Handling of Packet Dependencies: A Critical Issue for Highly Parallel Network Processors

Stephen Melvin
Flowstorm, Inc.

Yale Patt
University of Texas, Austin

Motivation

- Embedded Systems for Network Data Path
  - Programmability is increasing (flexibility)
  - Tradeoffs with power and performance
- Application Characteristics:
  - Low data locality (state)
  - Large memory footprint (state)
  - Small working sets (per packet workload)
  - Lots of parallelism (but not unlimited)

Motivation (continued)

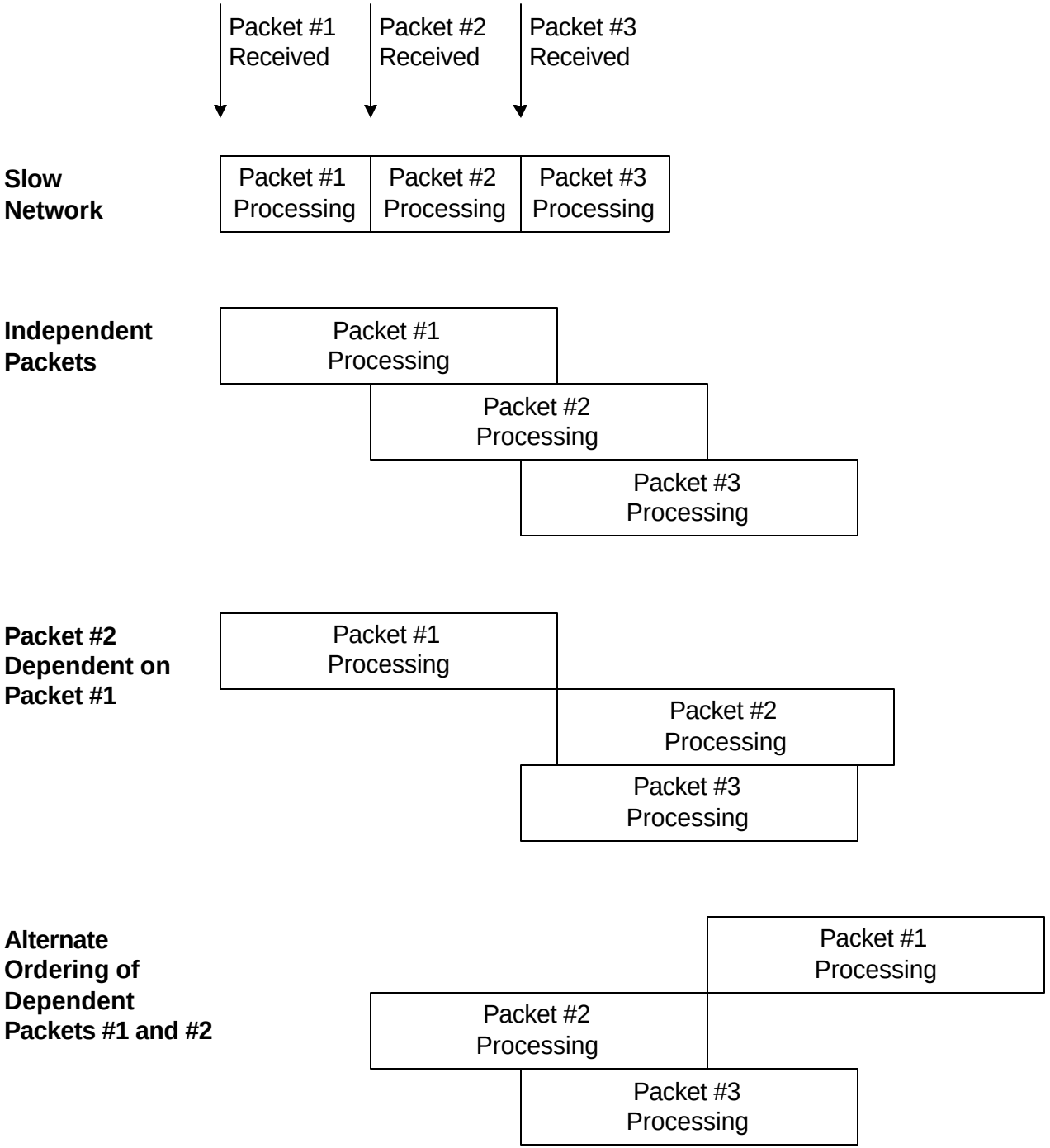
- Application characteristics are driving us to:
  - No data cache
  - Aggressive multithreading (100s to 1000s)
  - Sacrifice of per-thread performance
    - no speculation, single ported RFs, no ALU bypass
    - results in high power efficiency
- Looking forward, what fundamental problems arise?

