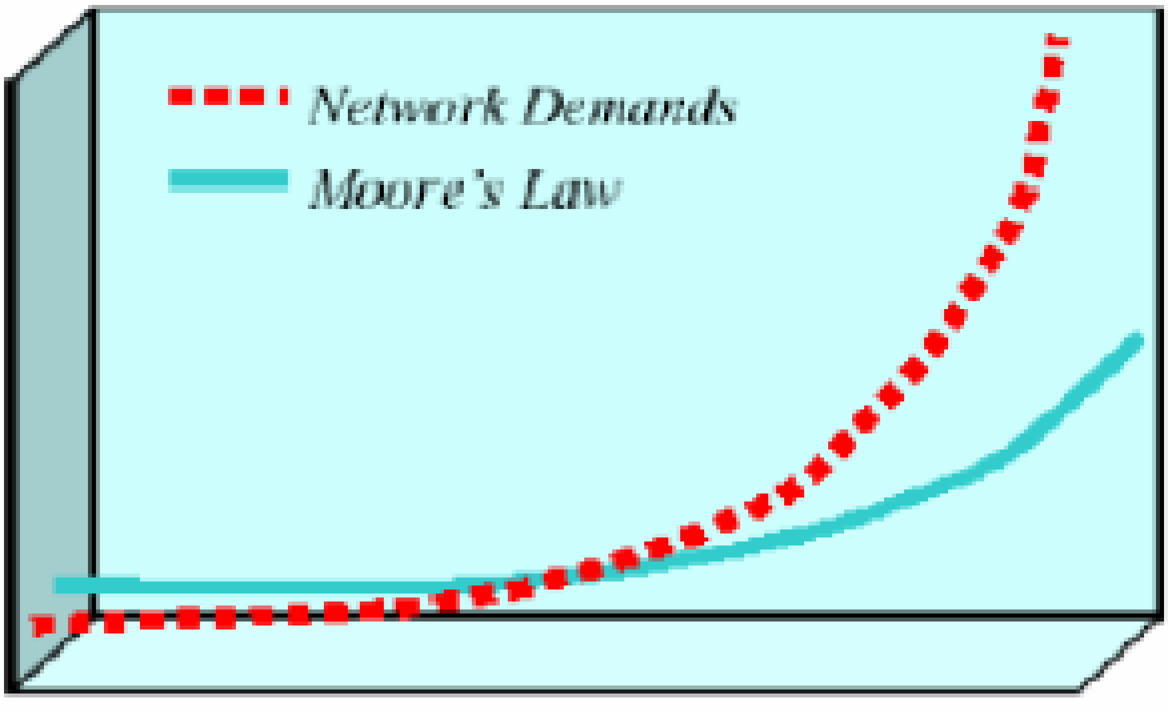
Automated Task Distribution in Multicore Network Processors using Statistical Analysis

Arindam Mallik, Yu Zhang, Gokhan Memik

Electrical Engineering and Computer Science Dept.
Northwestern University

Network Demand Gap



Gap increases with the time [Intel]

The Path to ASIPs

- Application Specific IC design
  - Costly
  - Unpredictable
- Fuels the rise of programmable devices or ASIPs (Application Specific Instruction Processors)
  - Networking
  - Multimedia
  - Graphics
- ASIPs
  - Architectures have been explored in great depth
  - Modest progress on programming environments
  - But, the success of users is dependent on their ability to program effectively

Why Network Processors ?

- Traditional processors in networks
  - General-purpose CPU
    - Not fast enough to handle new link speeds
  - ASIC
    - Good performance, but lack flexibility. New applications or protocols make the old processor obsolete
    - Frequent new applications
- Solution: Network Processors
  - Programmable processors optimized for networking applications
  - Reusability of the same processor core for different network applications

Overview

- Chip Multiprocessors
  - Most current processor architectures
  - Ideal for networking application
    - Data level parallelism
    - Task level parallelism
  - Dominating from the start - Intel IXP
- Low scalability of interconnect networks
  - Importance of local communication
  - Uniform task distribution

Outline

- Introduction
- Click Router Architecture
- Statistical Task Allocation
- Results
- Conclusion

