Databases and Concurrency

Lecture 4

October 3, 2006

Plan for today

- Paper discussion
- Clotho buffer management
- Compound workloads re-visited

Irrelevant Data Wastes Resources

- Performance of memory hierarchy crucial to DB

BUT:

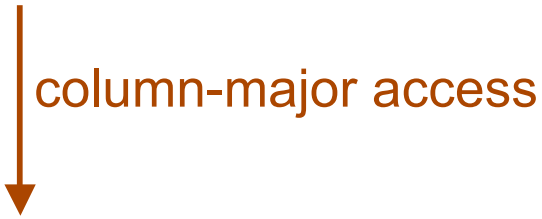
- DB fetches more data than needed from storage
  - TPC-H: > 80% data is irrelevant to query
- Irrelevant data impairs performance
  - Consumes precious I/O operations
  - Wastes memory space, pollutes CPU cache …

Optimize data layout to improve performance

Problems of Current Solutions

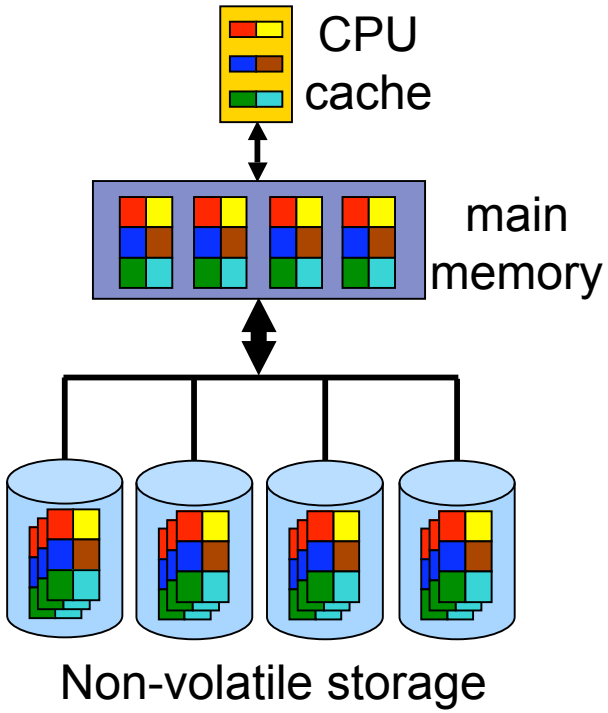
DB Table

A1	B1	C1
A2	B2	C2
A3	B3	C3



- Optimize for one access pattern
  - Trading off performance of others
- Suffer poor I/O performance
  - Irrelevant data
  - Ignorance of storage characteristics

Problems of single & static page layout



Current DB storage architecture

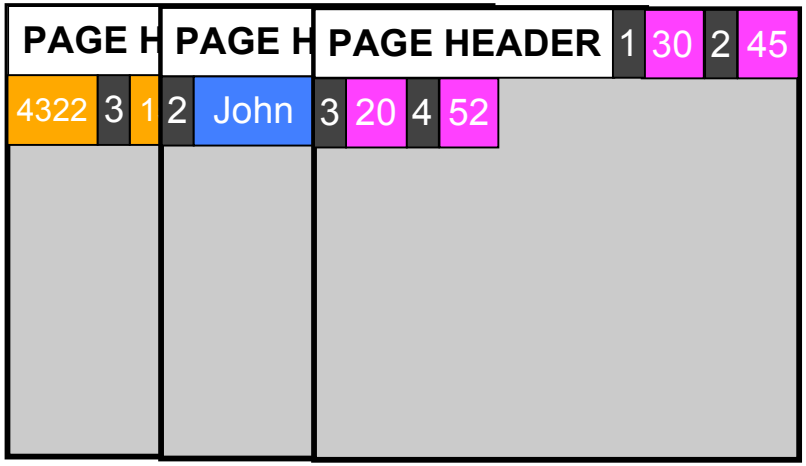
- Single page layout at all levels
  - Can't explore diff. device features
- Static page layout
  - Access useless data
  - Can't optimize diff. workloads

DB page layouts: store tables on disks

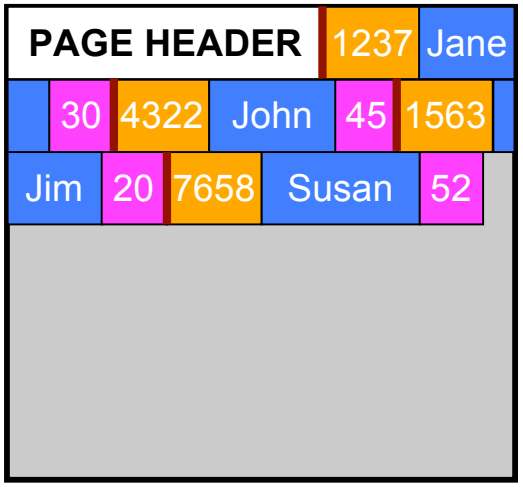
Table R

EID	Name	Age
1237	Jane	30
4322	John	45
1563	Jim	20
7658	Susan	52
...	...	...