  Limits to packet parallelism in a general purpose processor

Outline

- Introduction
- Simulation Study
  - Measurement of packet dependencies
  - Comparison of optimal vs. non-optimal solutions
- Hardware Model
  - Optimal solution with no software changes
  - Advanced features (conflict prediction, etc.)
- Future Work

What are Packet Dependencies?



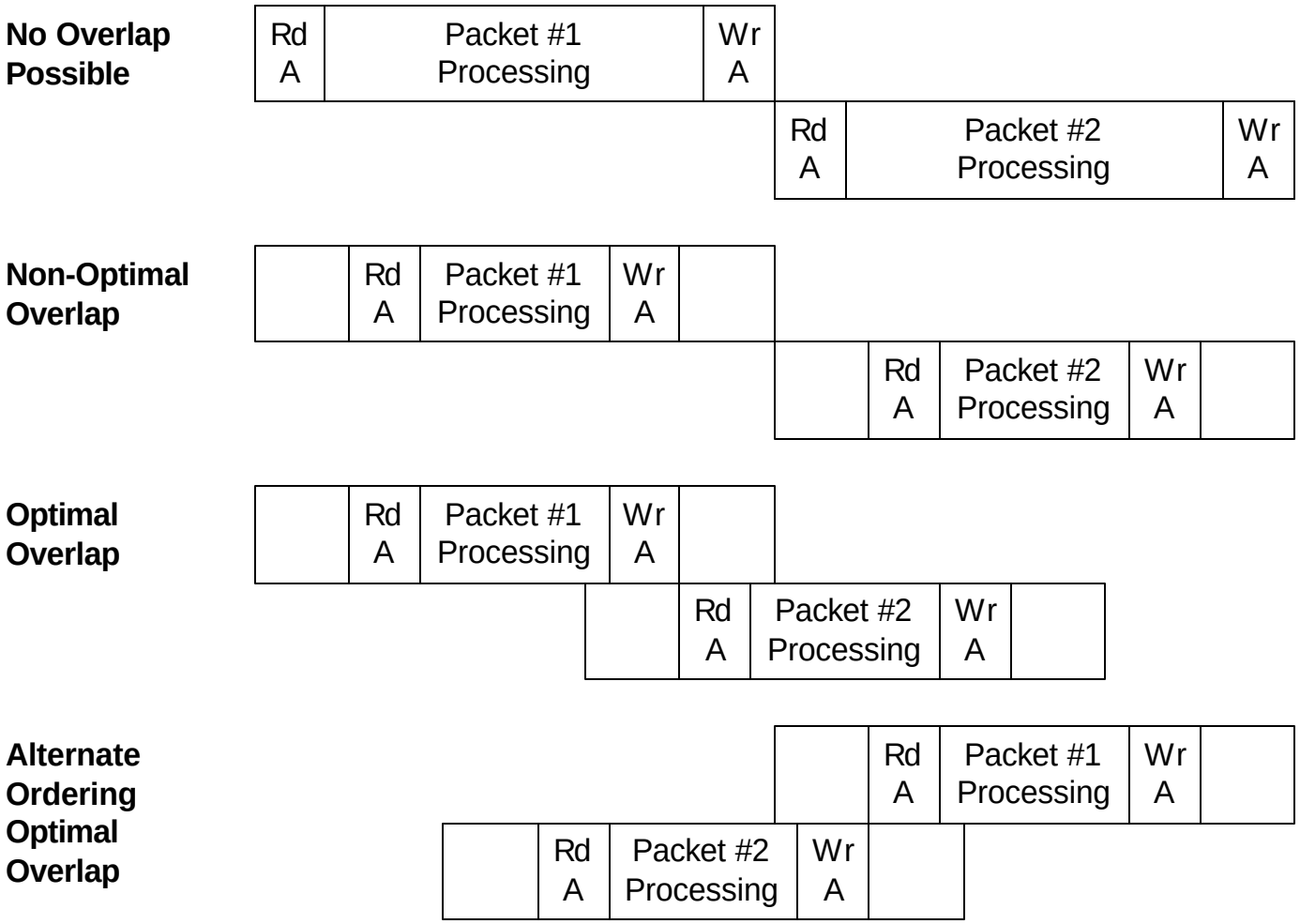
Why Are Packet Dependencies Important?