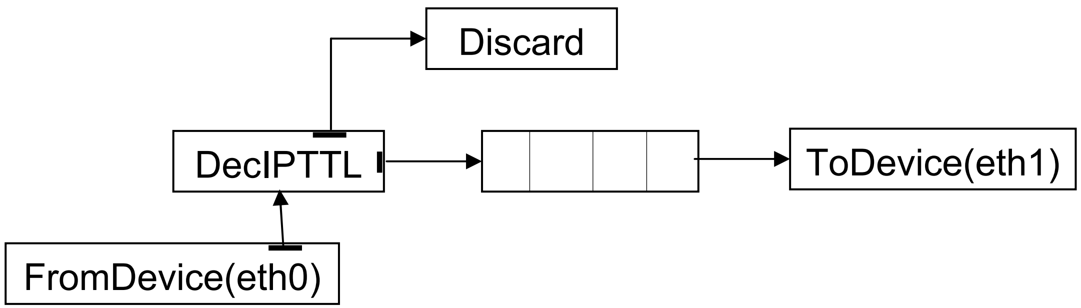
Modularity in Networking Apps

- Presence of well defined data segments (packets)
- Independent packet processing
- Overlooked modularity
  - Set of independent tasks performed on each packet - module
  - Majority of networking applications – collection of standard modules (ttl, checksum calculation)

Click Architecture

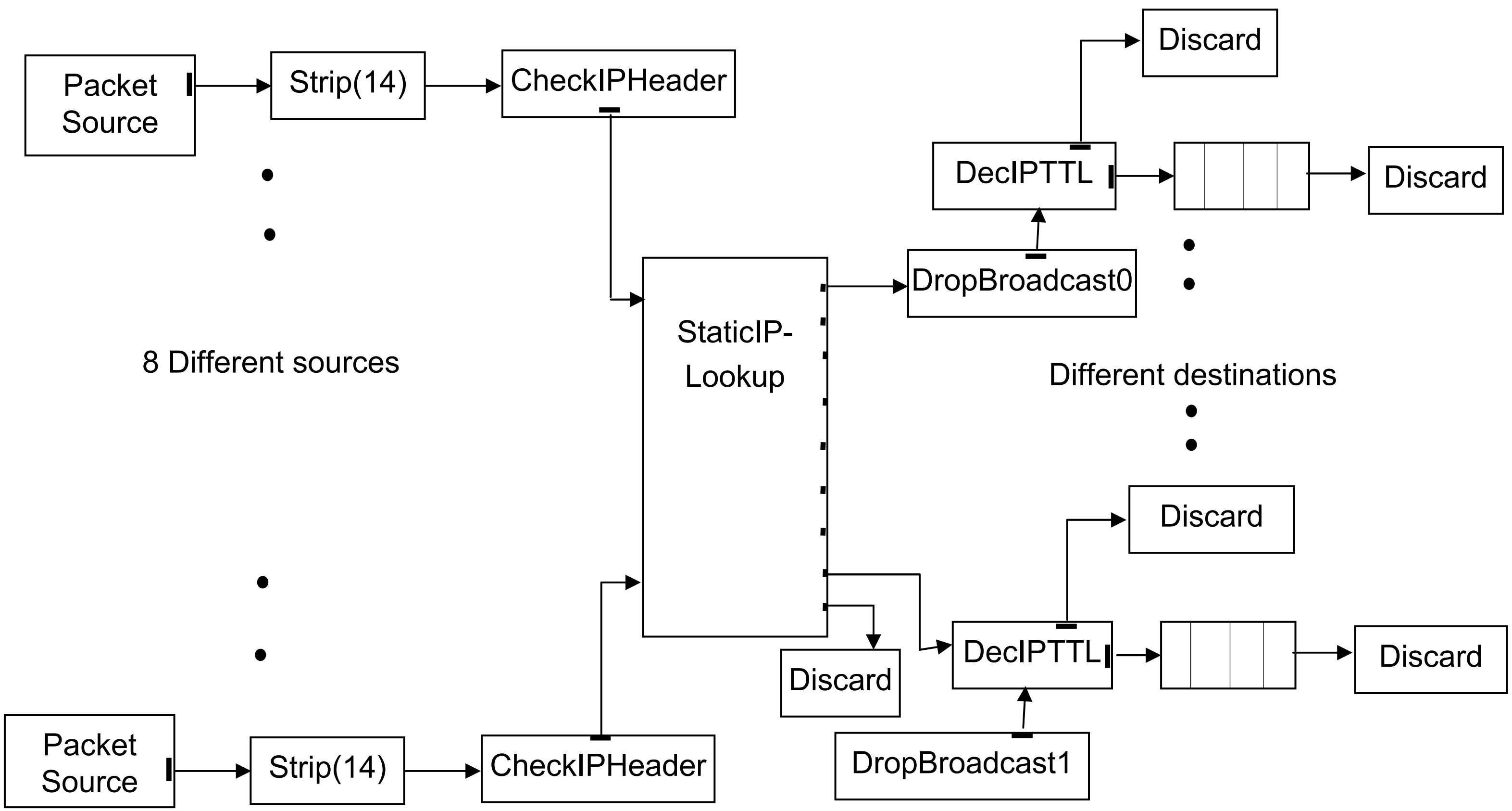
- Unit of processing
  - ‘element’–(From/ToDevice, GetHeader, Discard, Count…)
  - element encapsulates processing actions and state
  - elements have input and output ports
  - language level compositions of elements
- Router configuration
  - directed graph of elements (cycles ok), connected by ‘connections’ (at ports)
- Each packet follows connections
- Configuration string
  - parameters and initial state to instantiate an element