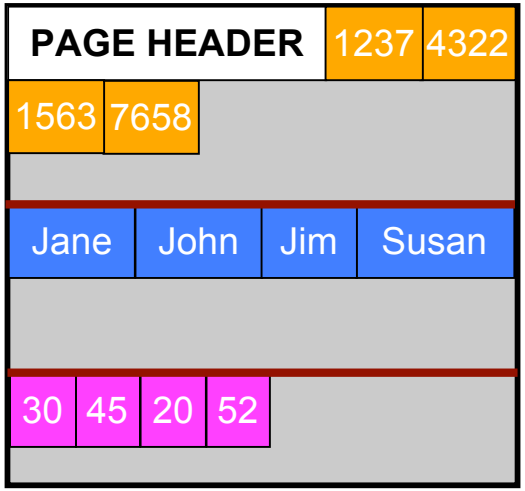
DSM: Store in column-major order



NSM: store in row-major order

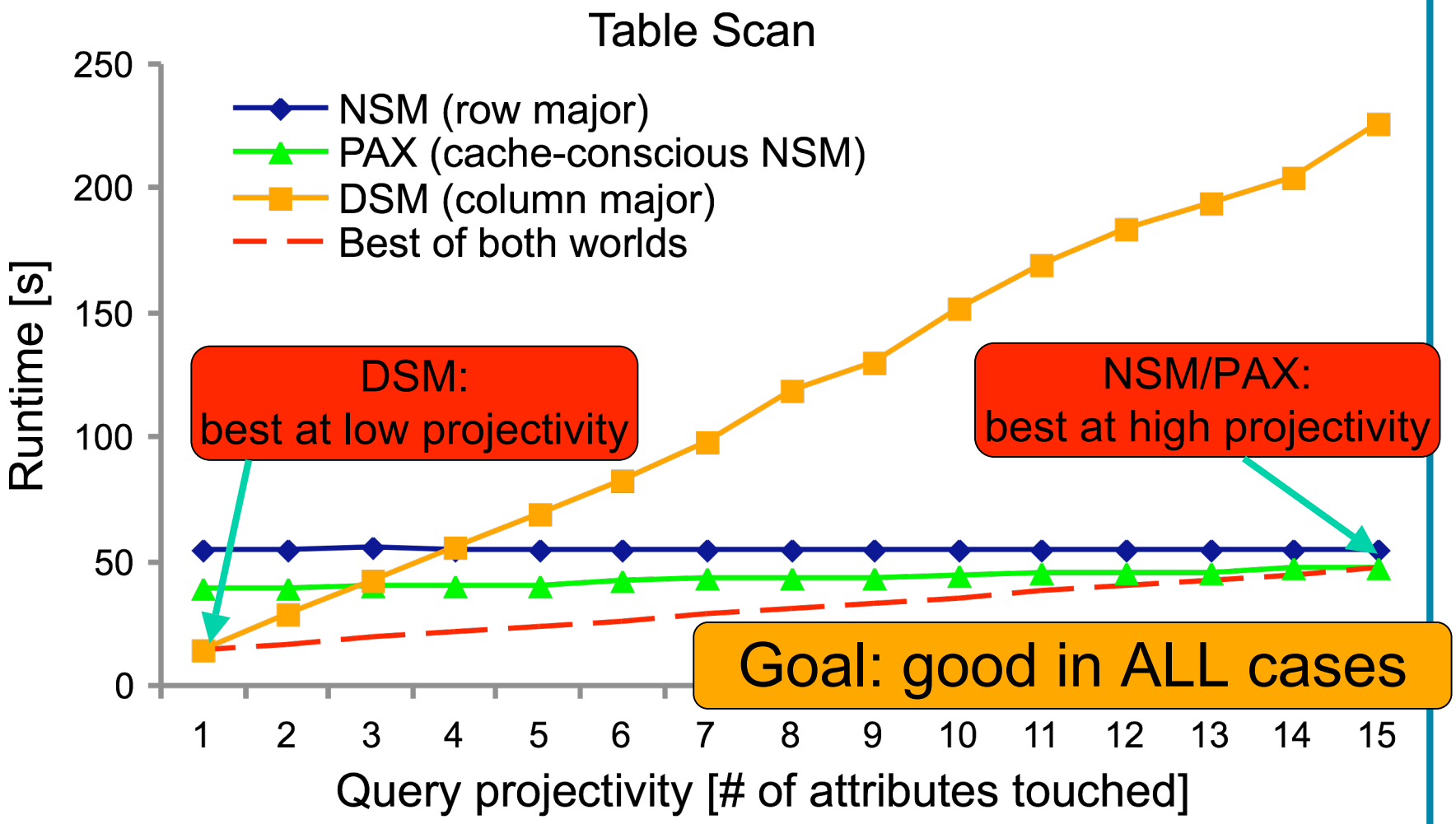


PAX: cache-conscious



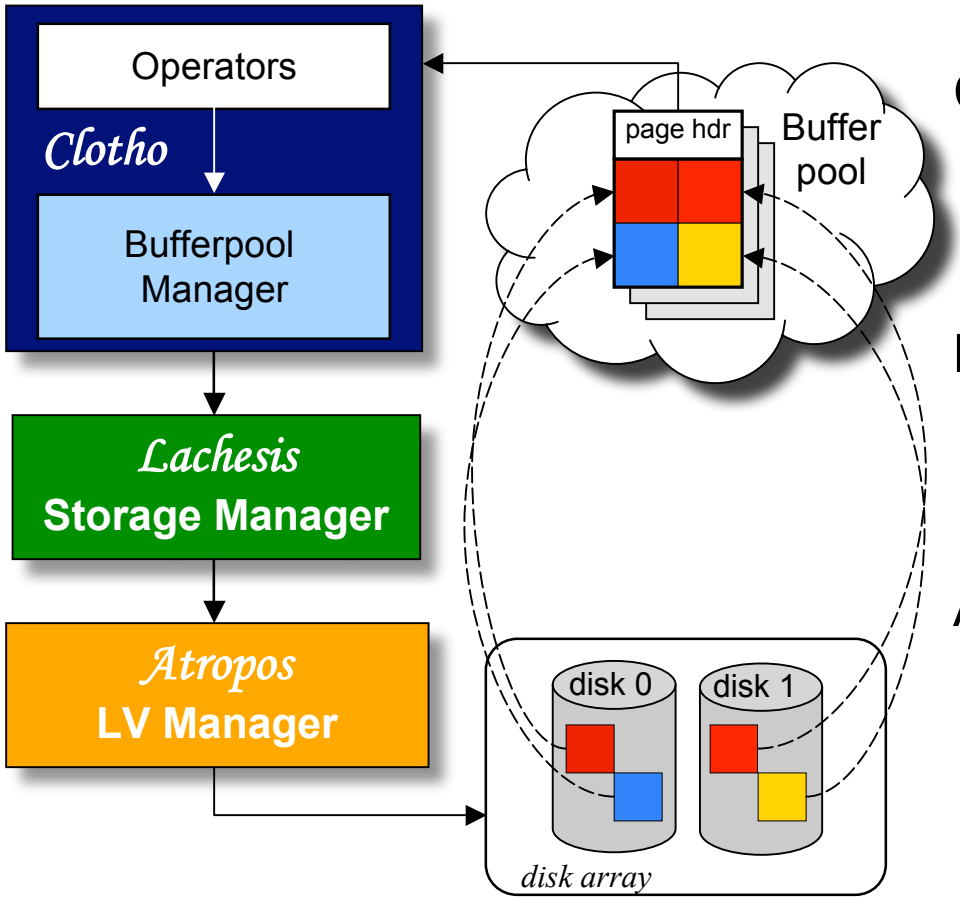
Page layout performance tradeoffs

Table: CREATE TABLE R (FLOAT a1, …, FLOAT a15) (1GB)
Query: SELECT a1, a2, …, FROM R WHERE a1 < Hi



