Weaving Relations for Cache Performance

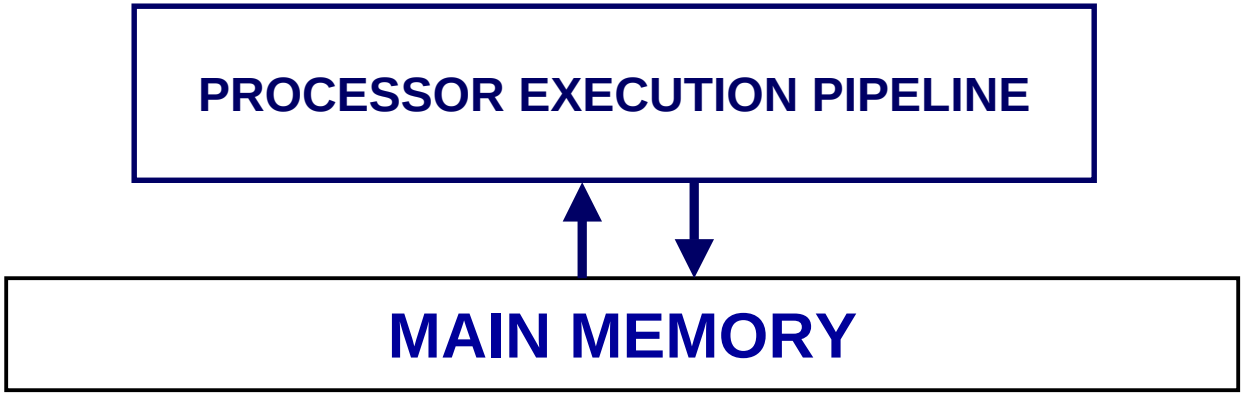
Anastassia Ailamaki

Carnegie Mellon

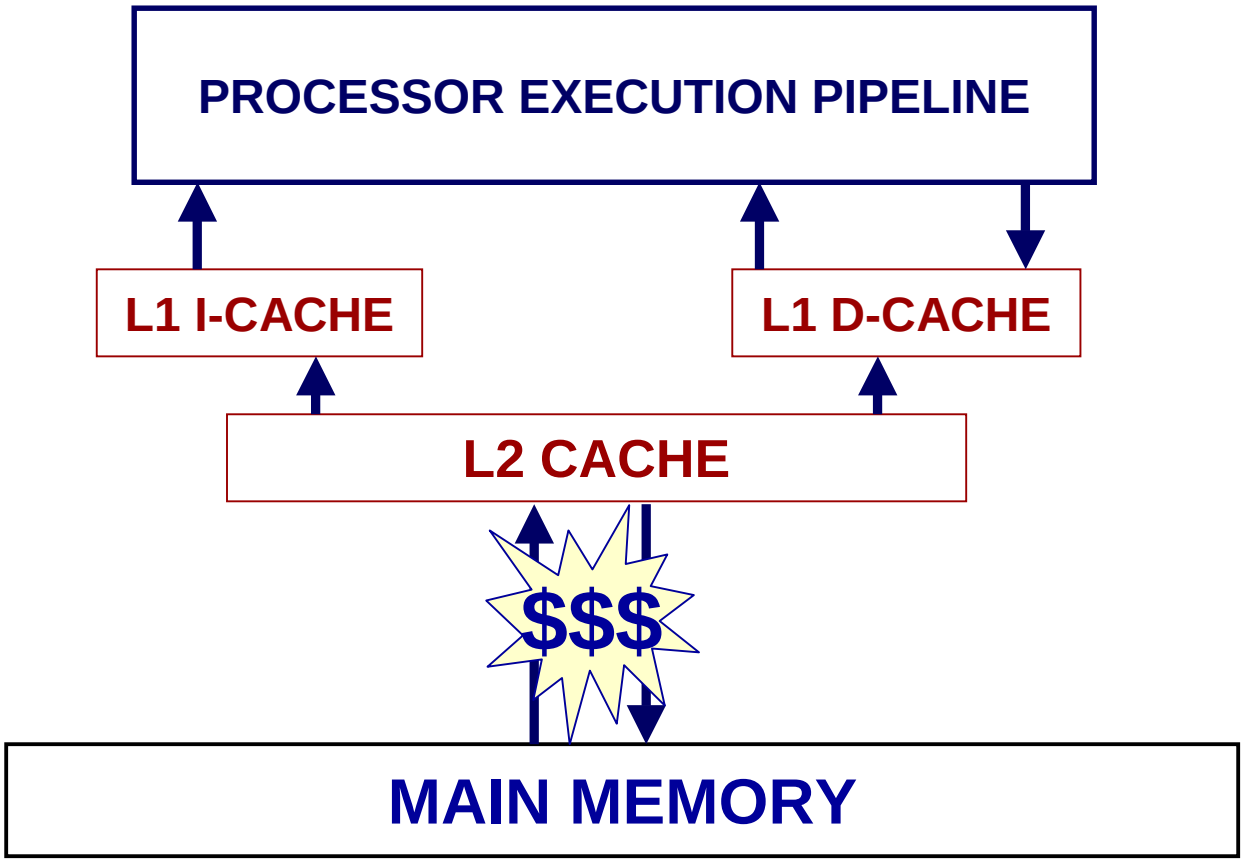
David DeWitt, Mark Hill, and Marios Skounakis