- Current solutions:
  - Non-parallel processing
  - Packets are all independent
  - Fixed functionality (rigid solutions work OK)
- But in highly parallel general purpose packet processors:

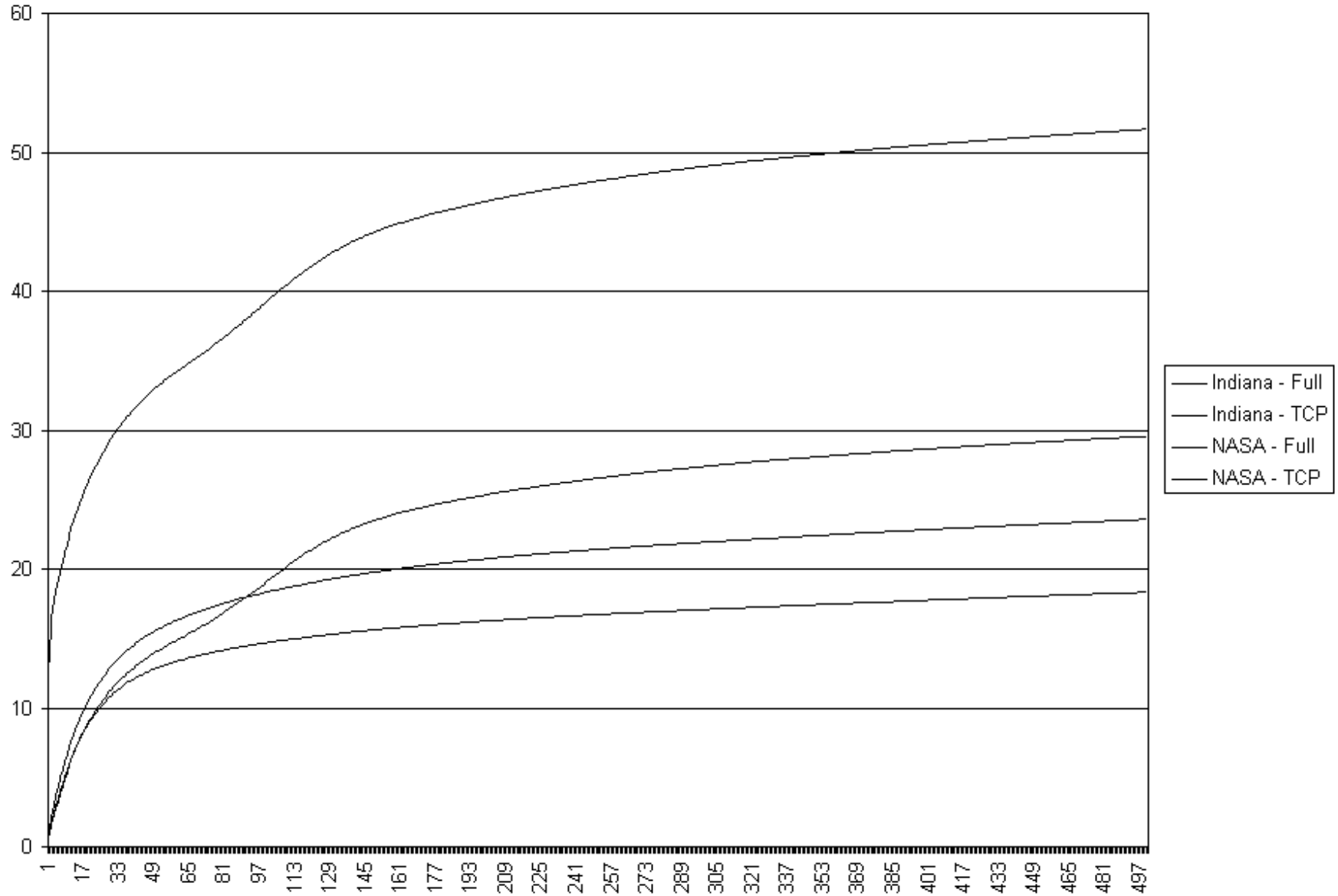
  Not easy to predict or characterize dependencies
- Key question:

  How important is optimal processing?

What is Optimal Handling of Dependencies?