Fates database storage architecture

Optimize layouts at all levels & for all workloads

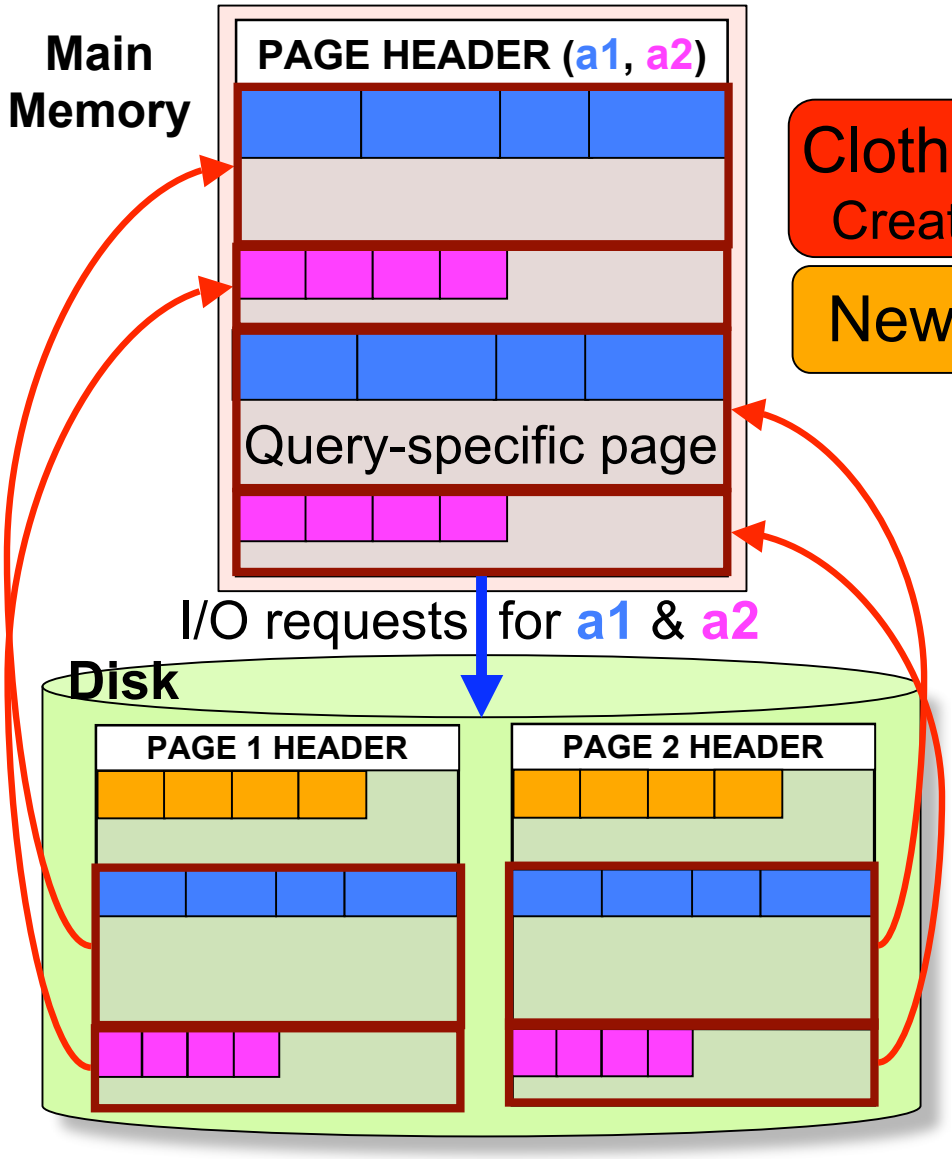


Clotho
manage query-specific pages

Lachesis
construct efficient I/Os

Atropos
access 2D data effectively

Fetch only useful data efficiently



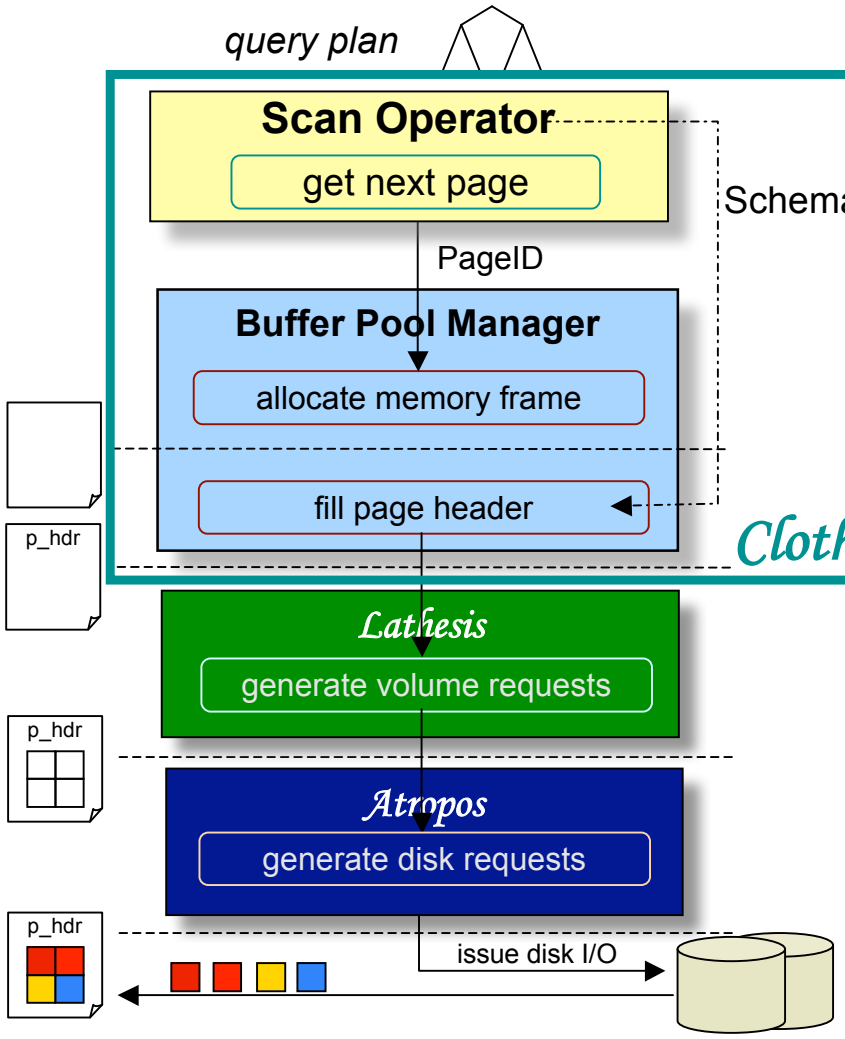
Clotho:
Create query-specific layout (CSM)

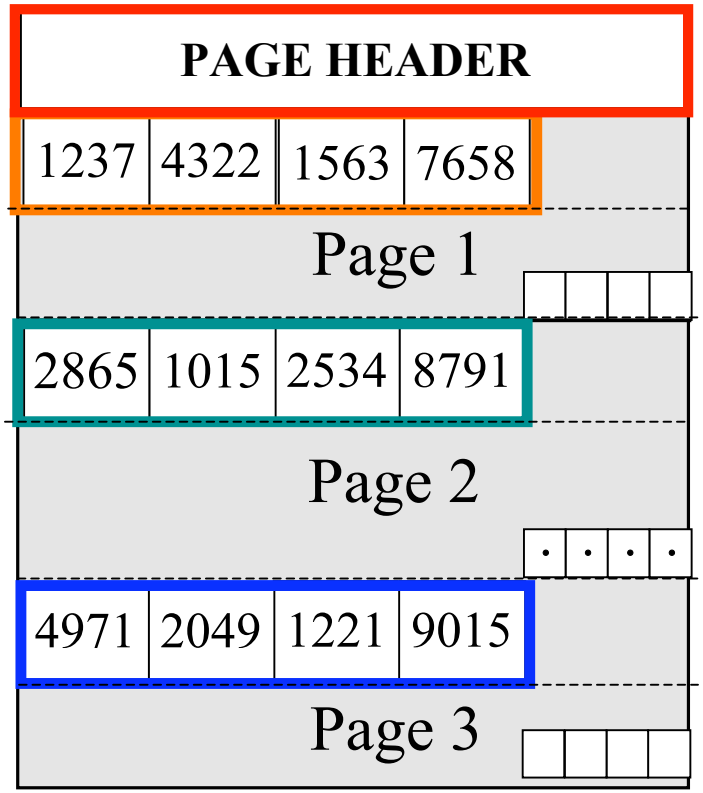
New bufferpool management

Atropos:
Guarantee efficient access

Efficient access to 2D data structures

Tailoring pages to queries





- Clotho: decouple in-memory layout from storage
- Efficient 2-dimensional access at each level of memory hierarchy

