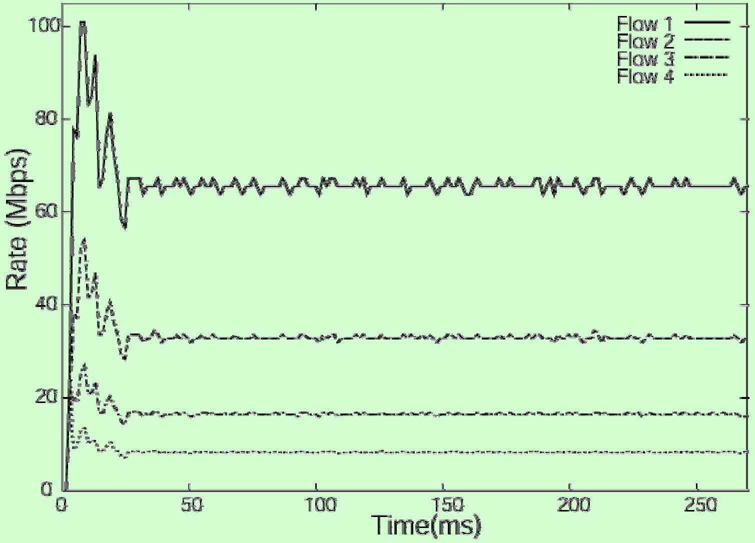
Simulation Results

Processing Resource Bottleneck Case

Flow No.	x_i	Fair Rate (Mbps)	Actual Rate (Mbps)
0	4.7	70.8	65.7
1	9.4	35.4	32.9
2	18.8	17.7	16.5
3	37.5	8.9	8.3

C_B = 300 Mbps
Tx queue length : 150 bytes
Ex queue length : 79217 bytes





Simulation Results

Both Resources Bottleneck Case I

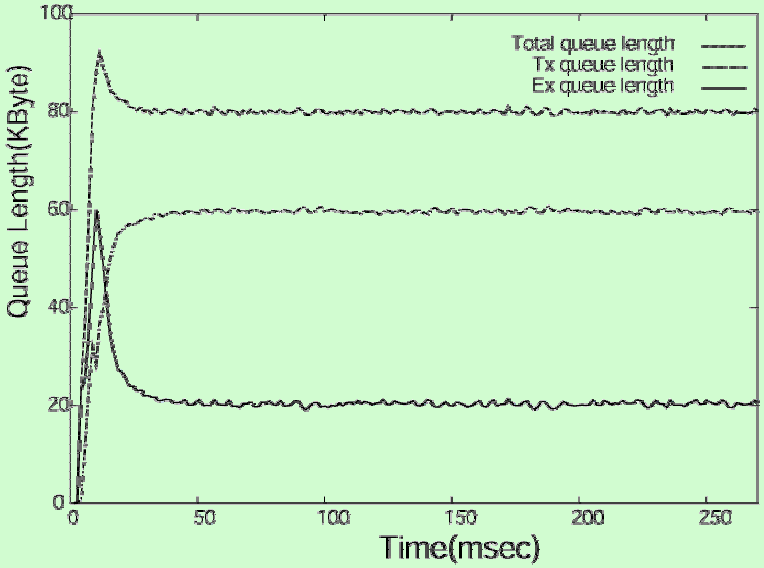
(BW is more bottlenecked: $\alpha = 0.746$)

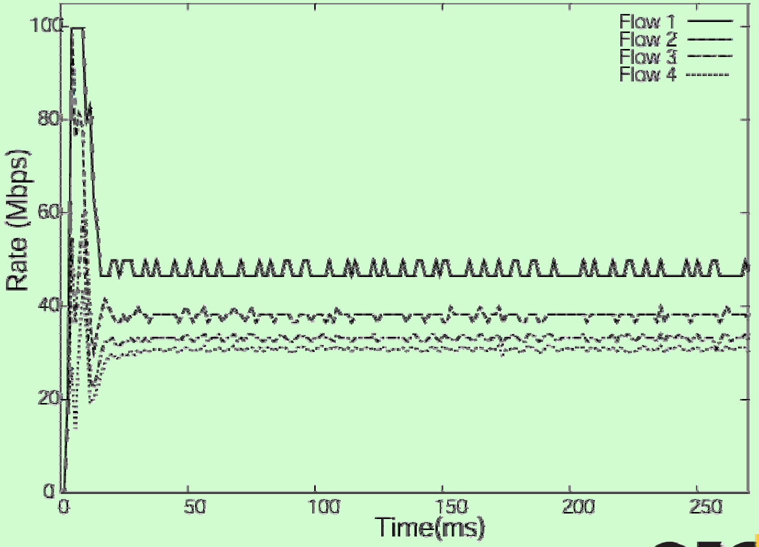
Flow No.	x_i	Fair Rate (Mbps)	Actual Rate (Mbps)
0	2.5	50.3	47.4
1	5.0	40.1	37.9
2	10.0	35.1	33.2
3	20.0	32.5	30.8