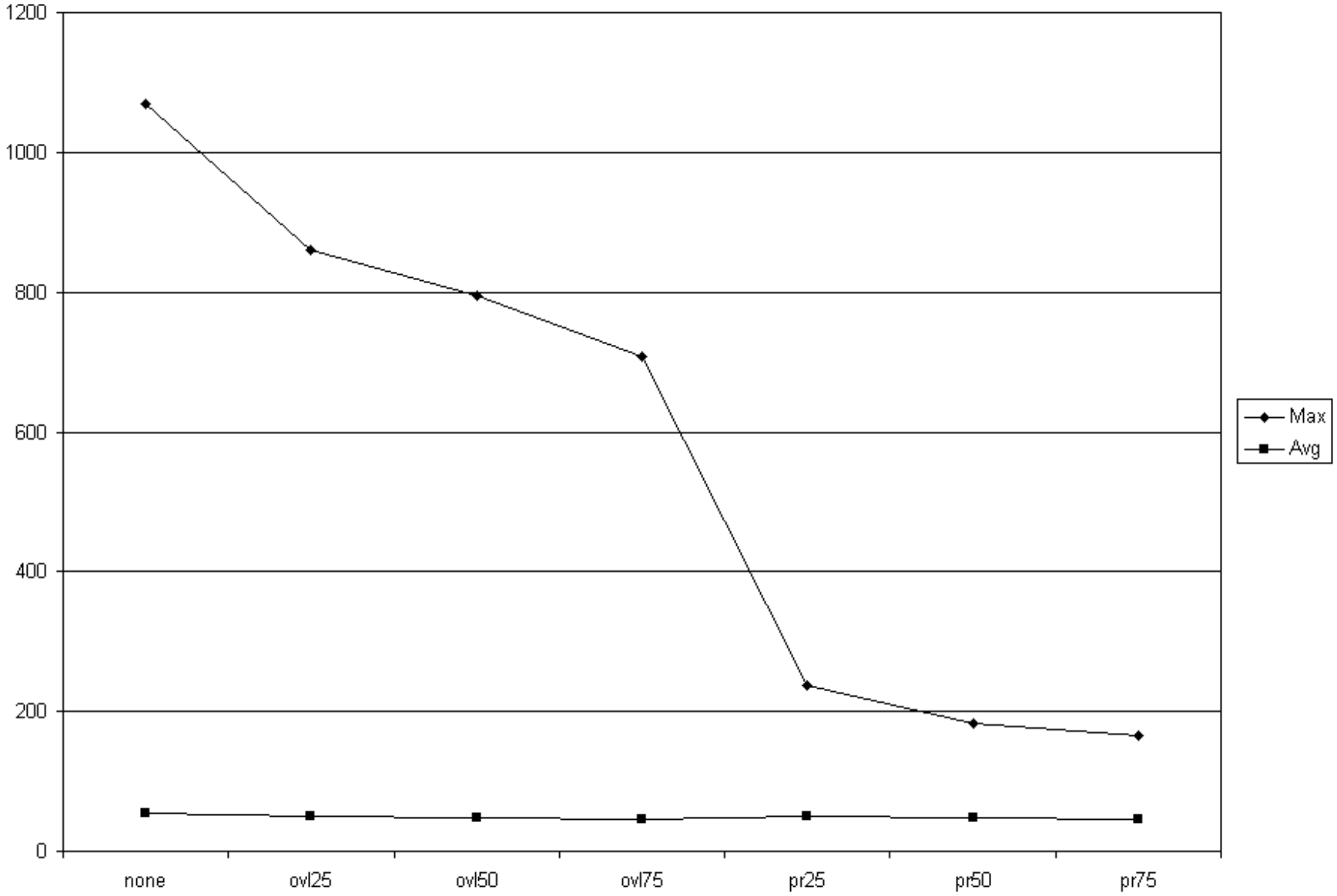
Dependency Probability vs. Window Size



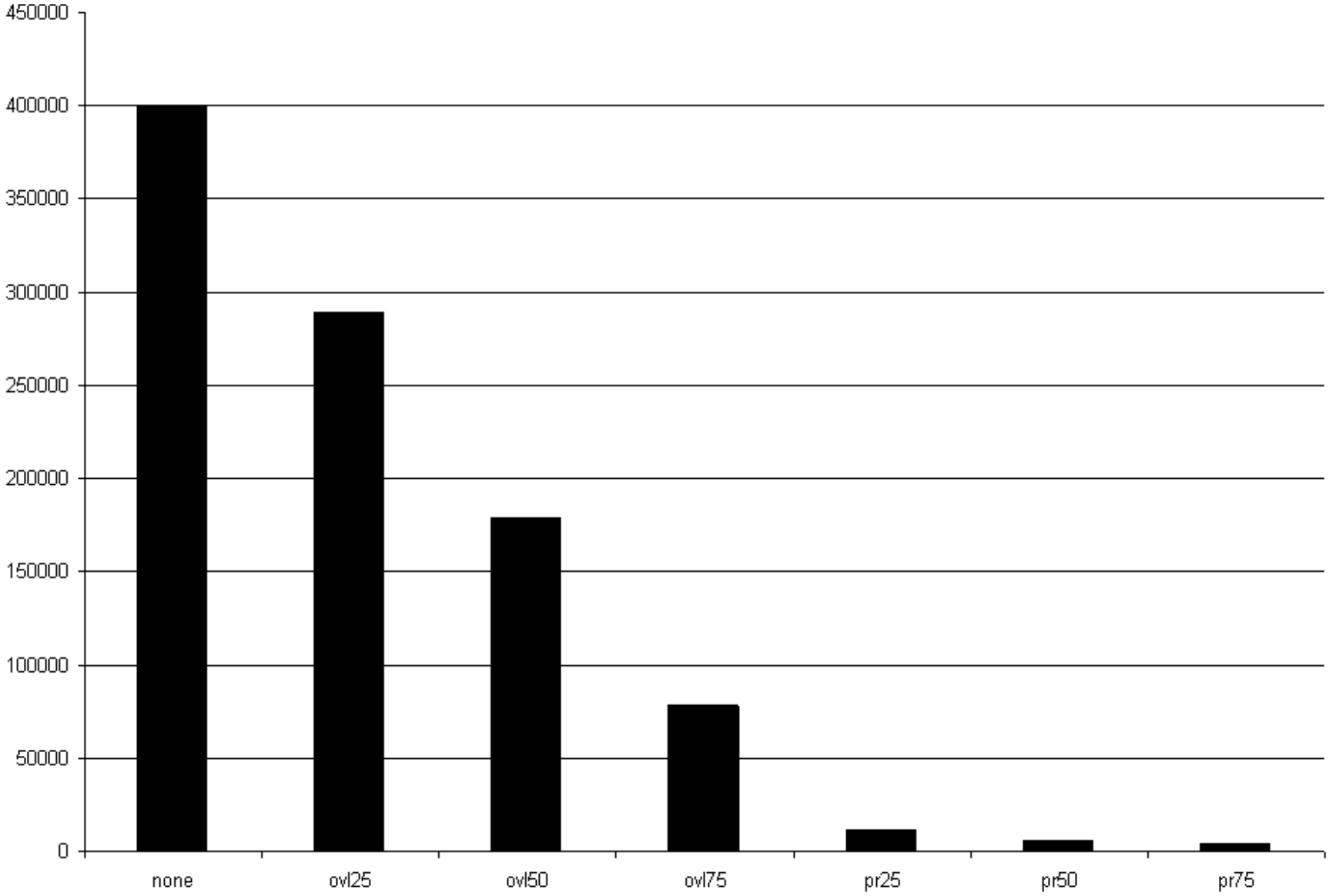
Packet Dependency Models

Model	Description
none	packets fully dependent, no overlap possible
ovl25	¼ of workload can be overlapped