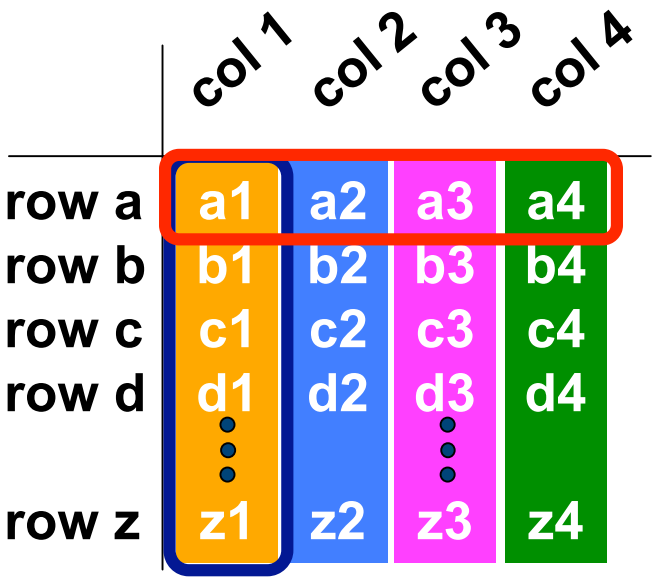
Outline

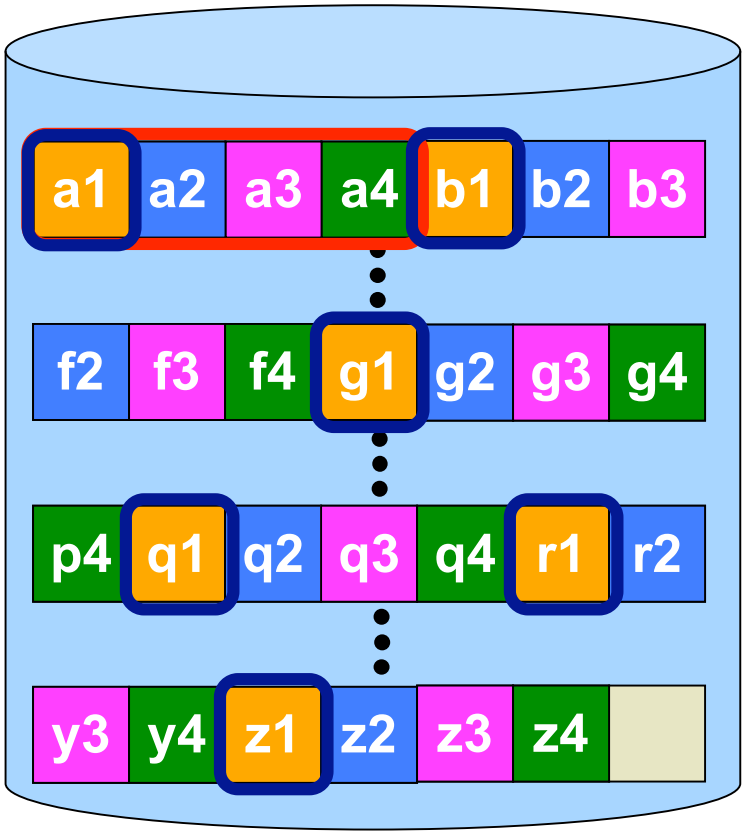
- Introduction
- Atropos: 2-dimensional access
- Clotho: DB bufferpool management
- Evaluation
- Related work
- Summary

Storing 2D data structures on disk

- Disk provides a linear abstraction
- Store 2D data onto 1D disk:
  - Dimension reduction
  - Linearize along one dimension
    - Choose a major order
- This is a problem, because …

