15-721 Database Management Systems

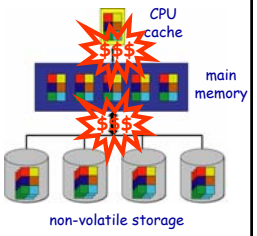
Data Placement in the FATES Automatically-Tuned Database Storage Manager

Instructor: Anastassia Ailamaki

http://www.cs.cmu.edu/~natassa

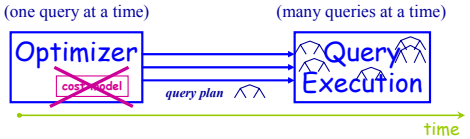
Current DB Storage Managers

- multi-level storage hierarchy
  - different devices at each level
  - different ways to access data on each device
- variable workloads and access patterns
  - device and workload-specific data placement



Goal: Reduce data transfer costs in memory hierarchy
Maximize utilization in memory and in processor cache

Planning and Executing Queries



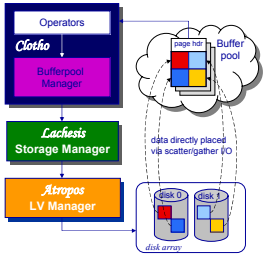
- query interference
  - planner's assumptions invalid at execution time
- manual tuning (error prone, not optimal)
  - error-prone, suboptimal
  - does not take into account disk characteristics

Goal: Restore optimizer's assumptions
Minimize manual tuning

Database storage managers should…

- minimize memory and disk delays
- maximize utilization at all storage levels
- restore query planner's assumptions
- export useful device characteristics
- encapsulate functionality for flexibility
- minimize manual tuning

The Fates DB Storage Manager



Clotho
new in–memory page layout
- query-specific organization
- decoupled from storage!

Lachesis
construction of efficient I/Os
- explicit hints
- restored planner assumptions

Atropos
new data organization
- exploits unique disk features
- transparent device mgmt

Efficient execution at *all* levels of memory hierarchy

Outline

- Introduction
- **Data Layout in DBMS**
- *Clotho* Storage Model
- *Atropos* Logical Volume Manager
- Experimental Results
- Conclusions

Previous Page Layouts

- N-ary Storage Model (NSM)
  - Records stored sequentially in pages
  - Optimized for full-record access
- Decomposition Storage Model (DSM)
  - Each attribute stored as a separate relation
  - Good for only partial-record access
- Partition Attributes Across (PAX)
  - In-page DSM
  - Good CPU cache performance
  - Inherits I/O performance from NSM

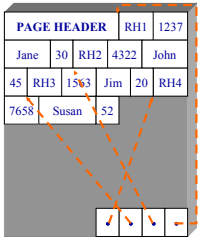
Each optimizes for a subset of applications/devices

"Classic" Disk Page Layout

- NSM (n-ary Storage Model, or Slotted Pages)

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

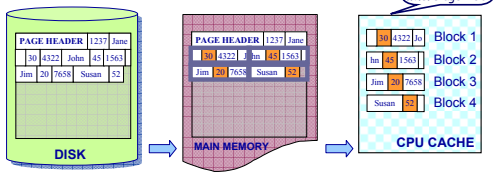
Attributes of a record are stored together

NSM in Memory Hierarchy

HIT Query accesses all attributes (full-record access)

Query evaluates attribute "age" (partial-record access)

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



- Optimized for full-record access
- Slow partial-record access
  - Wastes I/O bandwidth (fixed page layout)
  - Wastes memory bandwidth (garbage in cache)

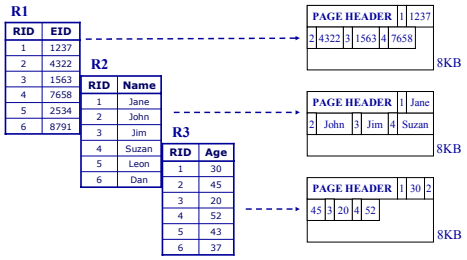
DSM (Decomposition Storage Model)