ovl50	½ of workload can be overlapped
ovl75	¾ of workload can be overlapped
pr25	25% probability of independence
pr50	50% probability of independence
pr75	75% probability of independence

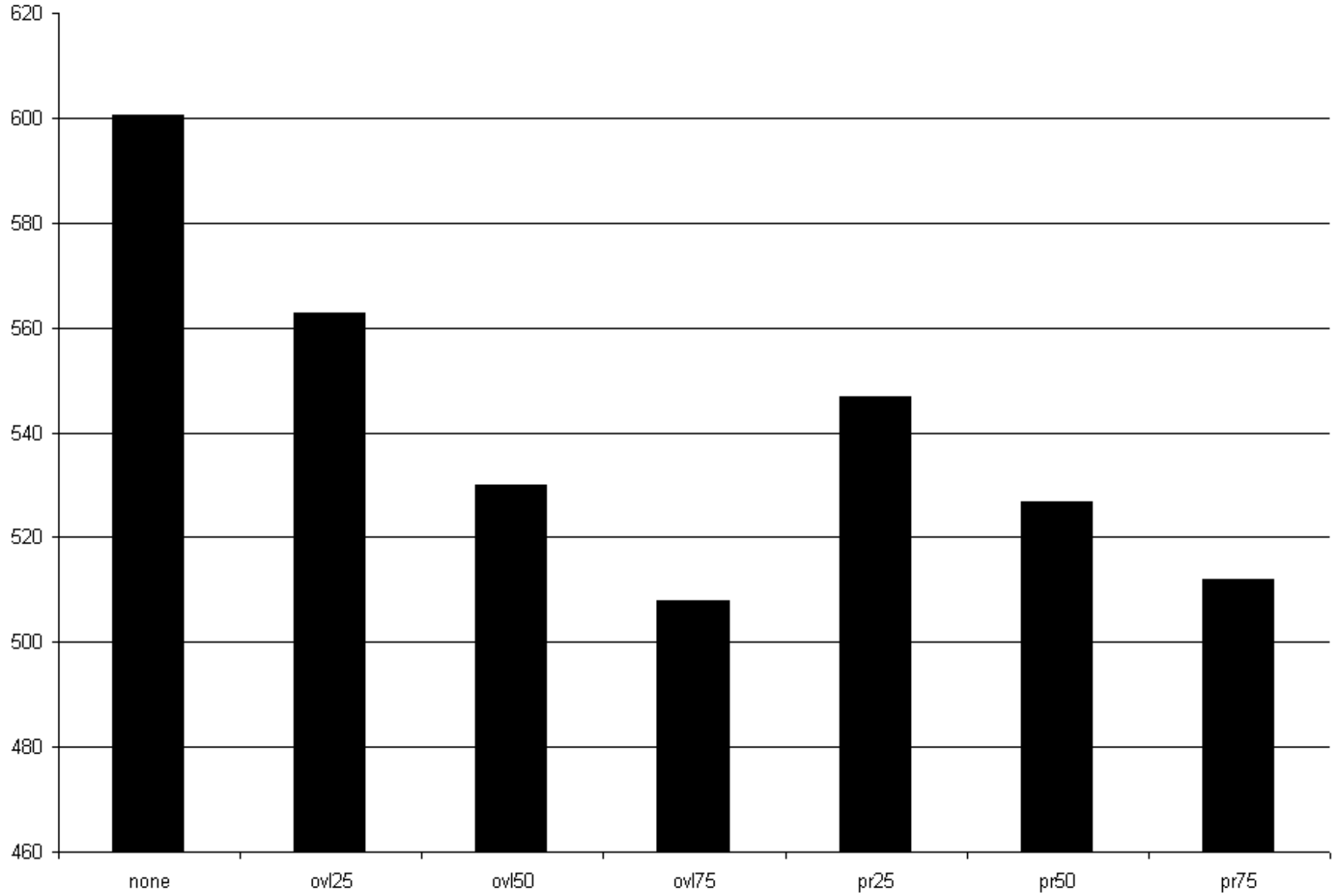
Number of Packets Active (Processing or Waiting)