Naïve layout: row-major order



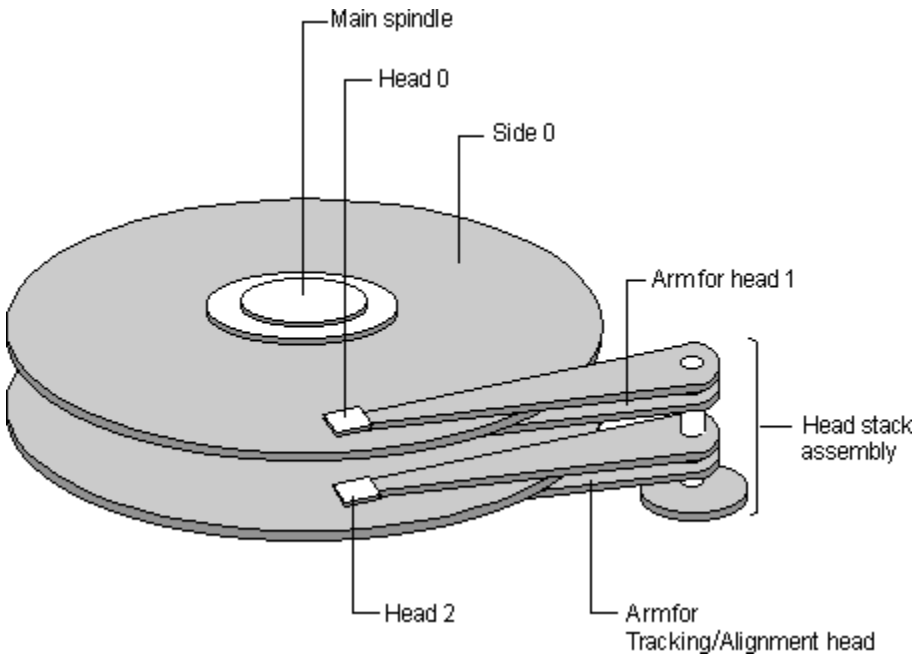


Linear representation

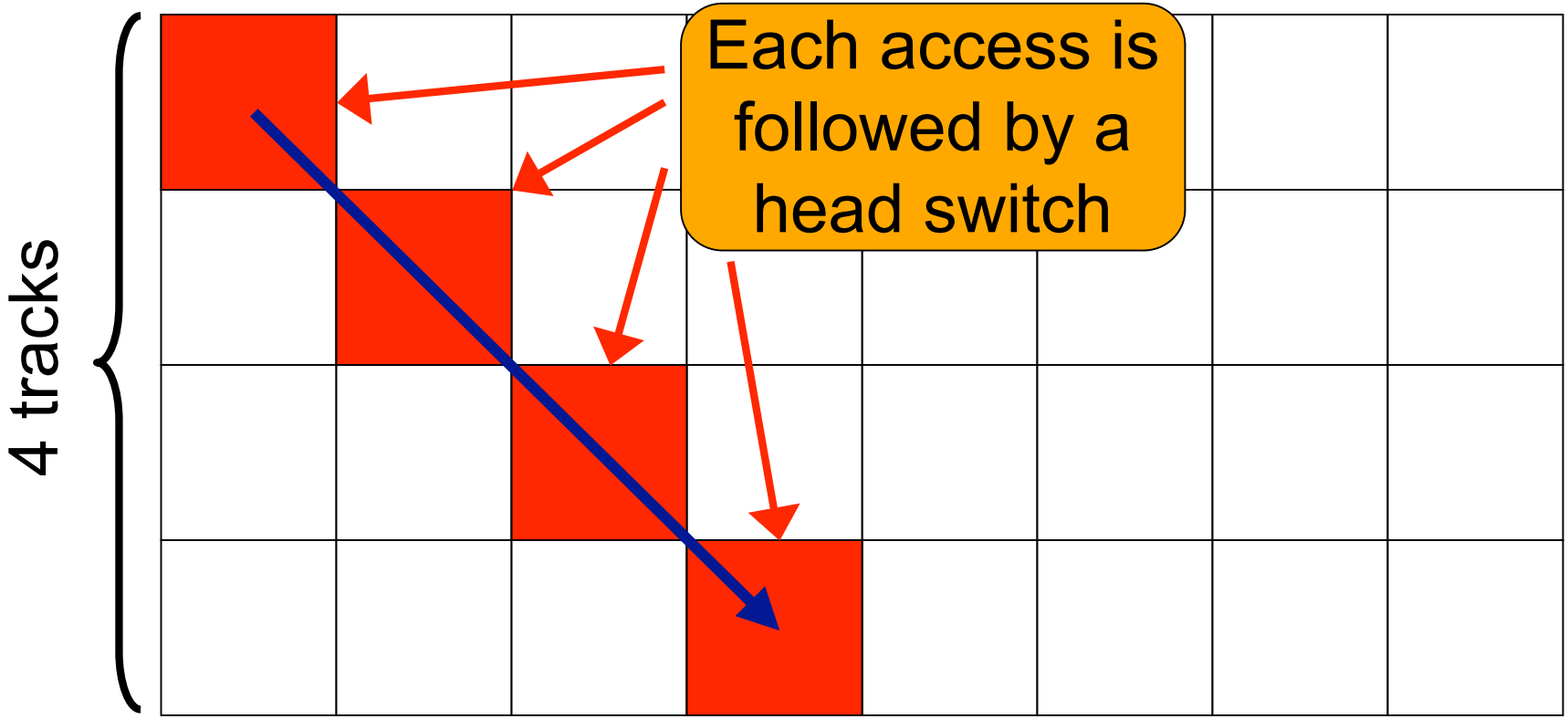
row-major access sequential,
column-major access random

One more dimension on disk?

- Disks are multidimensional machines
  - Cylinder, head, sector
  - Logical blocks map to <c,h,s>



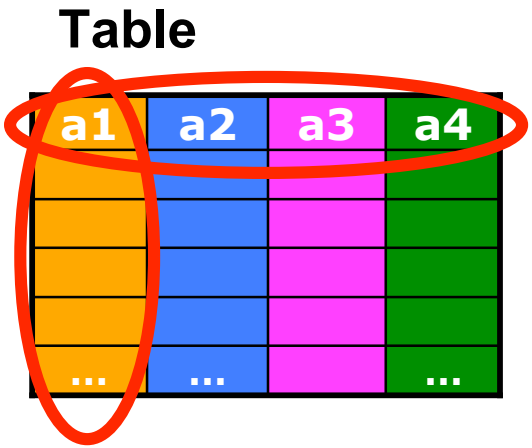
Semi-sequential access



- Not as efficient as sequential access
- More efficient than random access

Two dimensions of efficient access

Data placement on disk



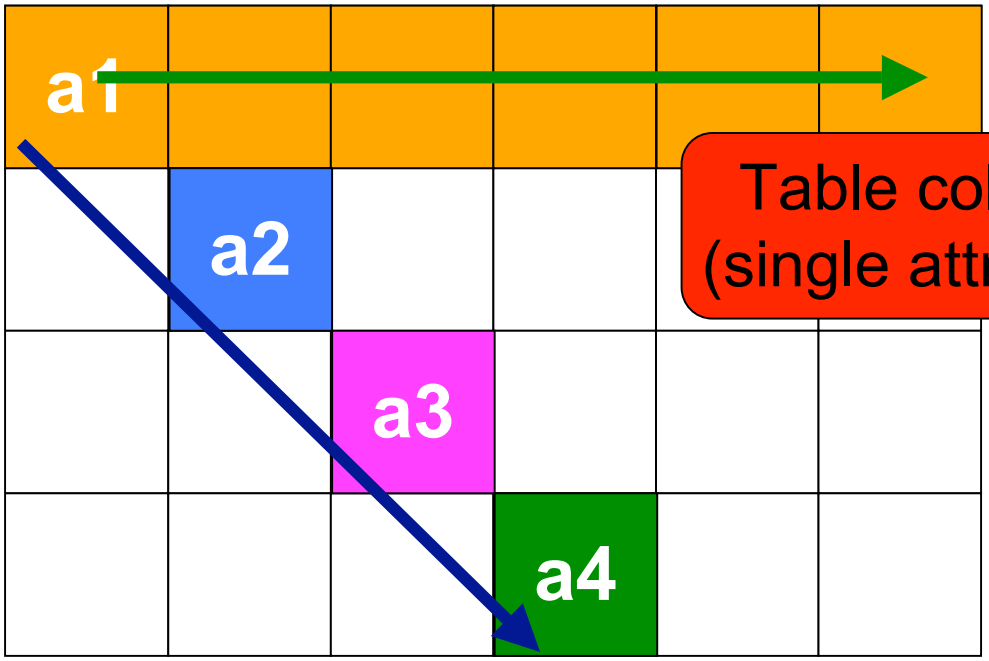


Table column (single attribute)

Table row (full record)

- Single attribute access: sequential
- Full record access: semi-sequential

Outline

- Introduction
- Atropos: 2-dimensional access
- Clotho: DB bufferpool management
- Evaluation
- Related work
- Summary

Bufferpool management

- Single query execution is simple
  - Pages w/ just the needed attributes
- Concurrent queries
  - Attributes in payload: disjoint, overlap, contain
- Updates complicate things further

Goals:

- Enable data sharing across concurrent queries
- Avoid duplicate data as much as possible
- Keep bookkeeping overhead low