[CK85]

EID	Name	Age
1237	Jane	30
4322	John	45
1563	Jim	20
7658	Susan	52
2534	Leon	43
8791	Dan	37

Partition original table into *n* 1-attribute sub-tables

DSM (Decomposition Storage Model)



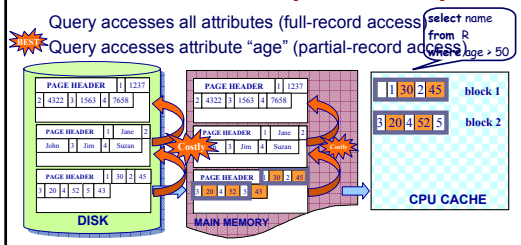
Partition original table into *n* 1-attribute sub-tables

Each sub-table stored separately in NSM pages

DSM in Memory Hierarchy

Query accesses all attributes (full-record access)

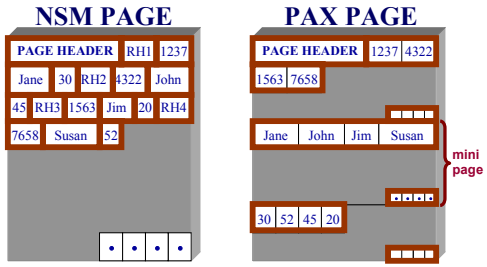
Query accesses attribute "age" (partial-record access)



- Optimized for partial-record access
- Slow full-record access
- Reconstructing full record may incur random I/O

Partition Attributes Across (PAX)

[ADH01]

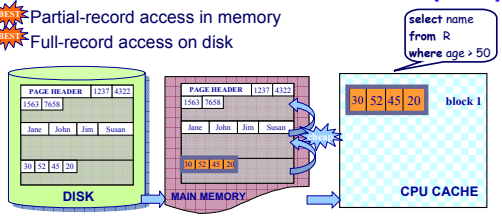


Idea: Partition data *within* page for spatial locality

PAX in Memory Hierarchy

[ADH01]

- Partial-record access in memory
- Full-record access on disk



- **Optimizes cache-to-memory communication**
- **Eliminates unnecessary trips to memory**
- **Retains NSM's I/O (page contents do not change)**

Summary (no replication)

Page layout	Cache-memory Performance		Memory-disk Performance	
	full-record access	partial record access	full-record access	partial record access
NSM	☺	☹	☺	☹
DSM	☹	☺	☹	☺
PAX	☺	☺	☺	☹

Need new placement method:

- Efficient full- and partial-record accesses
- Maximize utilization at all levels of memory hierarchy

Difficult!!! Different devices/access methods
Different workloads on the same database

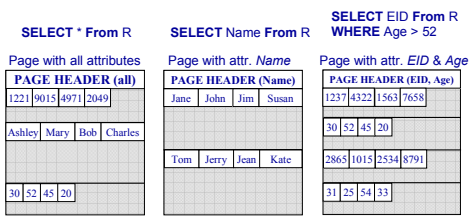
Outline

- Introduction
- Data Layout in DBMS
- ***Clotho* Storage Model**
  - **CSM page layout**
  - **buffer pool management**
- *Atropos* Logical Volume Manager
- Experimental Results
- Conclusions

Clotho Storage Model (CSM)

CSM: In-Memory Page

[SSS04]

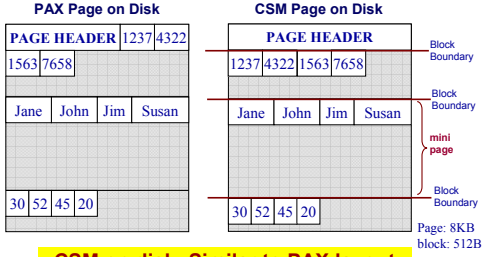


In-memory page contents tailored to query

Clotho Storage Model (CSM)