University of Wisconsin-Madison

Memory Hierarchies



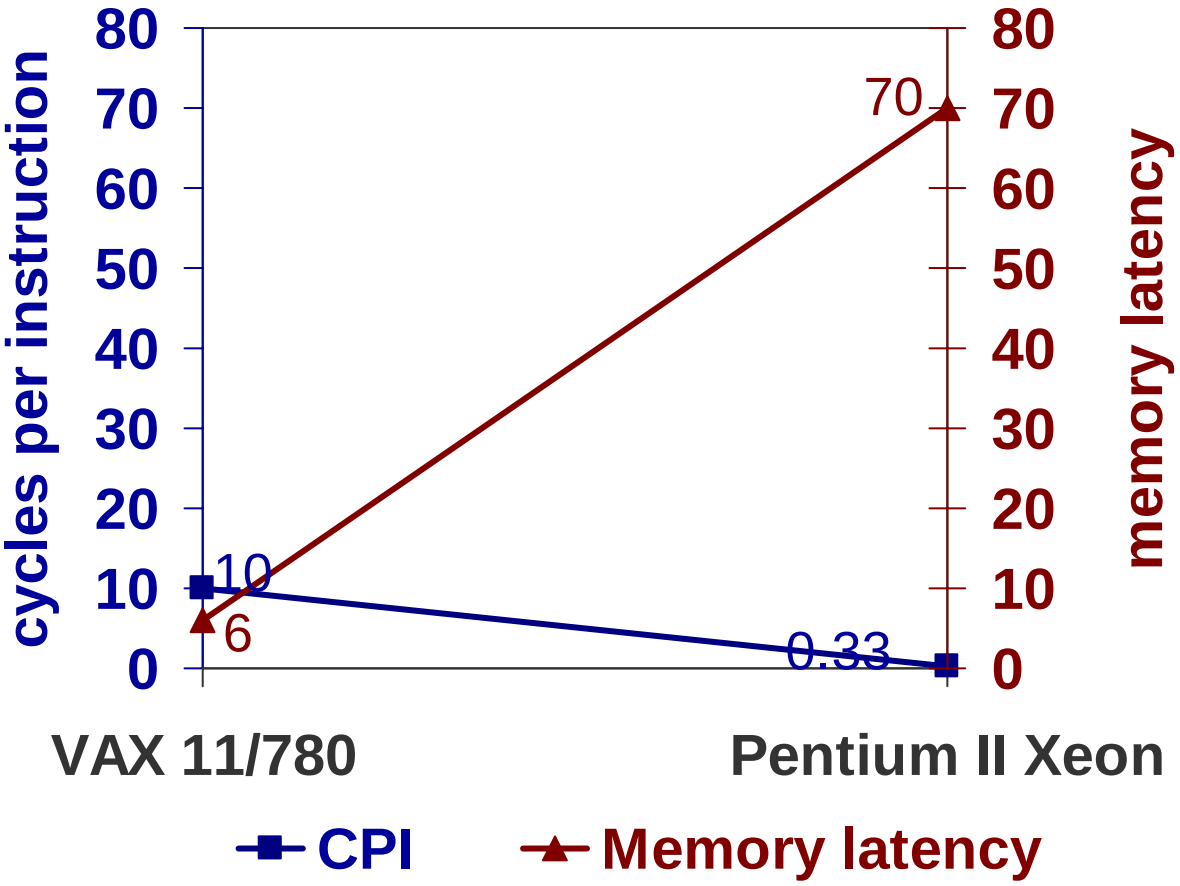
Memory Hierarchies



Cache misses are extremely expensive

Carnegie Mellon

Processor/Memory Speed Gap



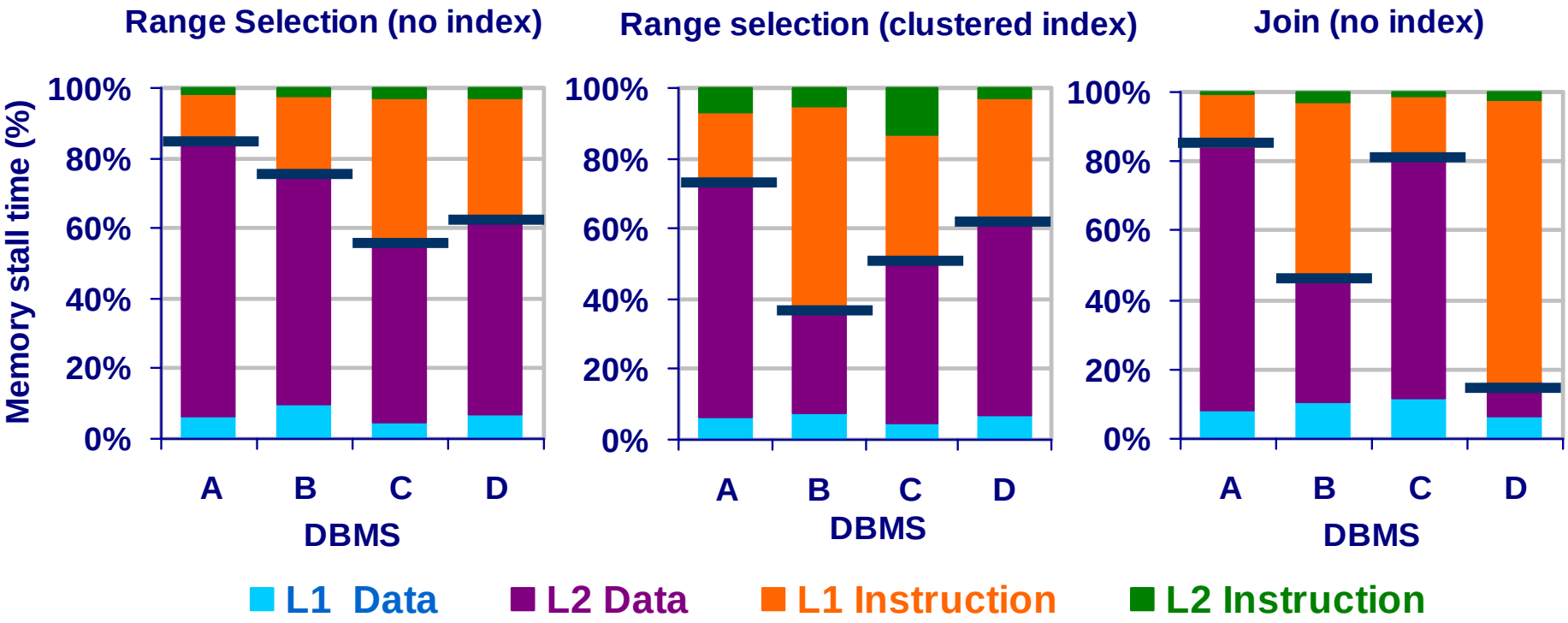
1 access to memory ≅ 1000 instruction opportunities

Carnegie Mellon

Anastassia Ailamaki, VLDB 2001

Breakdown of Memory Delays

- PII Xeon running NT 4.0, 4 commercial DBMSs: A,B,C,D
- Memory-related delays: 40%-80% of execution time



Data accesses on caches: 19%-86% of memory stalls

Carnegie Mellon

Data Placement on Disk Pages

- *Slotted Pages:* Used by all commercial DBMSs
  - ✓ Store table records sequentially
  - ☺ Intra-record locality (attributes of record *r* together)
  - ☹ Doesn't work well on today's memory hierarchies

- Alternative: Vertical partitioning [Copeland'85]
  - ✓ Store *n*-attribute table as *n* single-attribute tables
  - ☺ Inter-record locality, saves unnecessary I/O
  - ☹ Destroys intra-record locality => expensive to reconstruct record

- **Contribution: Partition Attributes Across**
  - ☺ **… have the cake and eat it, too**