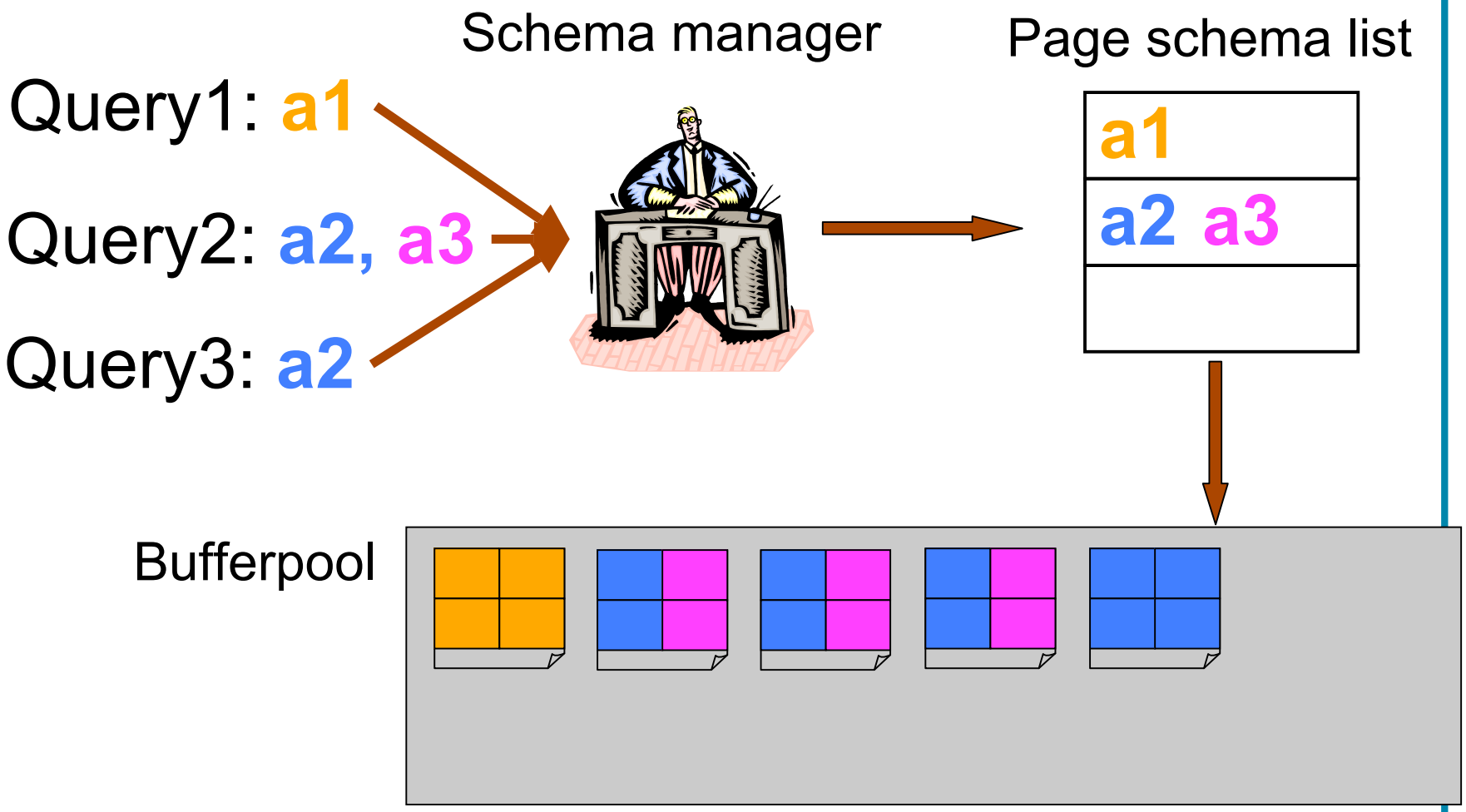
Bufferpool management

- *Schema*: list of attributes in a table
- Page schema: list of attributes in a query-specific page
- Queries with disjoint schemas
  - Use pages with exactly the same schemas
- Queries with overlapped schemas
  - Share pages with a “super schema”
- Page schemas dynamically change
  - Expand: new query overlaps with existing ones
  - Shrink: query finishes

Example: 

Bufferpool management example

Table **R** (*a1*, *a2*, *a3*)



Outline

- Introduction
- Atropos: 2-dimensional access
- Clotho: DB bufferpool management
- Evaluation
- Related work
- Summary

Experiment setup

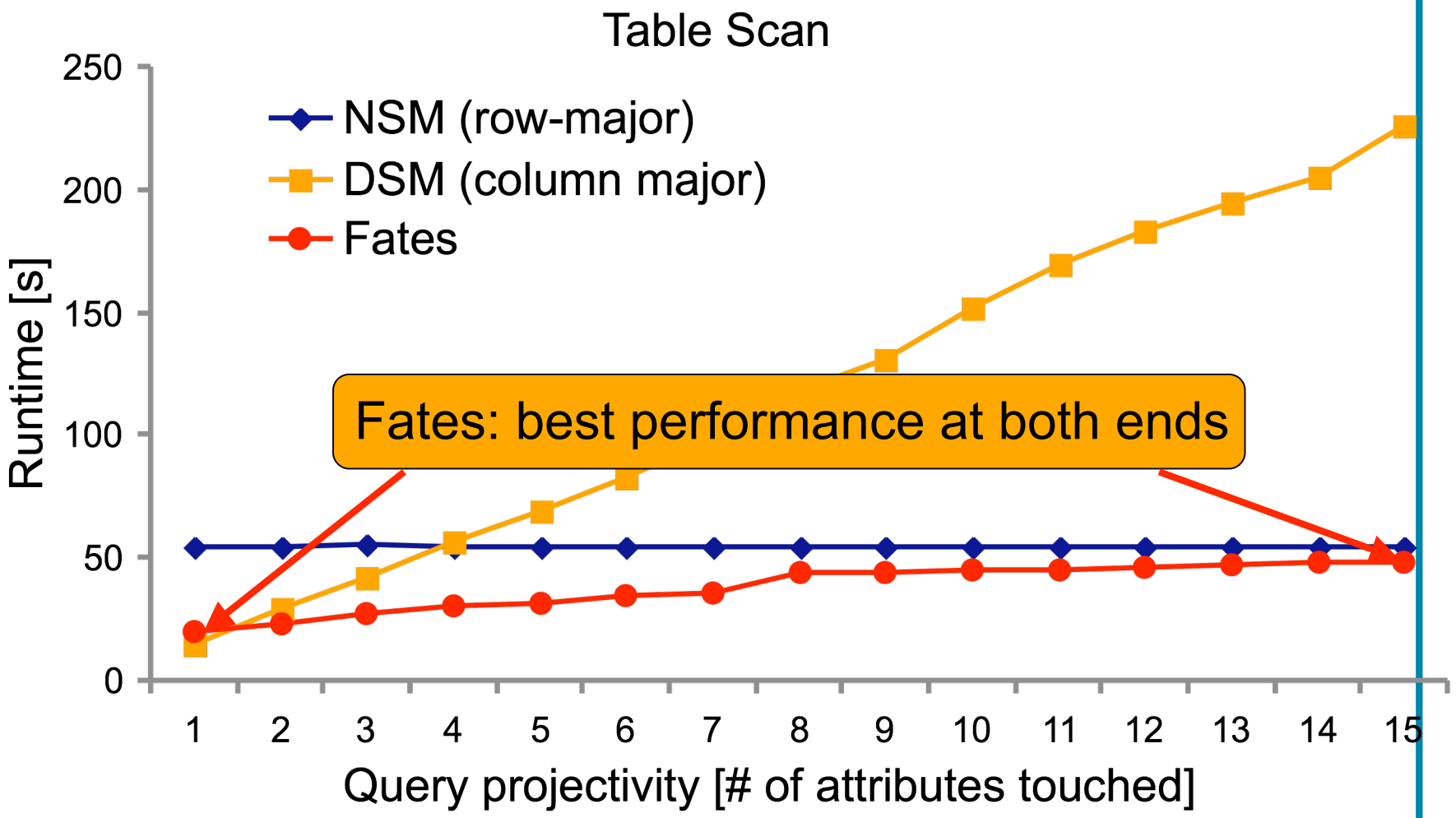
- Shore storage manager [Carey94]
  - Recovery, transaction & concurrency control…
  - NSM, DSM, and PAX layout
- Clotho
  - Query-specific (CSM) layout & its scan operator
  - Bufferpool management
- Atropos
  - Software logical volume manager
  - 4 Seagate Cheetah 36ES 10K RPM disks