Maximum Packet Latency



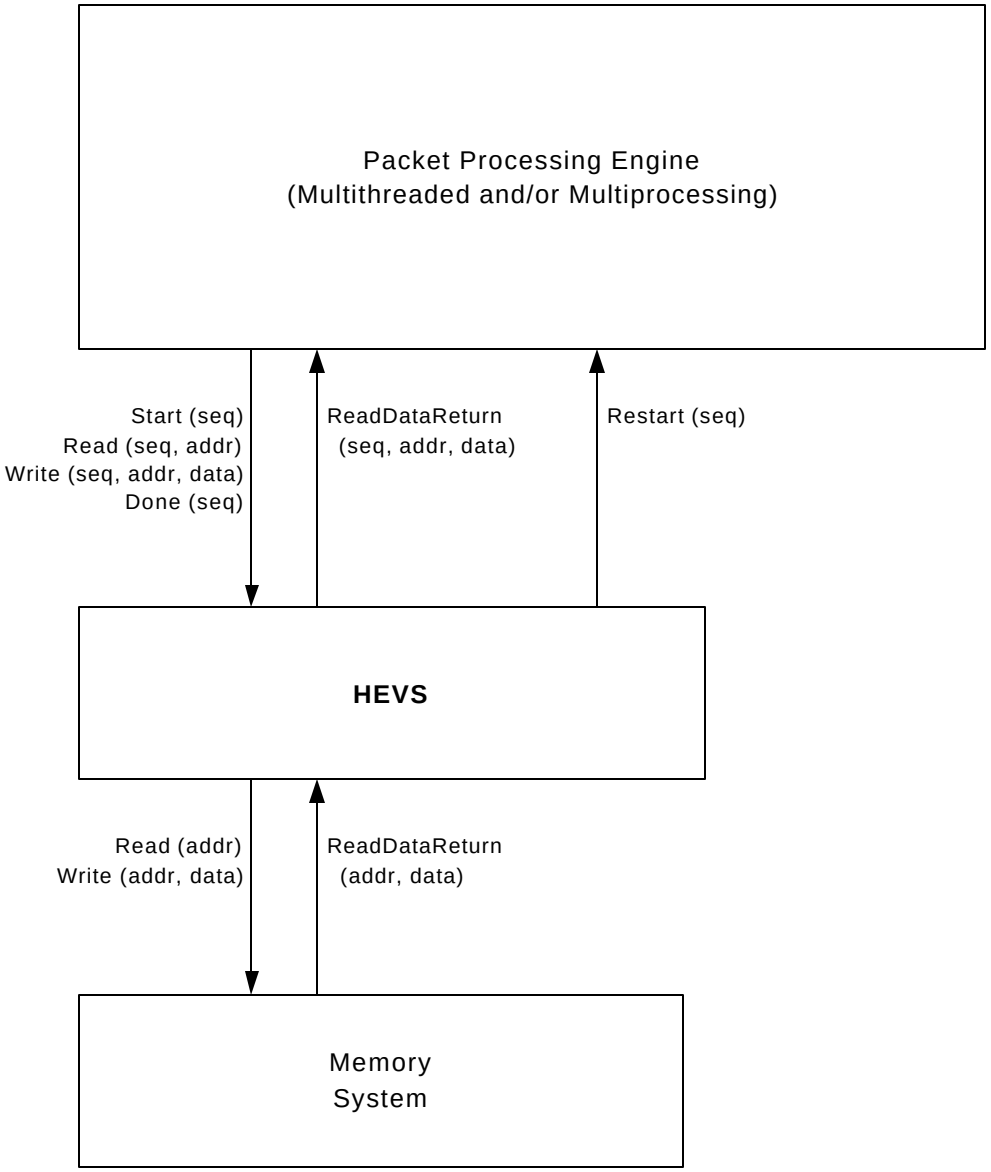
Average Packet Latency



Hardware Model

- Hardware Model for Optimal Processing
  - Apply sequence numbers to each access
  - Allow restart in case of conflict
- Advantages
  - Only restarts when actual conflict occurs
  - No changes to software are needed