CSM: On-disk (assumed) page

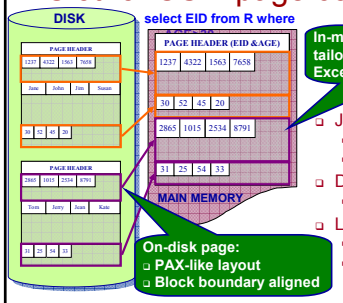
[SSS04]



CSM on disk: Similar to PAX layout
Minipages aligned to block boundary

Clotho: CSM page composition

[SSS04]



- Just the data you need
  - Query-specific pages!
  - Great cache performance
- Decoupled layout
  - Fits different hardware
- Low reconstruction cost
  - Done at I/O level
  - Guaranteed by Lachesis and Atropos [SSS04a]

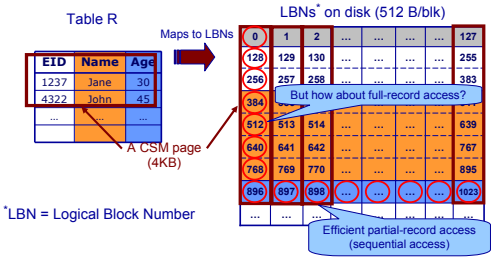
Sharing and consistency in buffer pool?

Clotho: Buffer Pool Management

- Use (page_id, p_sch) to look up records
  - Hit: page_id matches and (p_sch $\subseteq$ schema)
- Page schema management for queries
  - When new query comes
    - **If** (*exist* p_sch $\supseteq$ q_sch) use existing p_sch
    - **else** if (q_sch *disjoint with* all p_sch) add q_sch to p_sch list
    - **else** new p_sch = *union*(q_sch, all p_sch *overlapping with* q_sch)
  - When query finishes
    - *Remove* q_sch, *adjust* p_sch list
- Write statements (update/insert/delete)
  - Use full schema
  - Invalidate frames w/ same page_id, any schema

Minimal bookkeeping / maximum sharing

Time to look at the disk…



Need device-aware data placement!!!

Outline

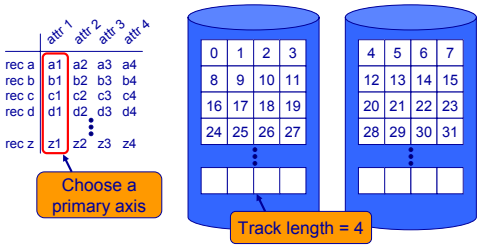
- Introduction
- Data Layout in DBMS
- *Clotho* Storage Model
- ***Atropos* Logical Volume Manager**
  - **basic idea**
  - logical organization
- Efficient disk accesses with *Lachesis*
- Experimental Results
- Conclusions

Atropos Logical Volume Manager

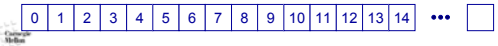
[SSS04a]

- Provides logical-to-physical mapping
- Issues I/Os to individual disks
- Exposes important device attributes
- Exploits device-specific disk characteristics
  - Track-aligned access
  - Semi-sequential access
- Accesses 2D data structures efficiently

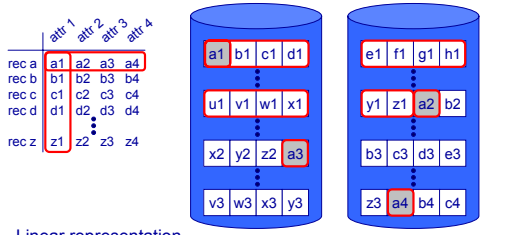
Storing 2D data structures



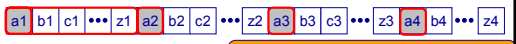
Linear representation



Naïve layout



Linear representation



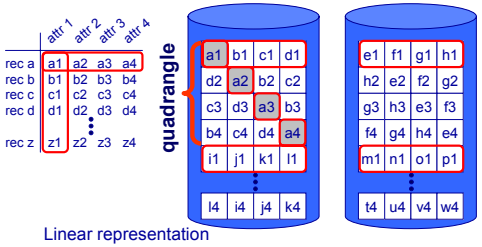
partial-record access sequential, full-record access random

We do not have to use the naïve layout!