C_B = 158 Mbps

Tx queue length : 56699 bytes

Ex queue length : 20316 bytes



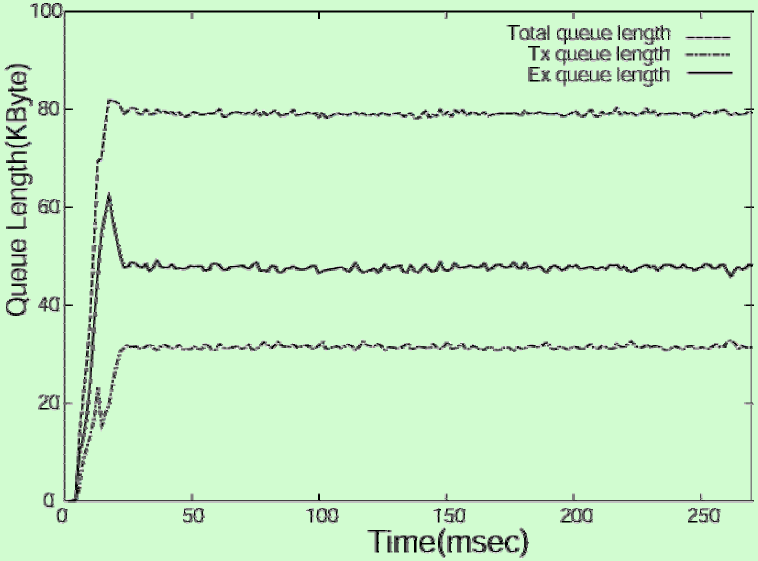


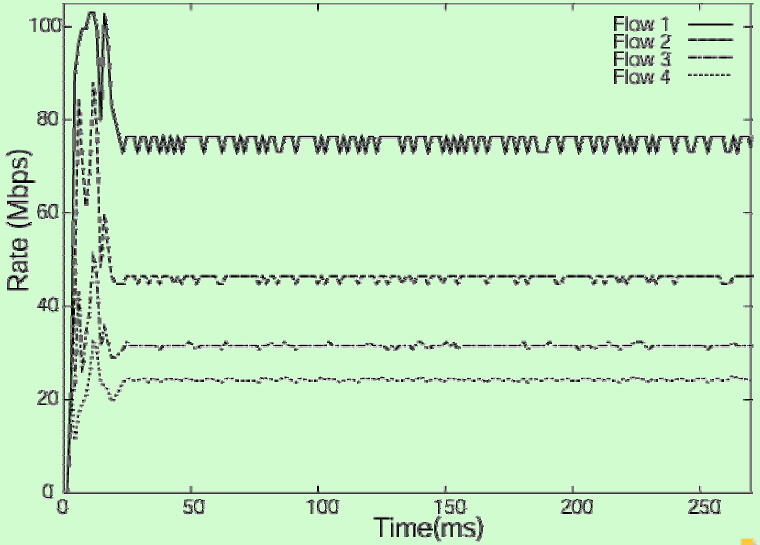
Simulation Results

Both Resource Bottleneck Case II
(CPU is more bottlenecked: $\alpha = 0.393$)

Flow No.	x_i	Fair Rate (Mbps)	Actual Rate (Mbps)
0	2.5	78.4	75.3
1	5.0	49.1	46.1
2	10.0	34.4	31.5
3	20.0	27.1	24.3

C_B = 189 Mbps
Tx queue length : 31441 bytes
Ex queue length : 47703 bytes





Conclusion

- ❑ Fair resource allocation when processing time is non-negligible
- ❑ New fairness characterization
- ❑ Control-theoretic algorithm
- ❑ Implementation of IP data path on IXP1200

- ❑ Future work
  - ❖ Distributed algorithm for a network of programmable nodes
  - ❖ Understanding in optimization perspective
  - ❖ Microscopic modeling of network processor architecture
  - ❖ Joint management of bandwidth, processing and power

References

[1] D. Herity, “Network Processor Programming”, *Embeded Systems Programming*, July 2001.

[2] D. L. Tennenhouse, J. M. Smith, “A Survey of Active Network Research”, IEEE Communications, 35(1):80-86, January 1997.

[3] Vijay Ramachandran, “Fair Resource Allocation in Active Networks”, *Computer Communications*, 2000.

[4] Song Chong et. al, “A Simple, Scalable, and Stable Explicit Rate Allocation Algorithm for Max-Min Flow Control with Minimum Rate Guarantee”, *IEEE/ACM Transactions on Networking*, Vol. 9, June 2001.

[5] T. Spalink, S. Karlin, L. Peterson, “Building a Robust Software-Based Router Using Network Processors”, *Proceedings of the 18th ACM Symposium on Operating Systems Principles(SOSP)*

[6] Y.D. Lin, “DiffServ over Network Processors: Implementation and Evaluation”, *IEEE/Proceedings of the 10th Symposium on High Performance Interconnections Hot Interconnects*, HotI 2002.

[7] Erik J. Johnson and Aaron R. Kunze, “IXP1200 Programming”, *Intel Press*, 2002.

[8] IXP1200 Hardware Reference Manual

[9] T. Wolf and M. Franklin, “CommBench – A Telecommunications Benchmark for Network Processors”