Inter-record locality + low record reconstruction cost

Carnegie Mellon

Outline

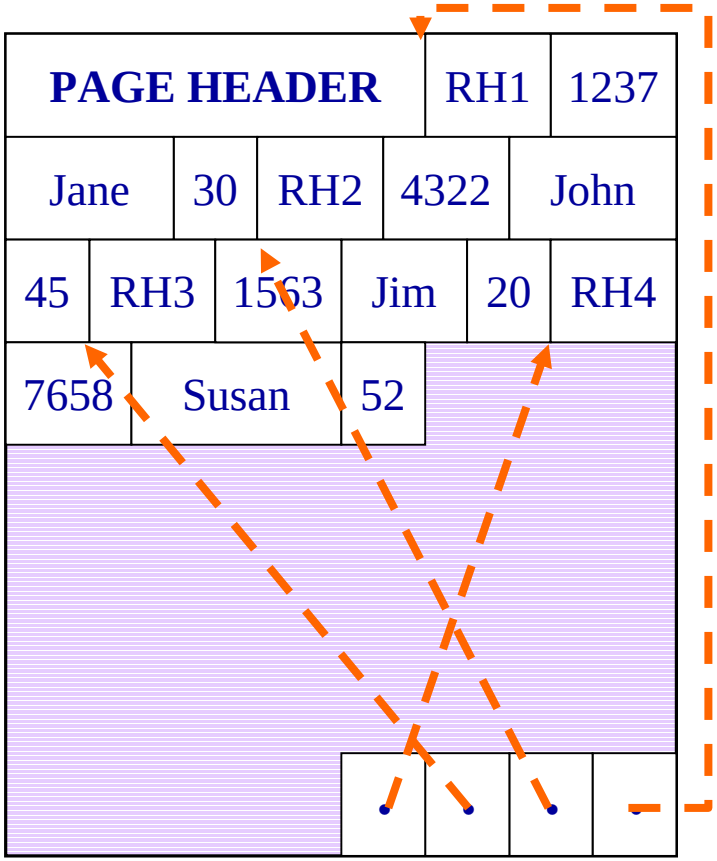
- The memory/processor speed gap
- **What's wrong with slotted pages?**
- Partition Attributes Across (PAX)
- Performance results
- Summary

Current Scheme: Slotted Pages

Formal name: NSM (N-ary Storage Model)

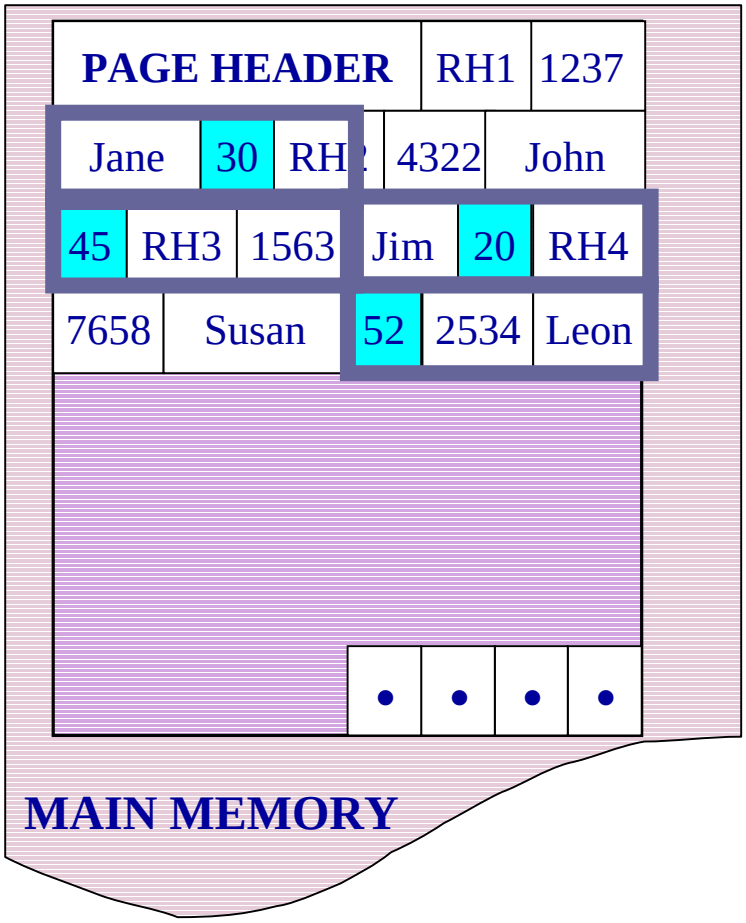
R

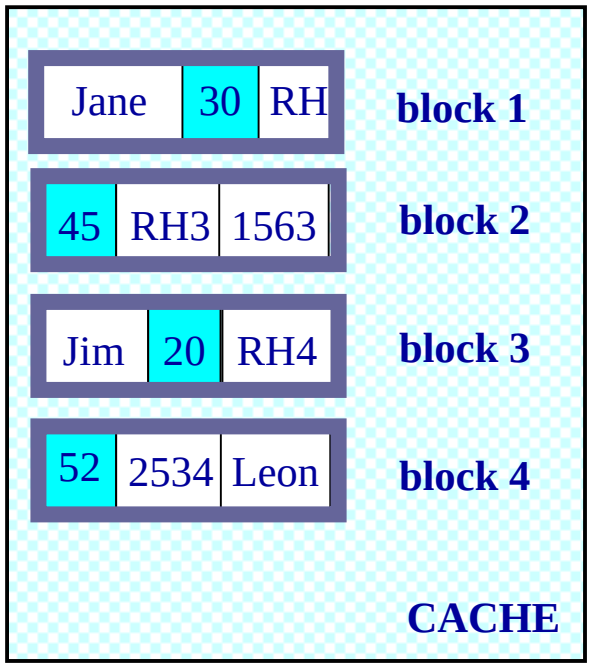
RID	SSN	Name	Age
1	1237	Jane	30
2	4322	John	45
3	1563	Jim	20
4	7658	Susan	52
5	2534	Leon	43
6	8791	Dan	37



- Records are stored sequentially
- Offsets to start of each record at end of page