Because we decouple the memory layout from storage organization, we can choose a better on-disk layout and avoid random access along the secondary axis.

Atropos layout

[SSS04a]

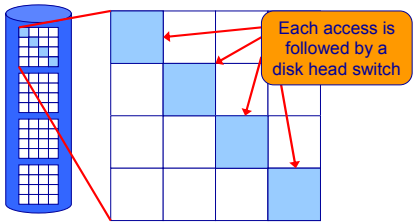


Linear representation

full-record access sequential, partial-record access semi-sequential

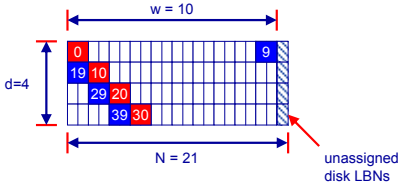
Semi-sequential access

- Exploits knowledge of head switch time
  - Also proposed in [Gorbatenko2002]



Mapping semi-sequential LBNs

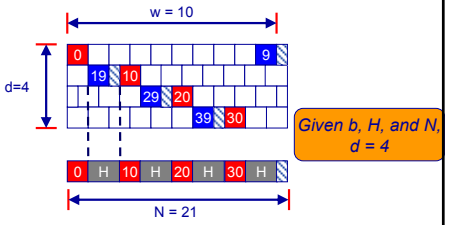
- Four tracks – 21 sectors per track
- Single quadrangle – b = 2, w = 10, d = 4



On next track access head reads next attribute

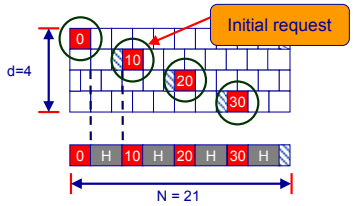
Mapping to physical disk locations

- Four tracks – 21 sectors per track
  - Track skew – H = 3 disk blocks



Issuing requests

- Issue *d* requests to the disk in a batch
- Scheduler chooses initial request
  - Initial seek time, rotational latency
- Remaining requests serviced semi-sequentially

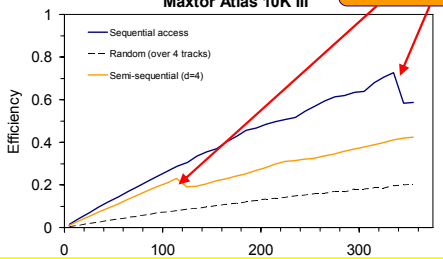


Access efficiency for 2D data

$$\text{efficiency} = \frac{\text{data transfer time}}{\text{total time}}$$

Maxtor Atlas 10K III

More than one revolution



Semi-sequential access much better than random

Atropos on-disk layout: summary

- Volume is organized into "quadrangles"
  - Parameters: block size, width, and depth
  - Disk's physical parameters determine limits
- Quadrangles enable
  - Sequential access for primary axis
  - Semi-sequential access for secondary axis

Atropos abstracts all these details away!

Outline

- Introduction
- Data Layout in DBMS
- *Clotho* Storage Model
- *Atropos* Logical Volume Manager
- **Experimental Results**
- Conclusions

Prototype Implementation

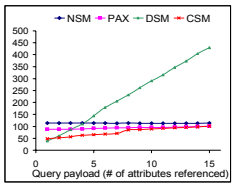
- Shore storage manager [Carey94]
  - Recovery, transaction & concurrency control…
  - NSM, DSM, and PAX layout
- Implement CSM layout & its scan operator
  - Changes *within* page layout module
  - Less than 1000 lines of code
- Implement new buffer pool manager