Experiment workloads

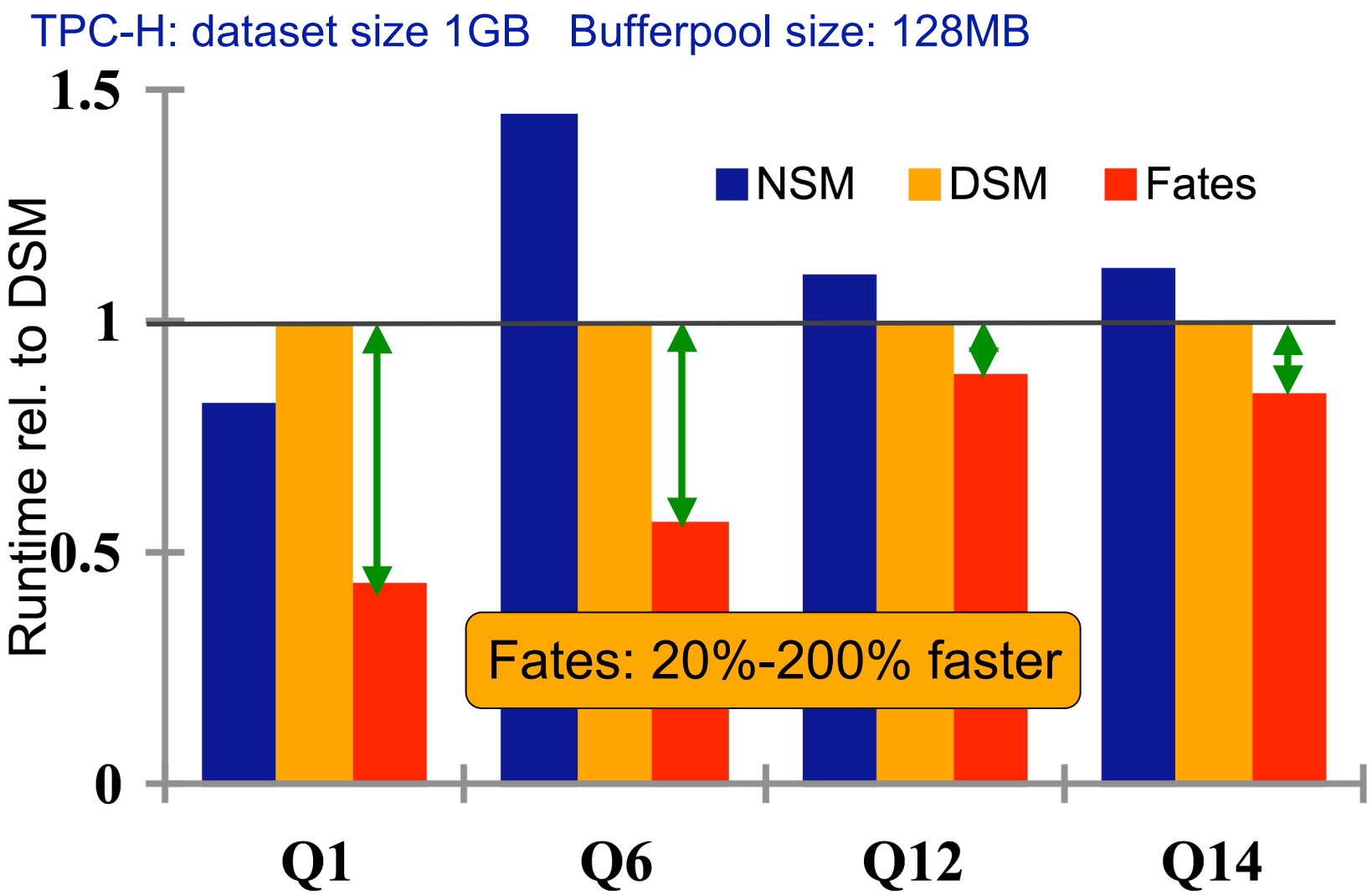
- Microbenchmark
  - Projectivity analysis (range query & point query)
- TPC-H (DSS workload)
  - Sequential access to partial record
  - Suitable for DSM
- TPC-C (OLTP workload)
  - Random access to full record
  - Suitable for NSM/PAX