Predicate Evaluation using NSM






```
select name
from  R
where age > 50
```

NSM pushes non-referenced data to the cache

Need New Data Page Layout

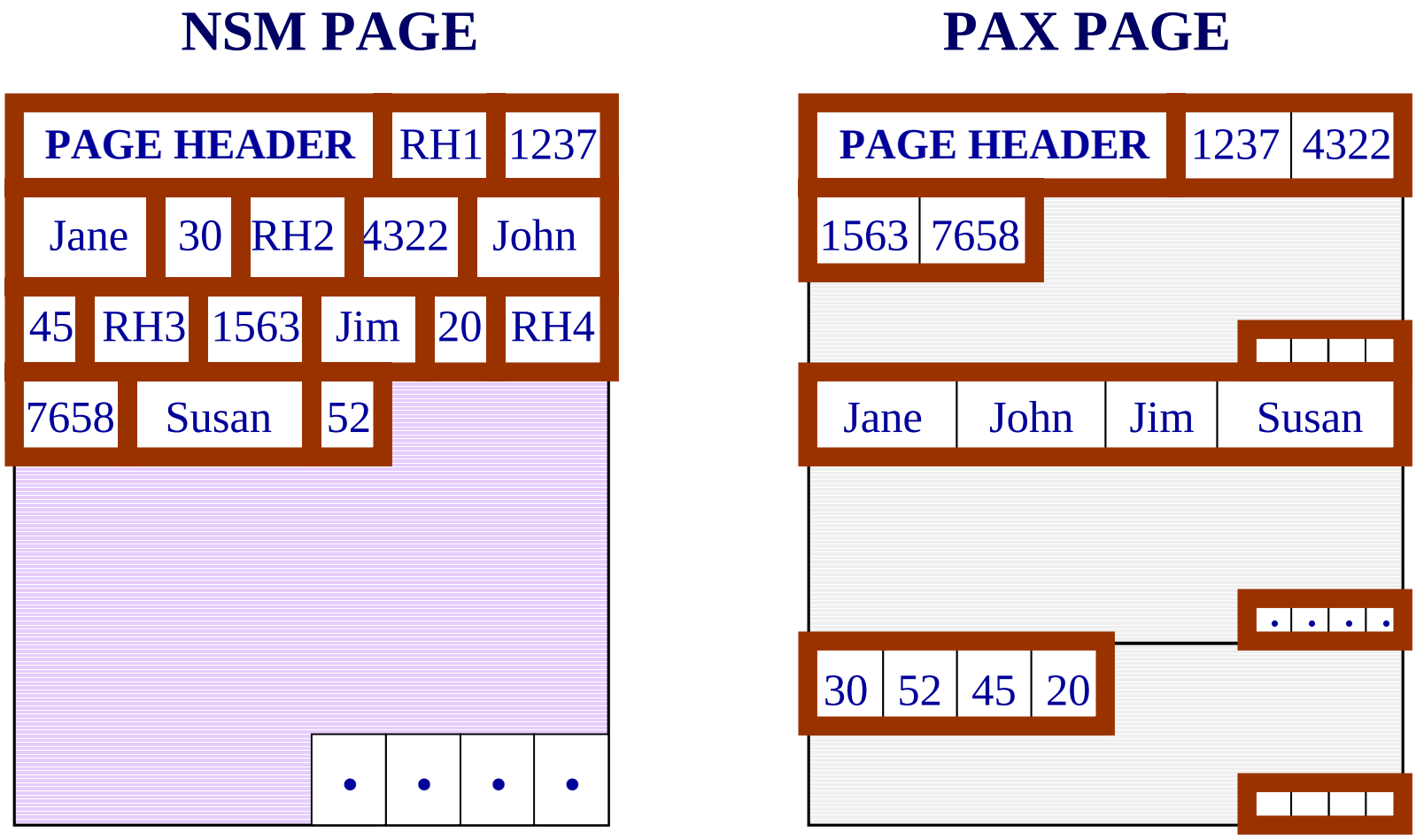
- Eliminates unnecessary memory accesses
- Improves inter-record locality
- Keeps a record's fields together
- Does not affect I/O performance

and, most importantly, is…

low-implementation-cost, high-impact

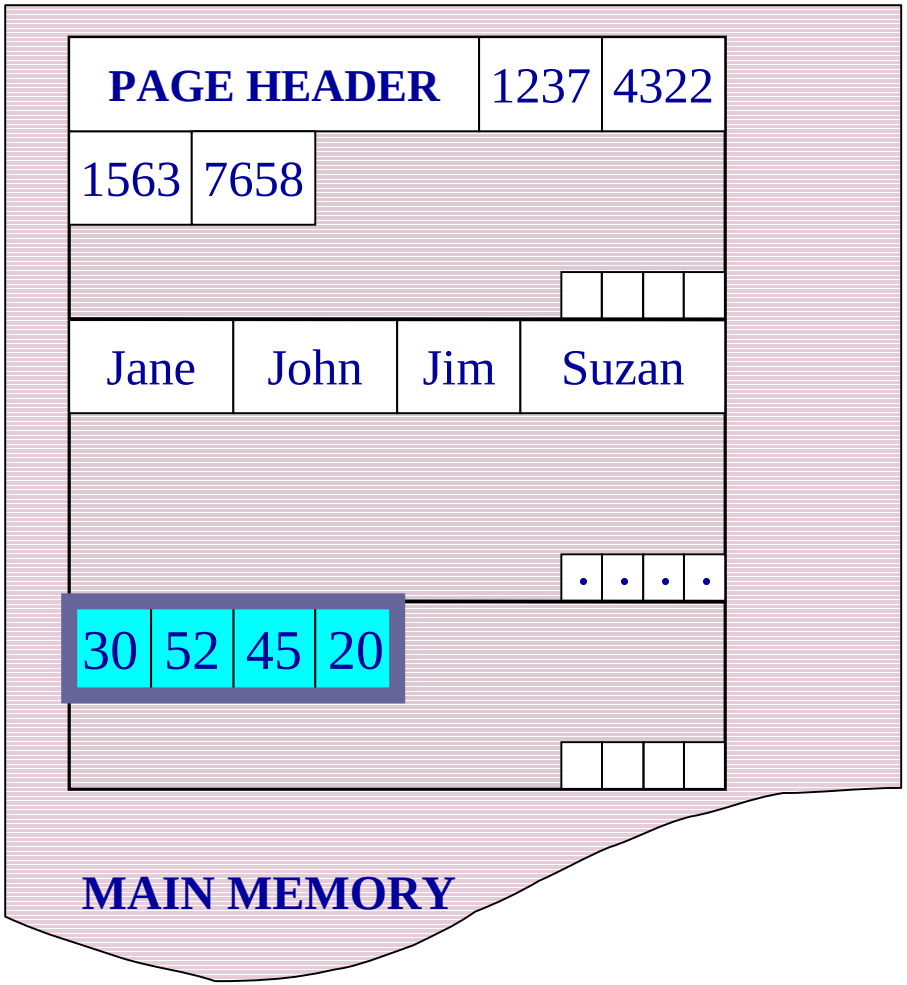
Carnegie Mellon

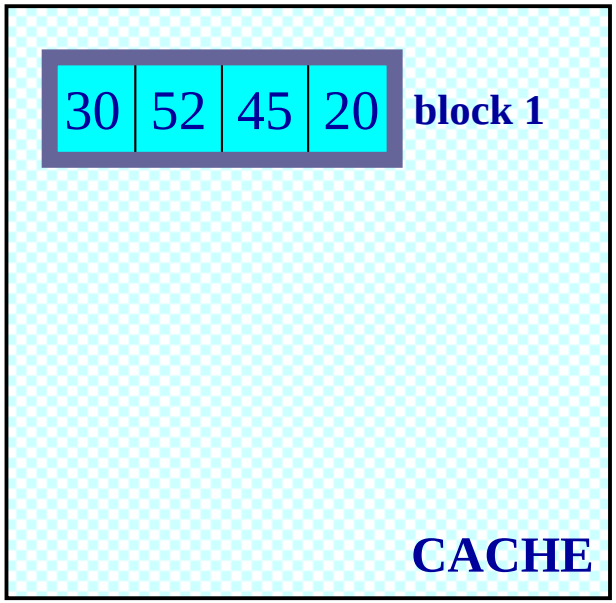
Partition Attributes Across (PAX)