Experiments: Shore Storage Mgr

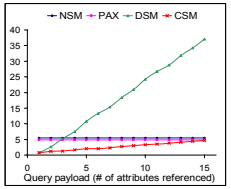
- 4 Seagate Cheetah 36ES 10K RPM disks
- Database workload
  - TPC-C (OLTP workload)
    - Random access to full record
    - Suitable for NSM
  - TPC-H (DSS workload)
    - Sequential access to partial record
    - Suitable for DSM

Sensitivity Analysis

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



Runtime on Disk LVM

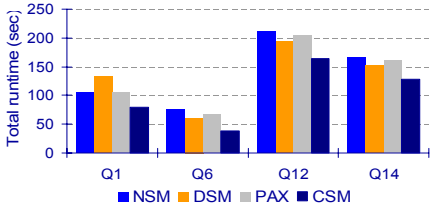


Runtime on MEMStore

CSM follows best performance as payload increases

TPC-H Results

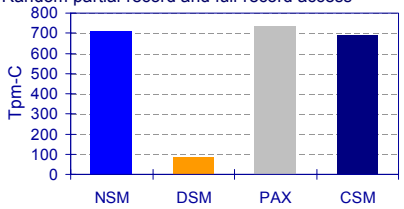
TPC-H: dataset size 1GB
Sequential partial-record access



CSM always faster for DSS applications

TPC-C Result

TPC-C: dataset size 1GB (10 Warehouses, 100 users)
Random partial-record and full-record access



- **CSM runs TPC-C almost as fast as NSM/PAX**
- **Updates: could do partial record access => higher TpmC**
- **Scatter-gather I/O => higher TpmC**

Data Placement Methods: Comparison

Page layout	Cache-memory Performance		Memory-disk Performance	
	full-record access	partial record access	full-record access	partial record access
NSM	☺	☹	☺	☹
DSM	☹	☺	☹	☺
PAX	☺	☺	☺	☹
CSM	☺	☺	☺	☺

CSM/Fates:
Efficient full- and partial-record accesses at all levels of memory hierarchy

Summary

- Device-independent performance hints
  - encapsulate device-specific characteristics
  - restore assumptions about I/O costs
- Proper division of responsibilities
  - DBMS does not control everything
  - storage subsystem executes I/Os efficiently
- Simplified storage management
  - hands-free performance tuning
- Decoupled data placement for robustness
  - easily add new devices

References

[SSS04] **Clotho: Decoupling memory page layout from storage organization.** M. Shao, J. Schindler, S.W. Schlosser, A. Ailamaki, G.R. Ganger. *International Conference on Very Large Data Bases (VLDB), Toronto, Canada, September 2004.*

[SSS04a] **Atropos: A Disk Array Volume Manager for Orchestrated Use of Disks.** J. Schindler, S.W. Schlosser, M. Shao, A. Ailamaki, G.R. Ganger. *USENIX Conference on File and Storage Technologies (FAST), San Francisco, California, March 2004.*

[SAG03] **Lachesis: Robust Database Storage Management Based on Device-specific Performance Characteristics.** J. Schindler, A. Ailamaki, and G. R. Ganger. *International Conference on Very Large Data Bases (VLDB), Berlin, Germany, September 2003.*

[ADH02] **Data Page Layouts for Relational Databases on Deep Memory Hierarchies.** A. Ailamaki, D. J. DeWitt, and M. D. Hill. *The VLDB Journal, 11(3), 2002.*

[ADH01] **Weaving Relations for Cache Performance.** A. Ailamaki, D.J. DeWitt, M.D. Hill, and M. Skounakis. *International Conference on Very Large Data Bases (VLDB), Rome, Italy, September 2001.*

[ADH99] **DBMSs on a Modern Processor: Where Does Time Go?** A. Ailamaki, D. J. DeWitt, M. D. Hill, D.A. Wood. *International Conference on Very Large Data Bases (VLDB), Edinburgh, UK, September 1999.*