System Overview



Read Table

	Seq.	Addr.
p:	1	A
q:	2	B
s:	3	B
t:	2	A

Write Table

	Seq.	Addr.	Data	Depend
r:	2	B	X	3
u:	1	A	X	(null)

Time Sequence:

1. Packet #1 reads location A
   - Entry **p:** created in Read Table
   - Write Table is searched, no matches found so memory read is performed

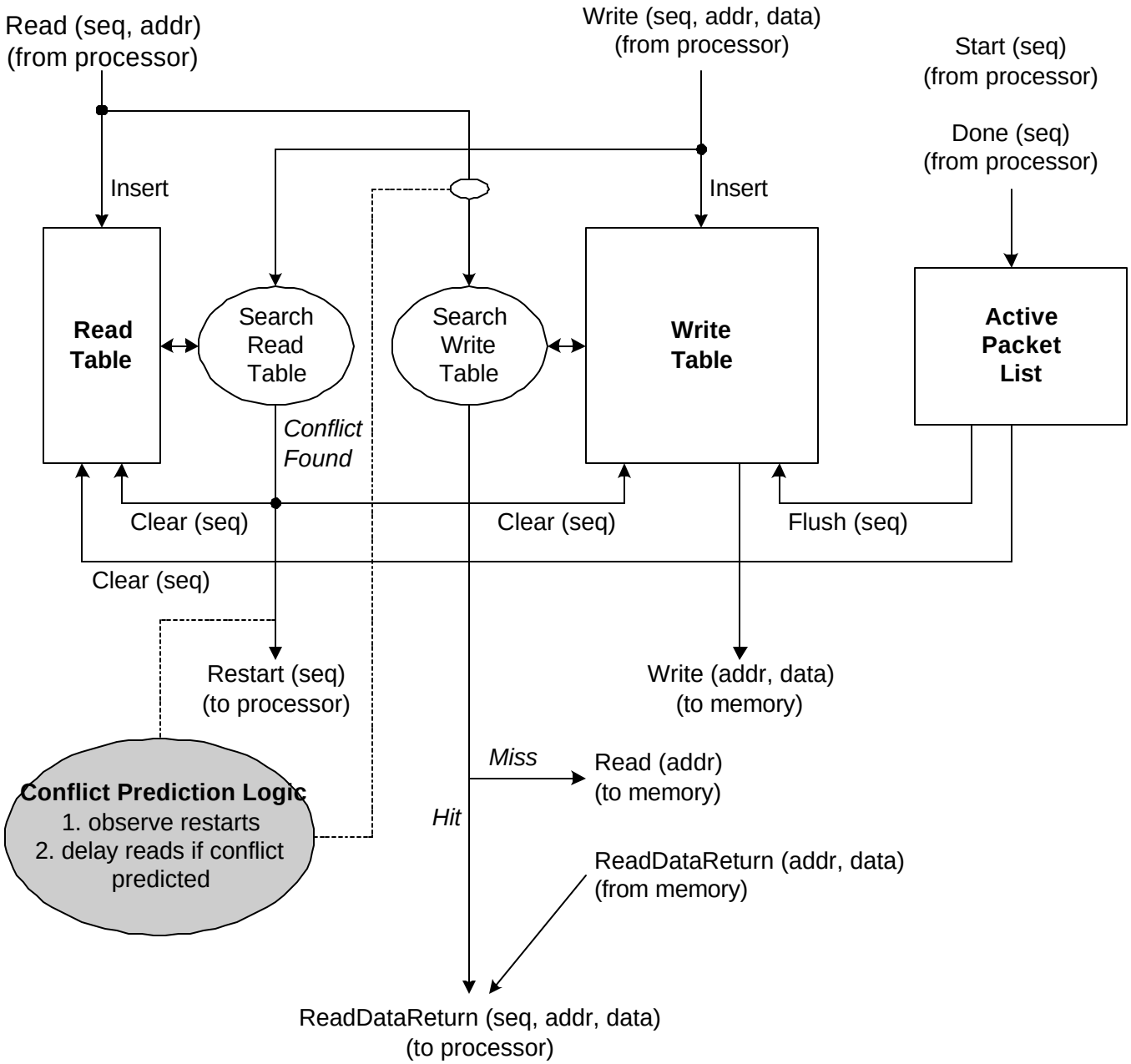
2. Packet #2 reads location B
   - Entry **q:** created in Read Table
   - Write Table is searched, no matches found so memory read is performed

3. Packet #2 writes location B
   - Entry **r:** created in Write Table
   - Read Table is searched, no conflicts found

4. Packet #3 reads location B
   - Entry **s:** created in Read Table
   - Write Table is searched, entry r: found, data X forwarded and dependency list updated