Partition data *within* the page for spatial locality

Predicate Evaluation using PAX

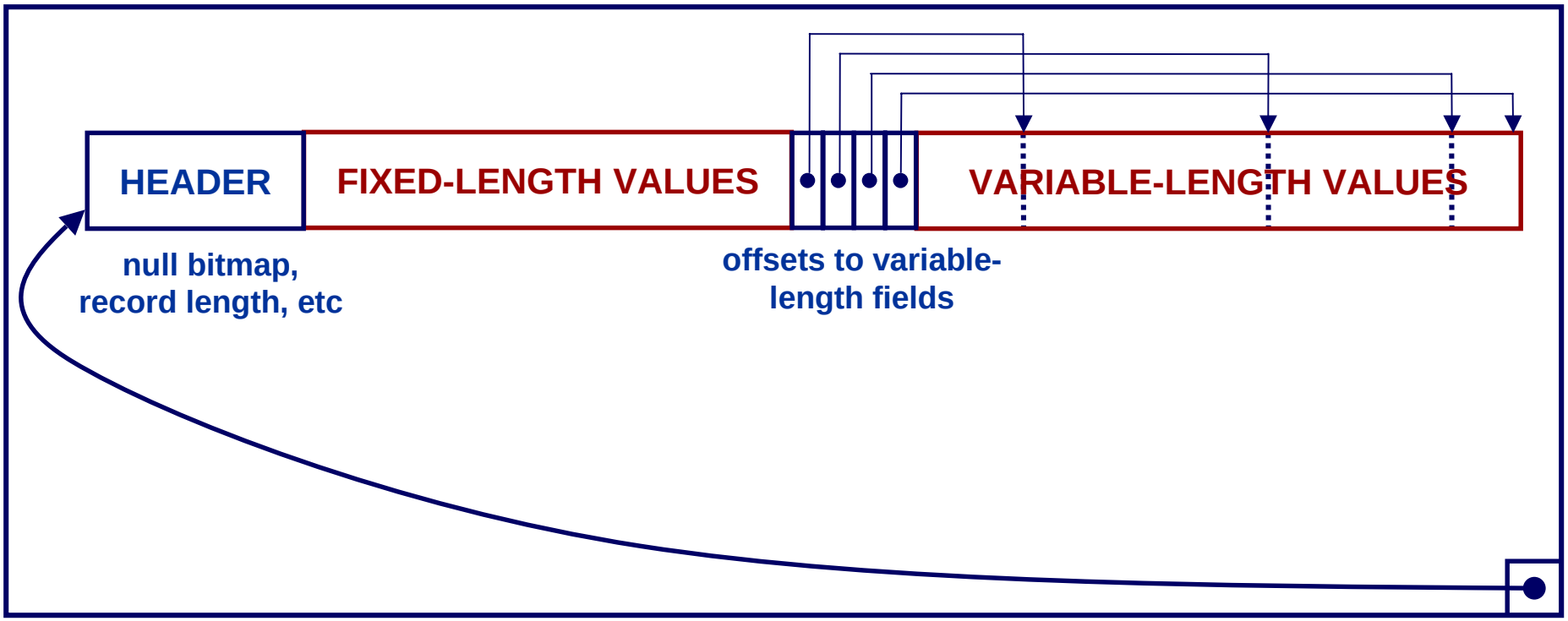





```
select name
from  R
where age > 50
```

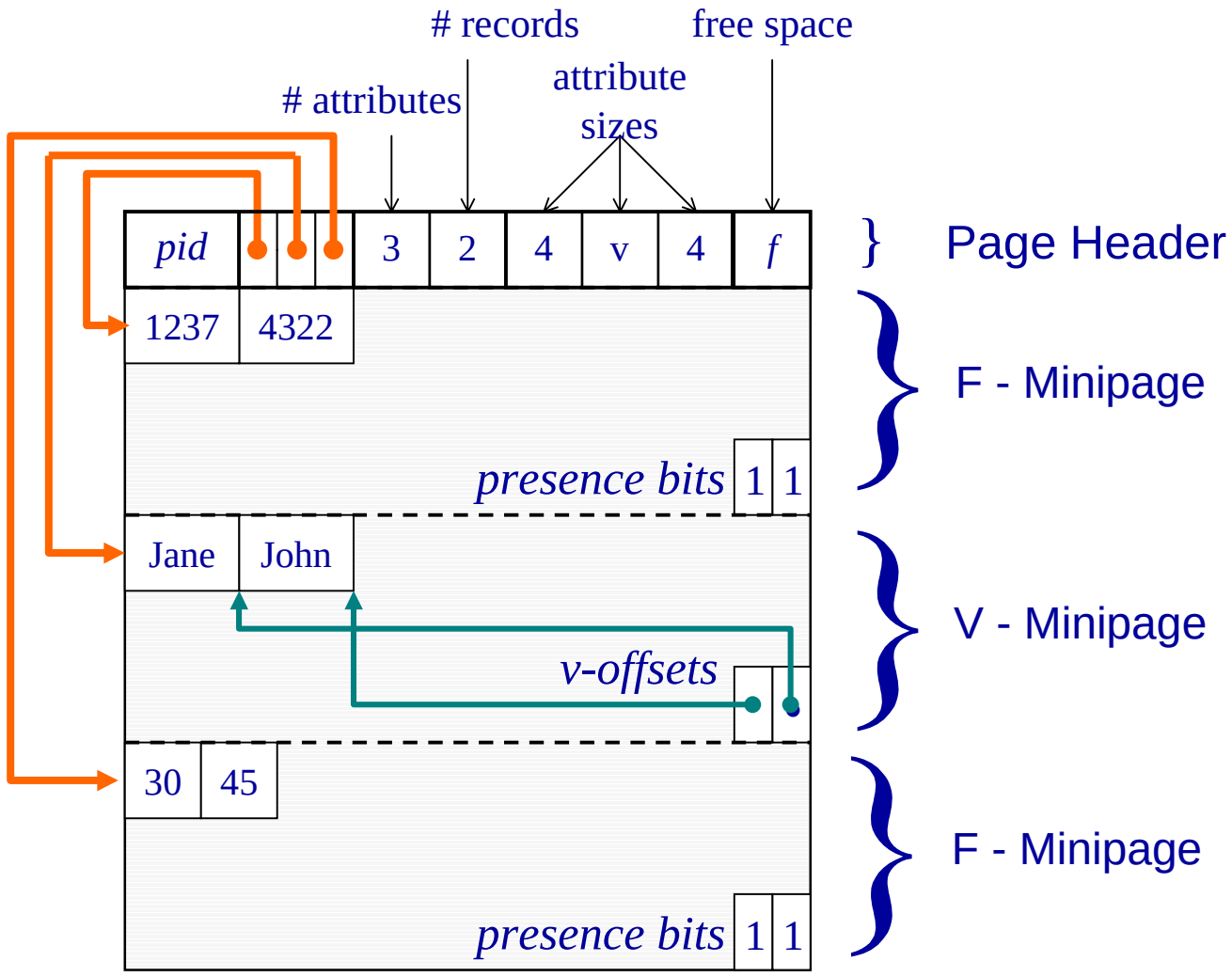
Fewer cache misses, low reconstruction cost

A Real NSM Record



NSM: All fields of record stored together + slots

PAX: Detailed Design



PAX: Group fields + amortizes record headers

Outline

- The memory/processor speed gap
- What's wrong with slotted pages?
- Partition Attributes Across (PAX)
- **Performance results**
- Summary

Sanity Check: Basic Evaluation

- Main-memory resident R, numeric fields
- Query:

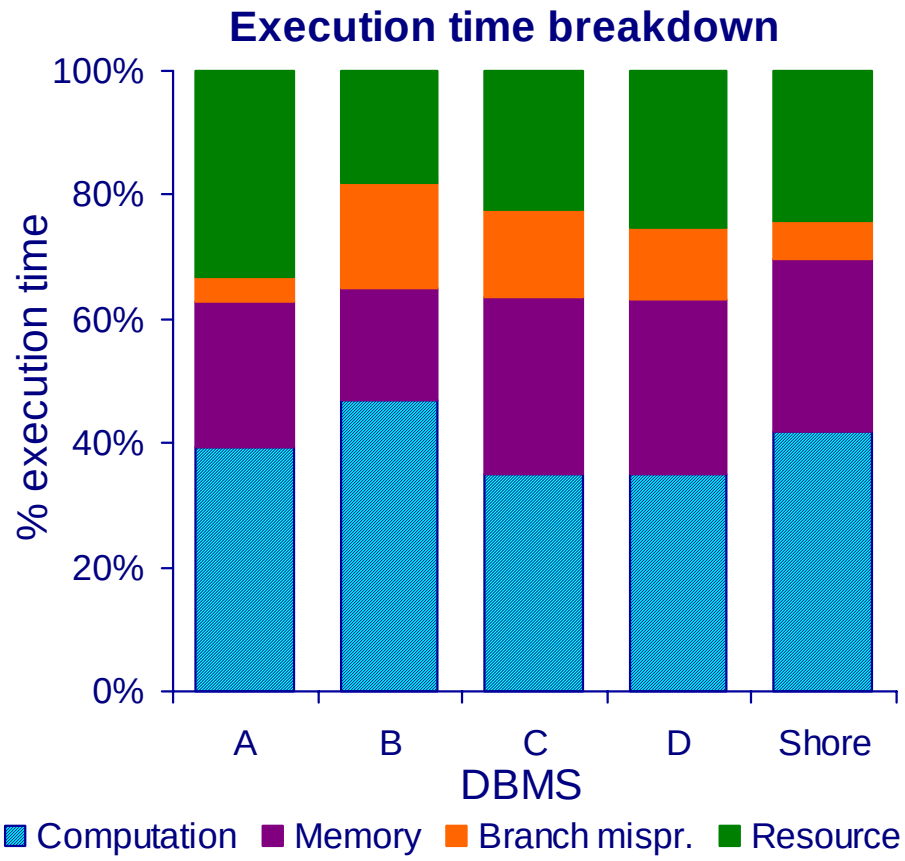
  select avg (a_i)

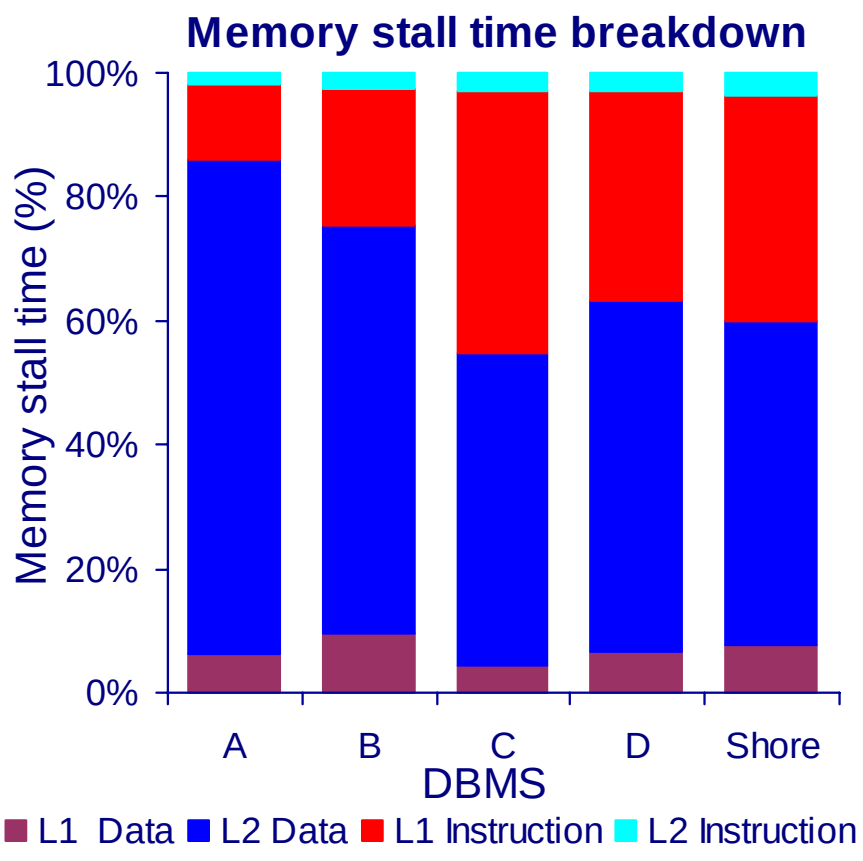
  from R

  where a_j >= Lo **and** a_j <= Hi
- PII Xeon running Windows NT 4
- 16KB L1-I, 16KB L1-D, 512 KB L2, 512 MB RAM
- Used processor counters
- Implemented schemes on Shore Storage Manager
  - Similar behavior to commercial Database Systems

Why Use Shore?

- Compare Shore query behavior with commercial DBMS
- Execution time & memory delays (range selection)