Click Configuration Example



- Configuration checking the TTL value of a packet

IPv4 Router Example



Statistical Task Allocation

- Systolic Array Architecture
  - Execution cores arranged in pipelined fashion
  - Global communication using shared bus
- Goal : Uniform Task Allocation
  - Automated
  - Each core sends partially processed packet to the next one

Module Distribution Algorithm

- Profiling
  - Statistical Analysis of packet processing time
- Streamlining
  - Find total execution time of a packet
  - Use DFS on the element tree
- Task Distribution
  - Assign elements to different stages/modules
- Local optimization

Statistical Analysis of Packet Processing

- Individual Elements
  - Executed for 5000 packets
  - Execution time recorded for each packet
  - Mean (μ) and standard deviation (σ) calculated from the statistics
  - expression ($\mu+k\sigma$) estimates variation of utilization

Prob. Distn. of IPv4 Elements

Elements	Mean (μ)	SD (σ)	Processing time threshold				
			μ	μ +σ	μ+2σ	μ+3σ	μ+4σ
strip0	241.28	29.31	50	0.64	0.64	0.64	0
chkip0	713.01	59.77	50	0.64	0.64	0.64	0.64
RtLkUp	336.56	266.88	20.03	20.03	10.01	0.03	0.03
DBC0	212.30	21.18	34.32	28.57	1.29	0.18	0.18
DcTTL0	317.78	20.34	26.45	12.98	2.09	0	0