Projectivity analysis

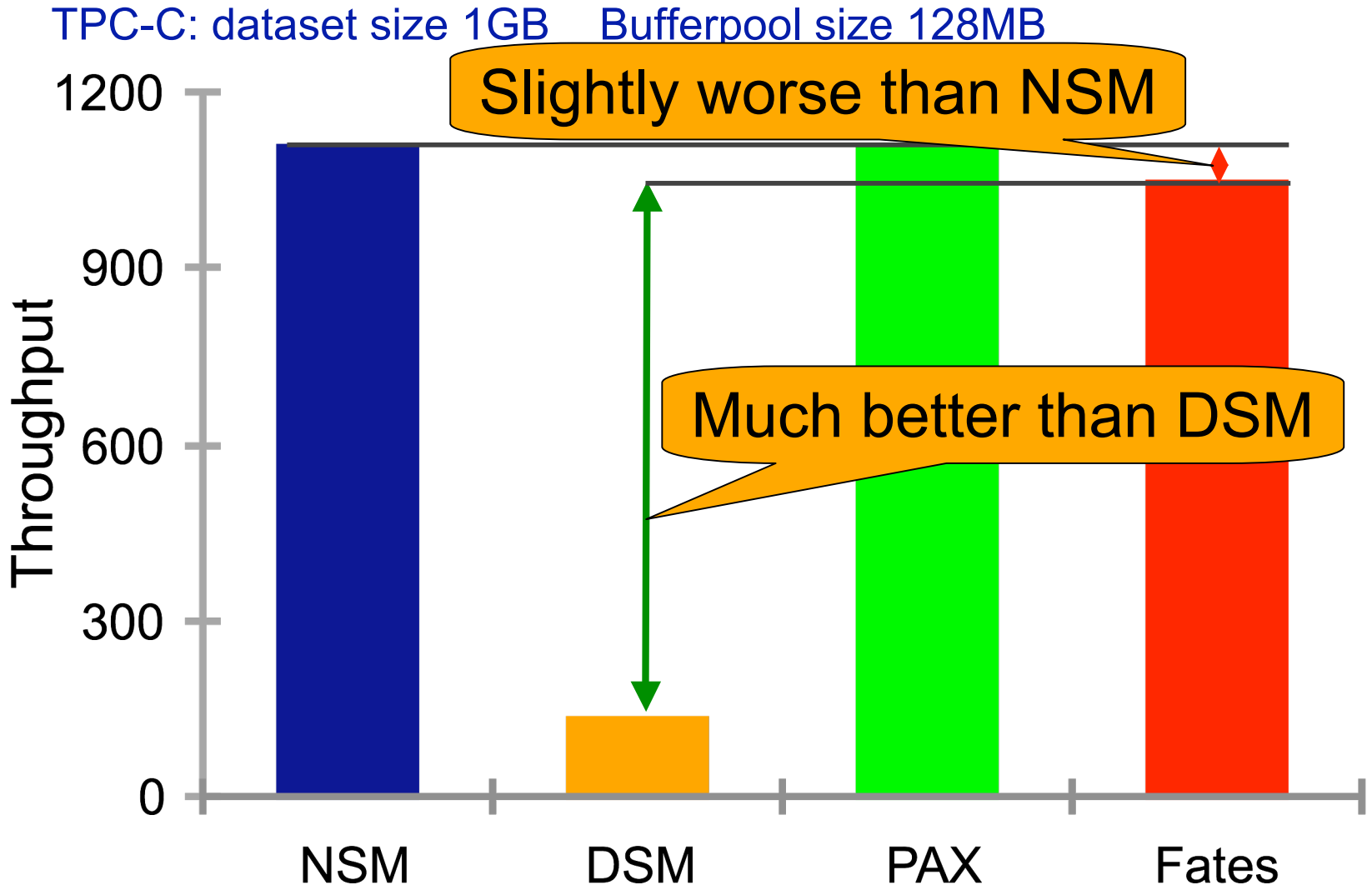
Table: CREATE TABLE R (FLOAT a1, …, FLOAT a15) (1GB)
Query: SELECT a1, a2, …, FROM R WHERE a1 < Hi



TPC-H: partial record access

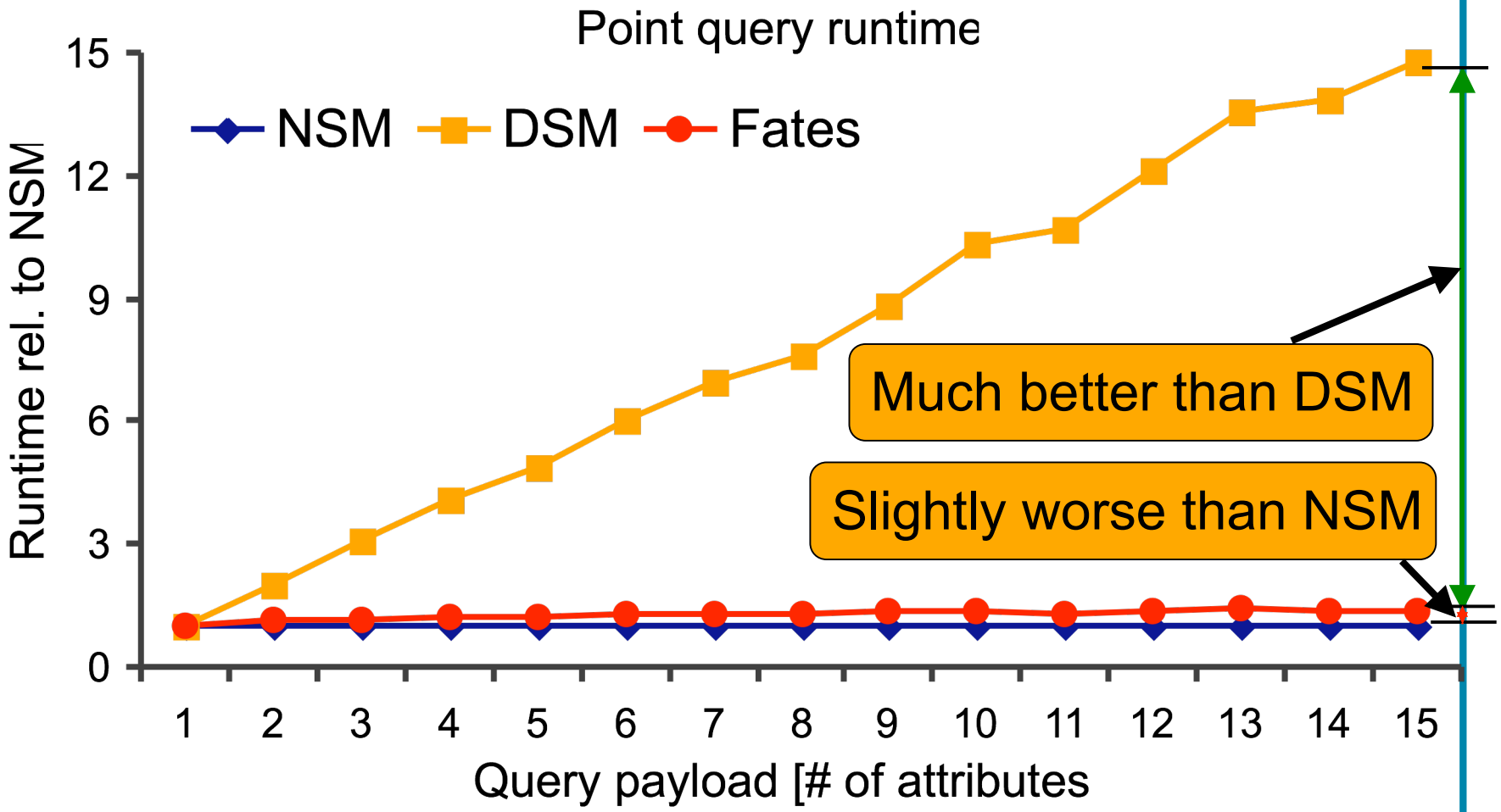


TPC-C: full record access



Projectivity analysis of random access

Table: CREATE TABLE R (FLOAT a1, …, FLOAT a15) (1GB)
Query: SELECT a1, a2, …, FROM R WHERE a1 = Hi



Related work

- NSM, DSM, PAX
- Fractured Mirrors [Ramamurthy02]
  - Keep two copies of data (NSM and DSM)
  - Maintenance is expensive, wastes space
- Data Morphing [Hankins03]
  - Cache-conscious storage technique
  - Same I/O performance as NSM/PAX

Fates summary

- Atropos
  - Efficient access to 2D data structures on disks
- Clotho
  - Flexible page layouts customized to queries
- Fates achieves its goal
  - No performance tradeoffs
  - Optimize both partial & full record access

What's next …

- Next lecture: 10/10
- Readings will be posted tomorrow