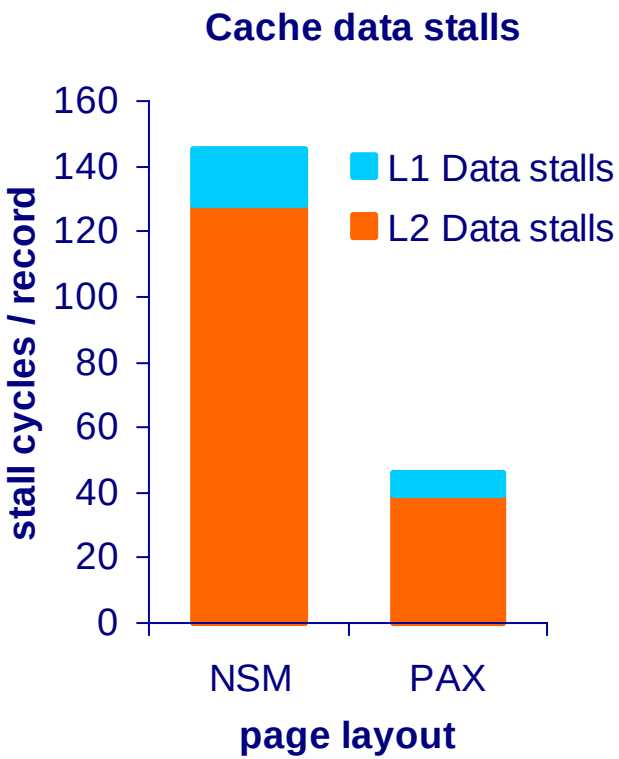


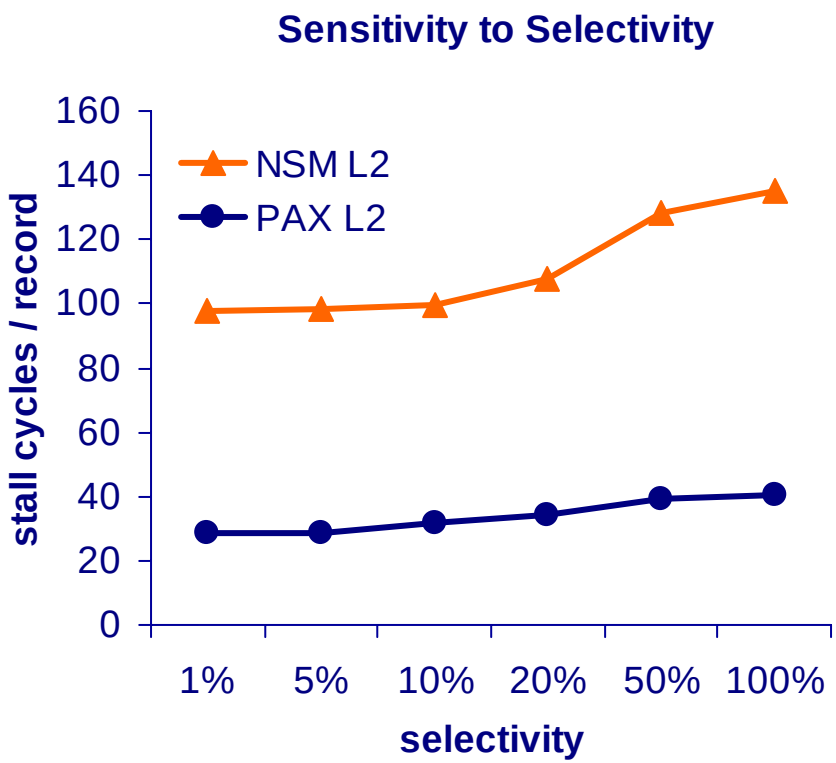


We can use Shore to evaluate DSS workload behavior

Carnegie Mellon

Effect on Accessing Cache Data

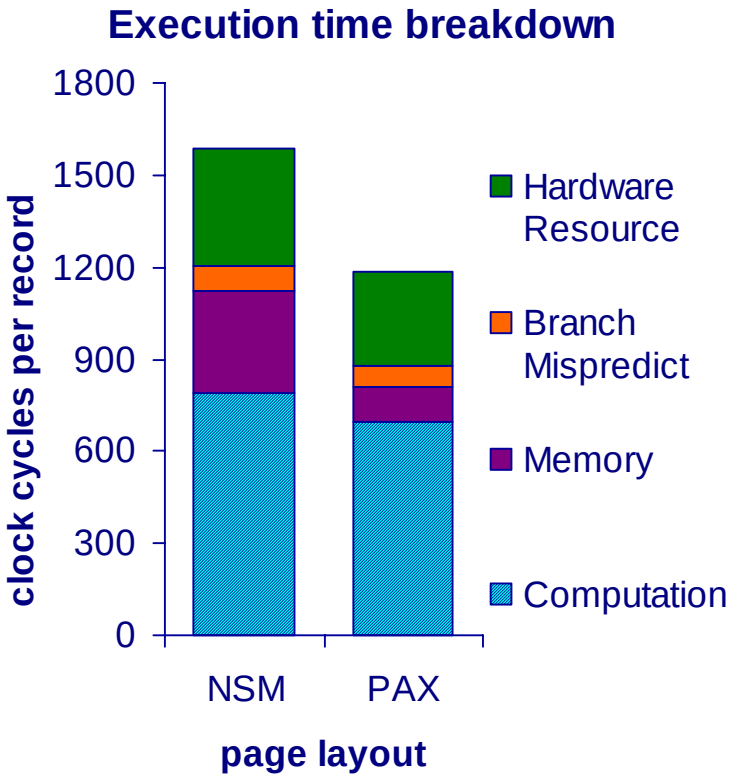


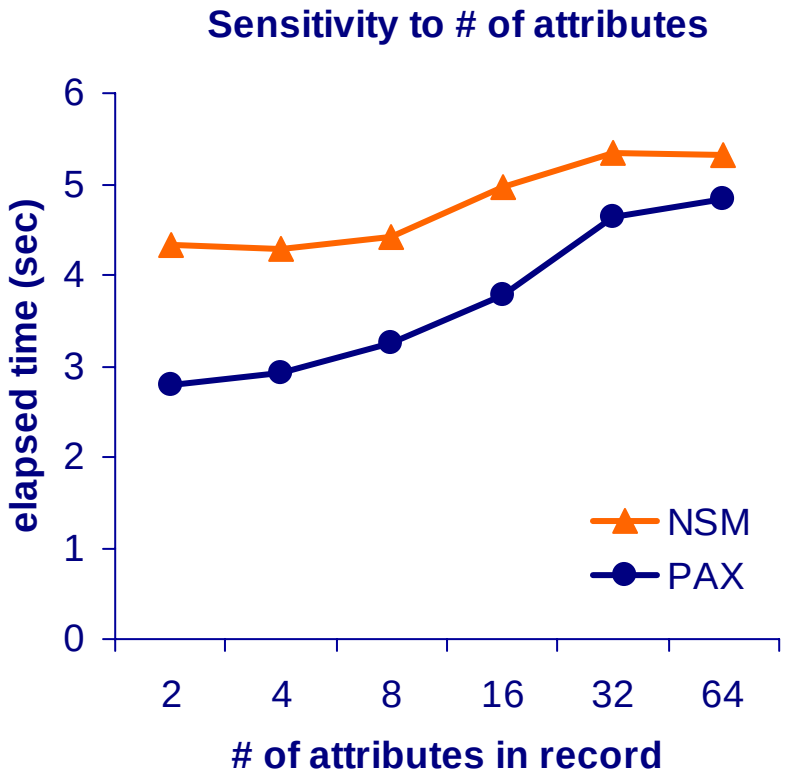


- PAX saves 70% of NSM's data cache penalty
- PAX reduces cache misses at both L1 and L2
- Selectivity doesn't matter for PAX data stalls