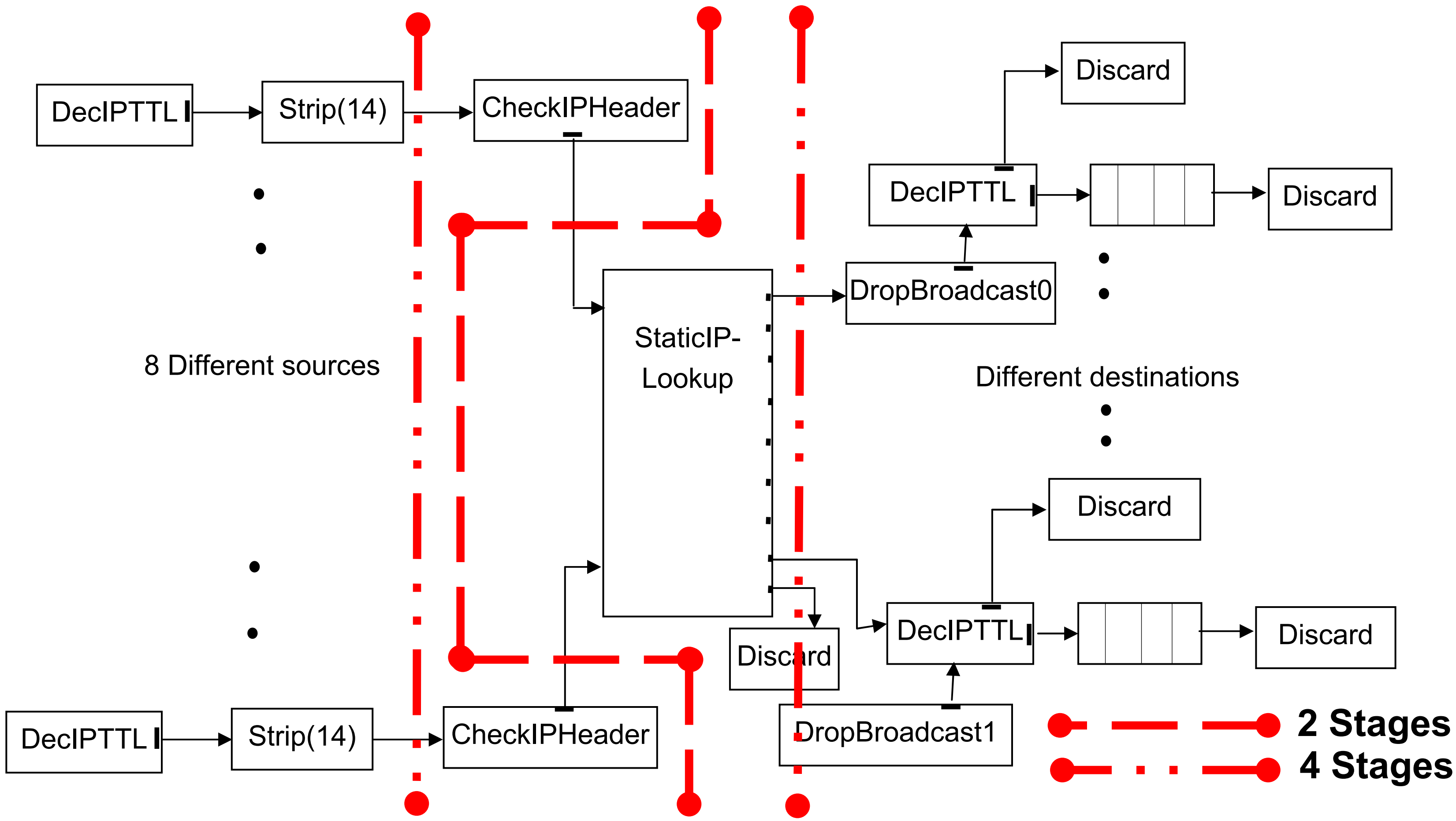
Prob. Distn. of IPv4 Router Stages

Stages	Mean (μ)	SD (σ)	Processing time threshold				
			μ	$\mu+\sigma$	$\mu+2\sigma$	$\mu+3\sigma$	$\mu+4\sigma$
Stage0	227.38	24.14	35.06	20.00	3.64	0.00	0.00
Stage1	691.18	30.48	23.19	14.29	1.86	0.08	0.00
Stage2	500.43	29.52	27.18	24.31	5.66	0.11	0.11
Stage3	314.72	20.33	27.78	23.14	7.14	0.28	0.00