5. Packet #2 reads location A
   - Entry **t:** created in Read Table
   - Write Table is searched, no matches found so memory read is performed

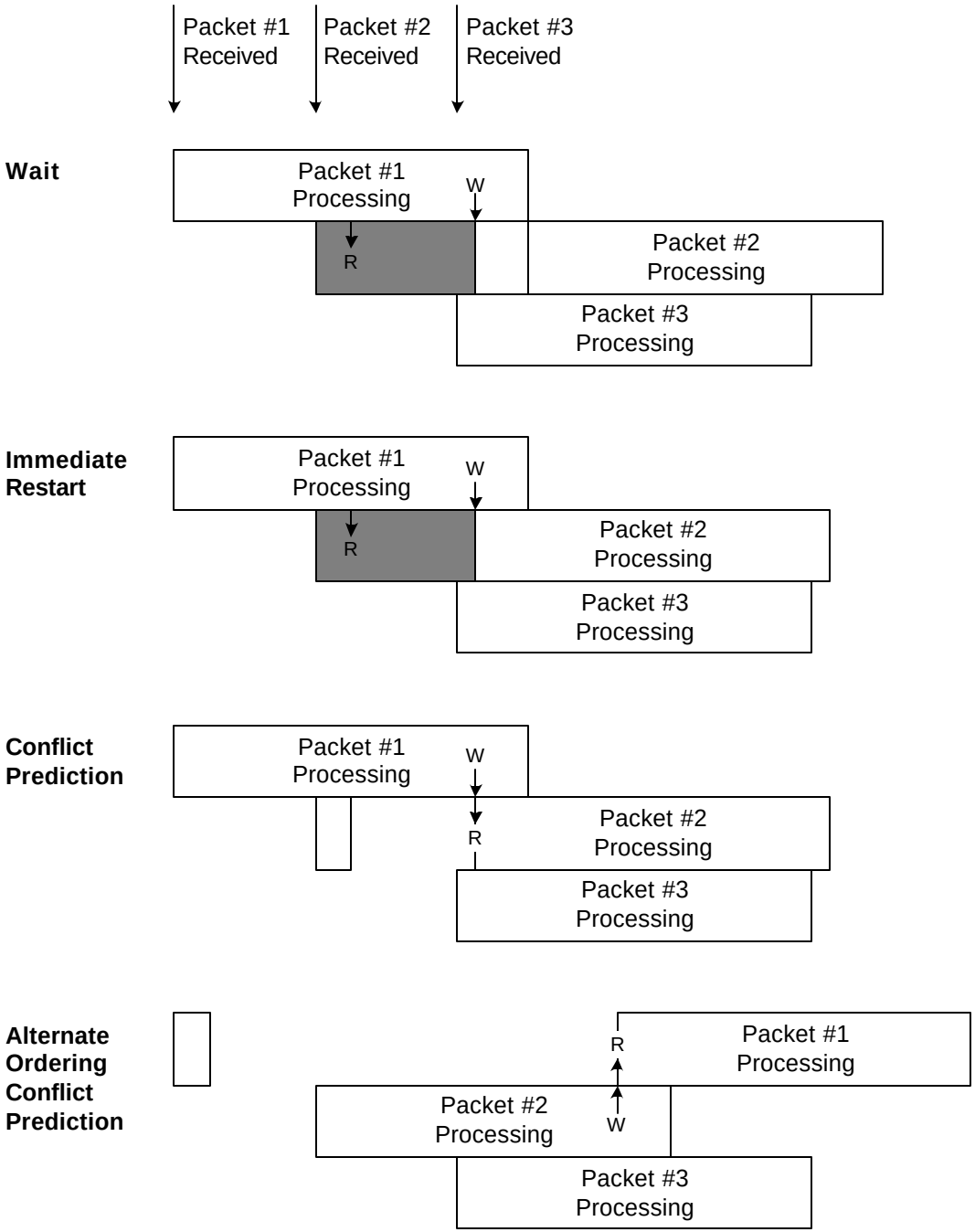
6. Packet #1 writes location A
   - Entry **u:** created in Write Table
   - Read Table is searched for newer sequence read, entry **t:** is found
     - Conflict is signaled to processor, Packet #2 is restarted
     - Entry **q:** and all other sequence 2 entries are deleted
     - Deletion of entry **r:** triggers Packet #3 restart signaled



Cost Reduction Possibilities

- Combine write entries in a single line
  - from the same sequence number
  - valid bit vector
  - approaches cost of ordinary write buffer
- Reads blocked into a single entry
  - from the same sequence number
- Pipelining expensive operations

Conflict Detection Processing Examples



Future Work

- More Simulation Studies
  - Longer and higher speed traces
  - Real workloads
- Develop RTL Models for Hardware Solutions
  - Explore area/time/power tradeoffs
  - Simulation on real workloads and traces