Carnegie Mellon

Time and Sensitivity Analysis





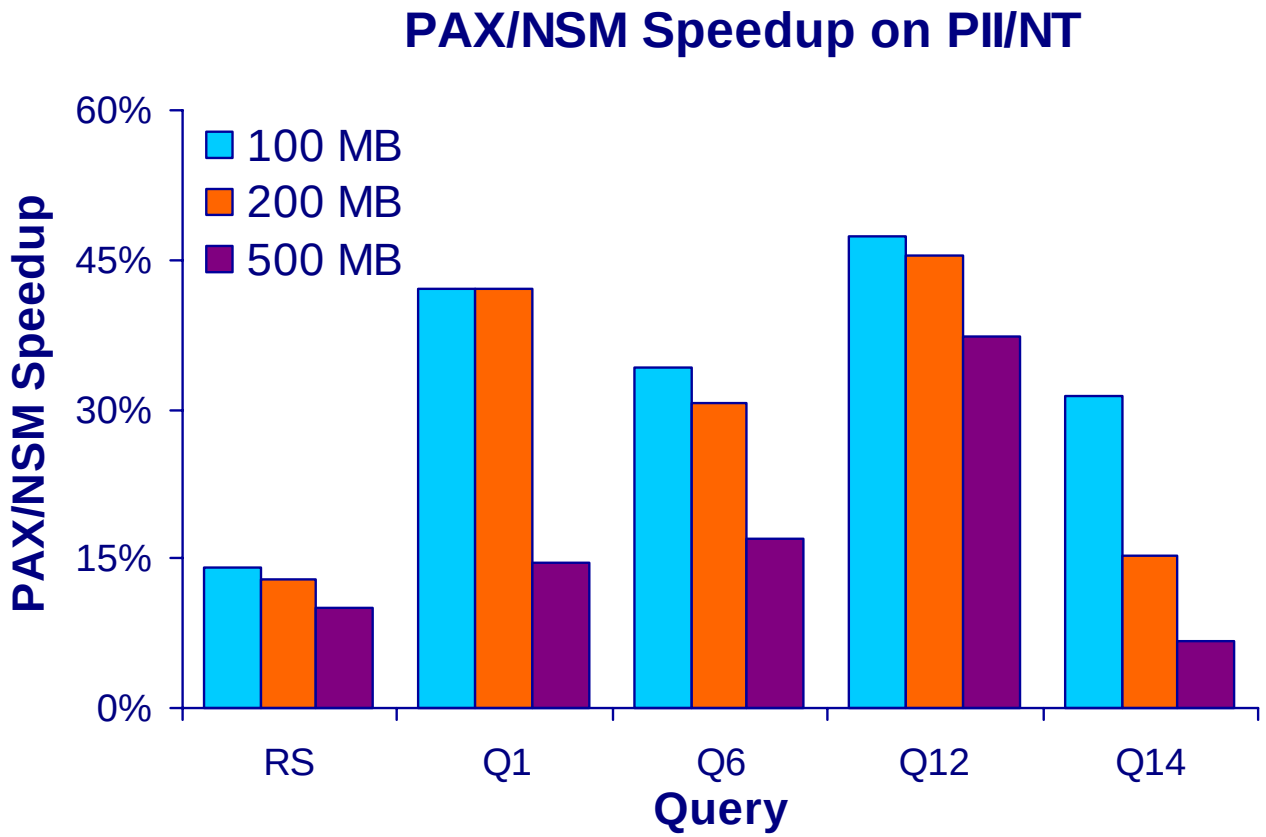
- PAX: 75% less memory penalty than NSM (10% of time)
- Execution times converge as number of attrs increases

Carnegie Mellon

Evaluation Using a DSS Benchmark

- 100M, 200M, and 500M TPC-H DBs
- Queries:
  1. Range Selections w/ variable parameters (RS)
  2. TPC-H Q1 and Q6
     - sequential scans
     - lots of aggregates (*sum*, *avg*, *count*)
     - grouping/ordering of results
  3. TPC-H Q12 and Q14
     - (Adaptive Hybrid) Hash Join
     - complex ‘where’ clause, conditional aggregates
- 128MB buffer pool

TPC-H Queries: Speedup



- PAX improves performance even with I/O
- Speedup differs across DB sizes

Carnegie Mellon

Updates

- Policy: Update in-place
- Variable-length: Shift when needed
- PAX only needs shift minipage data

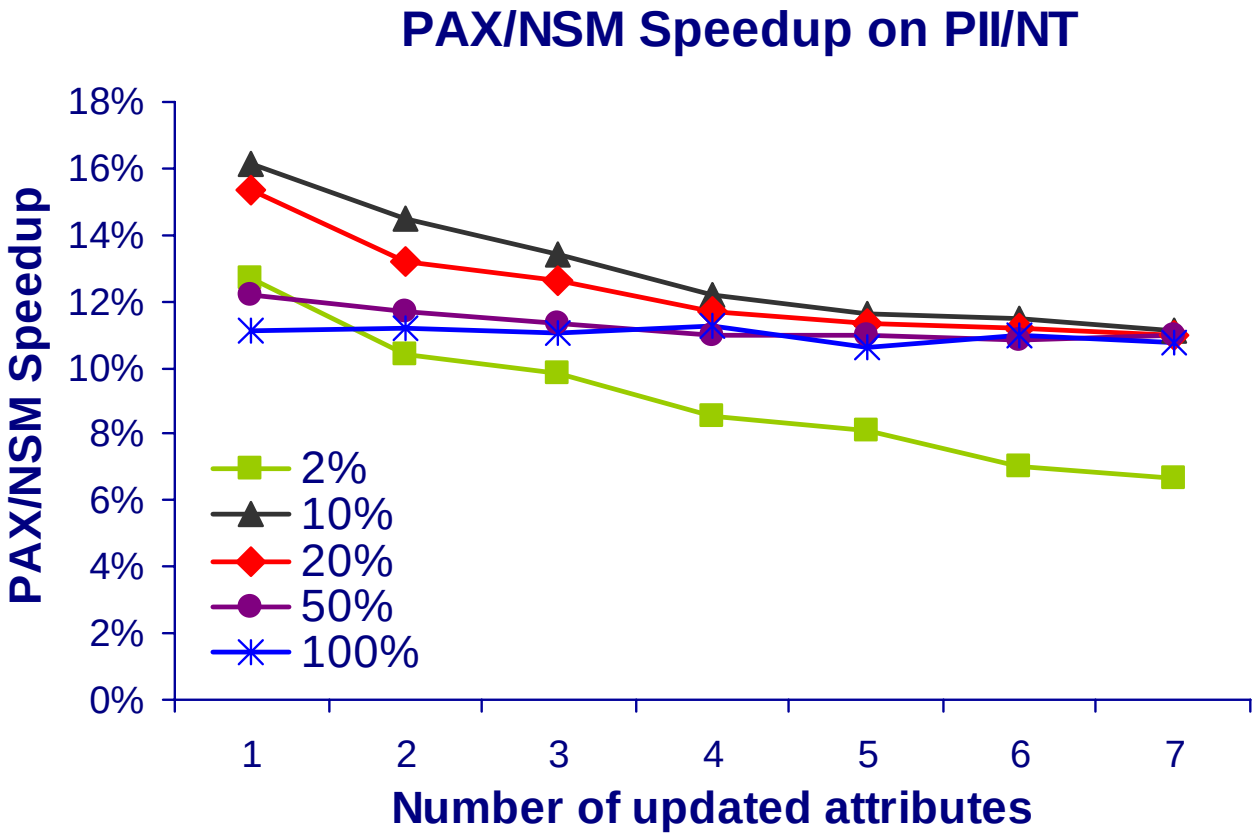
- Update statement:

  update R

  set $a_p = a_p + b$

  where $a_q > Lo$ and $a_q < Hi$

Updates: Speedup



- PAX always speeds queries up (7-17%)
- Lower selectivity => reads dominate speedup
- High selectivity => speedup dominated by write-backs

Carnegie Mellon

Summary

- PAX: a *low-cost, high-impact* DP technique
- Performance
  - Eliminates unnecessary memory references
  - High utilization of cache space/bandwidth
  - Faster than NSM (does not affect I/O)
- Usability
  - Orthogonal to other storage decisions
  - “Easy” to implement in large existing DBMSs

Carnegie Mellon