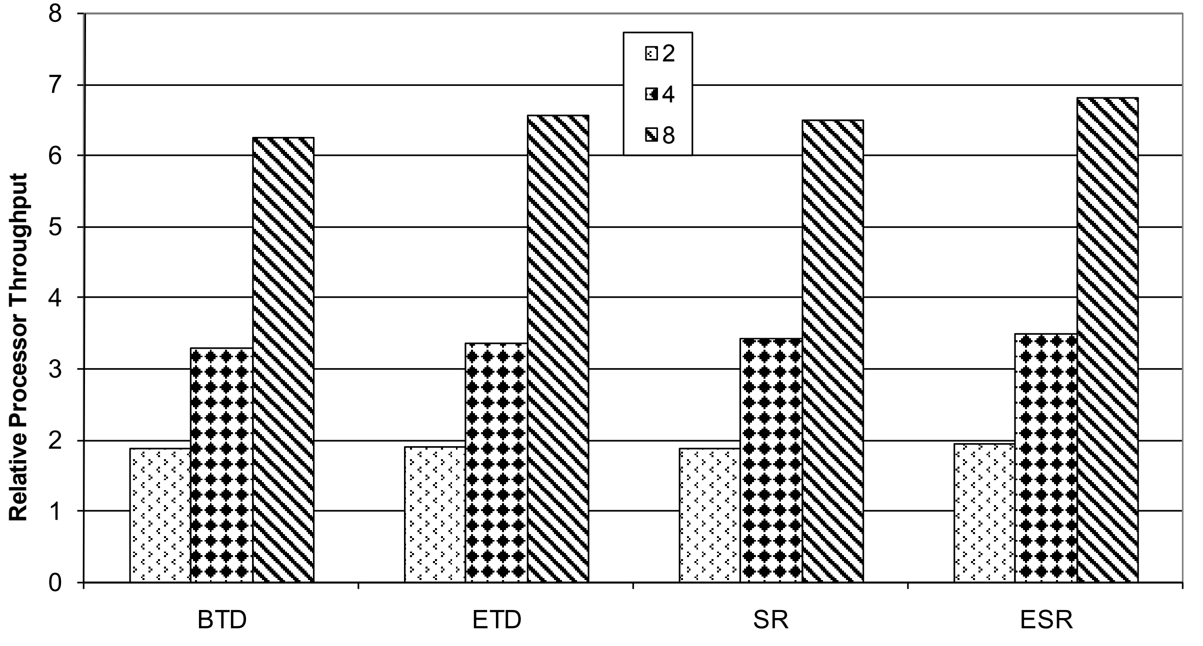
Optimized Strategies

- Base Task Distribution - BTD
  - Uniform task allocation depending on the mean execution time
- Extended Task Distribution - ETD
  - Slack $k\sigma$ added to estimated processing time
- Selective Replication - SR
  - Replicate modules parallelize packet processing
- Extended Selective Replication - ESR
  - Select elements with longer execution time

Module Distribution Illustration

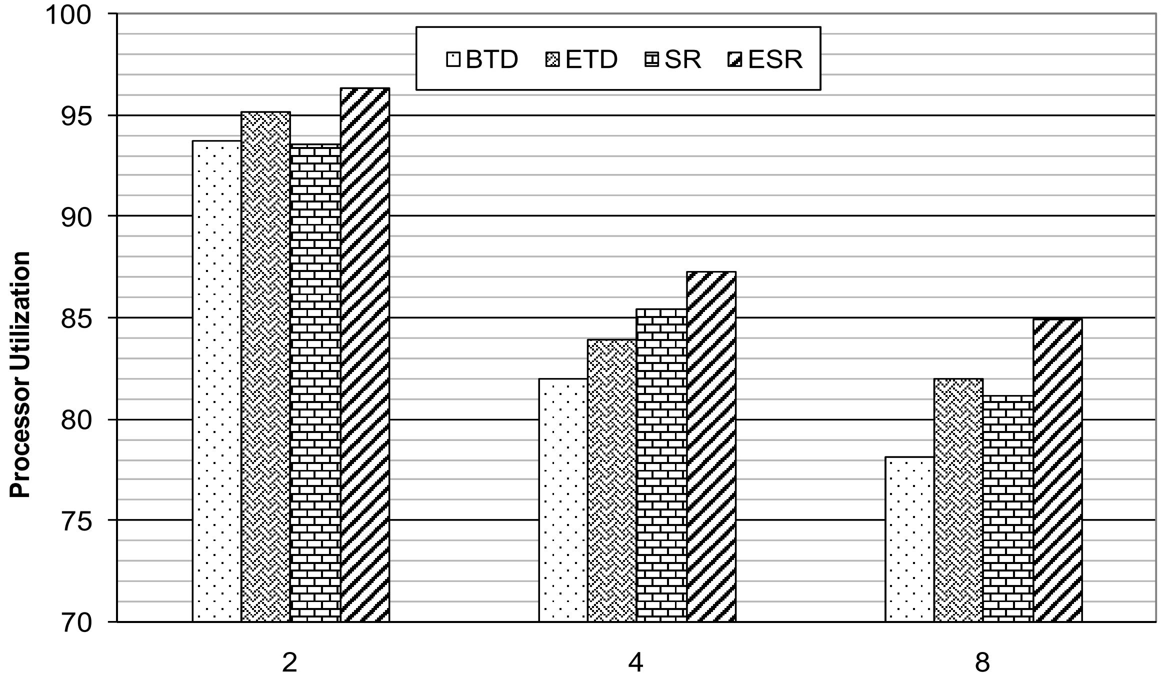


Relative Throughput Analysis



Processor throughput for DRR application

Resource Utilization Analysis



Resource utilization in DRR application

Contributions

- Analyzed modularity in networking applications using statistical methods
- Proposed intelligent task allocation based on variation in processing time
- Generic nature of the task allocation method applicable to CMP task distribution

Acknowledgements

- Click Development Group
- Anonymous reviewers

THANK YOU

yzh702@eecs.northwestern.edu