Locality-aware Buffer Management to Speedup Disk Accesses

(Progress Report)

Xiaodong Zhang

Ohio State University

In collaboration with

Song Jiang, Wayne State University

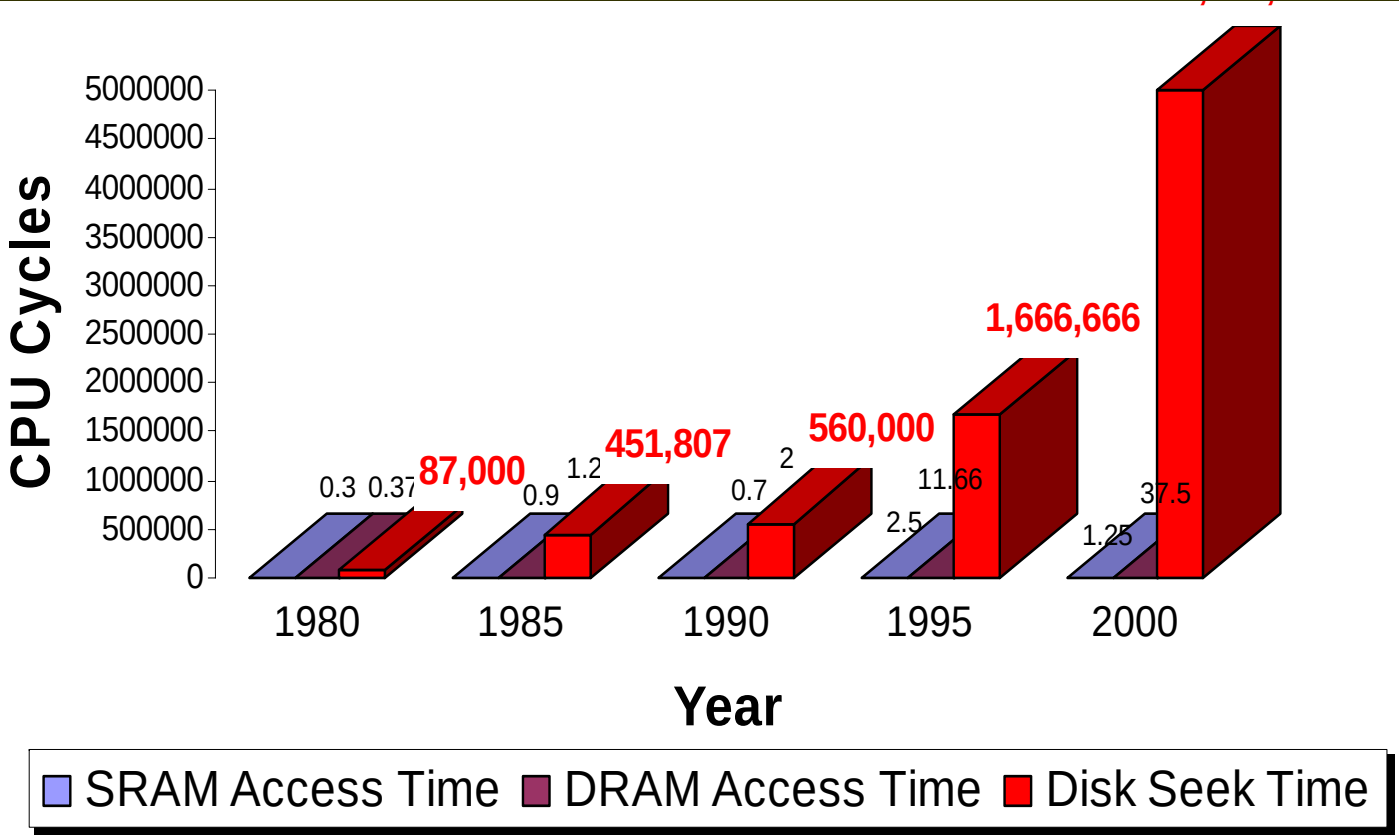
Feng Chen and Xiaoning Ding, Ohio State

Kei Davis, Los Alamos National Lab

"Disk Wall" is an Critical Issue

- Many data-intensive applications generate huge data sets in disks world wide in very fast speed.
  - Stanford Linear Accelerator's BaBar detector generates 1 TB/day in PEP-II Storage Ring.
  - LANL Turbulence Simulation: processing 100+ TB.
  - Google searches and accesses over 10 billion web pages and tens of TB data in Internet.
  - Internet traffic is expected to increase from 1 to 16 million TB/month due to multimedia data.
  - We carry very large digital data, films, photos, …
- Data home is the cost-effective & reliable Disks
  - Slow disk data access is the major bottleneck

The disks in 2000 are 57 times "SLOWER" than their ancestors in 1980 --- increasingly widen the Speed Gap between Peta-Scale computing and Peta-Byte acesses.



Bryant and O'Hallaron, "Computer Systems: A Programmer's Perspective", Prentice Hall, 2003

Two Major Factors Limiting Disk Performance

- Disk speed is limited by mechanical constraints.
  - seek/rotation (random access is slow/high power)
- Access of sequential blocks is fastest.
- OS is not disk-layout aware
  - Limited info for I/O data caching and prefetching

Randomly Scattered Disk Accesses

- Scientific computing
  - Scalable IO (SIO) Report: “in many applications majority of the requests are for small amount of data (less than a few Kbytes)” [Reed 1997]
  - CHARISMA Report: “large, regular data structures are distributed among processes with interleaved accesses of shared files” [Kotz 1996]
- Workloads on popular operating systems
  - UNIX: most accessed files are short in length (80% are smaller than 26 Kbytes) [Ousterhout,1991]
  - Windows NT: 40% I/O operations are to files shorter than 2KBytes [Vogels, 1999]

Random and Scattered Disk Accesses Caused by Multiple Sequential Objects

- Modern disk arrays:
  - HP FC-60 disk arrays: “Most workloads have a range of small and large jumps in sequential accesses and interferences between concurrent access streams”. [Keeton 2001]
  - Detecting sources of irregular disk access patterns: “Although sequential full-file access is relatively common, most data objects are much smaller than the disk request sizes needed to achieve good efficiency.” [Shindler 2002]
- Peta-Byte data analysis is very slow:
  - Many Peta-Bytes of active data for BaBar expperiments
  - Data analysis: random analysis of small blocks.
  - A researcher has several hundred data streams in batch mode
  - Several hundred concurrent researchers are active.

Existing Approaches and Limits

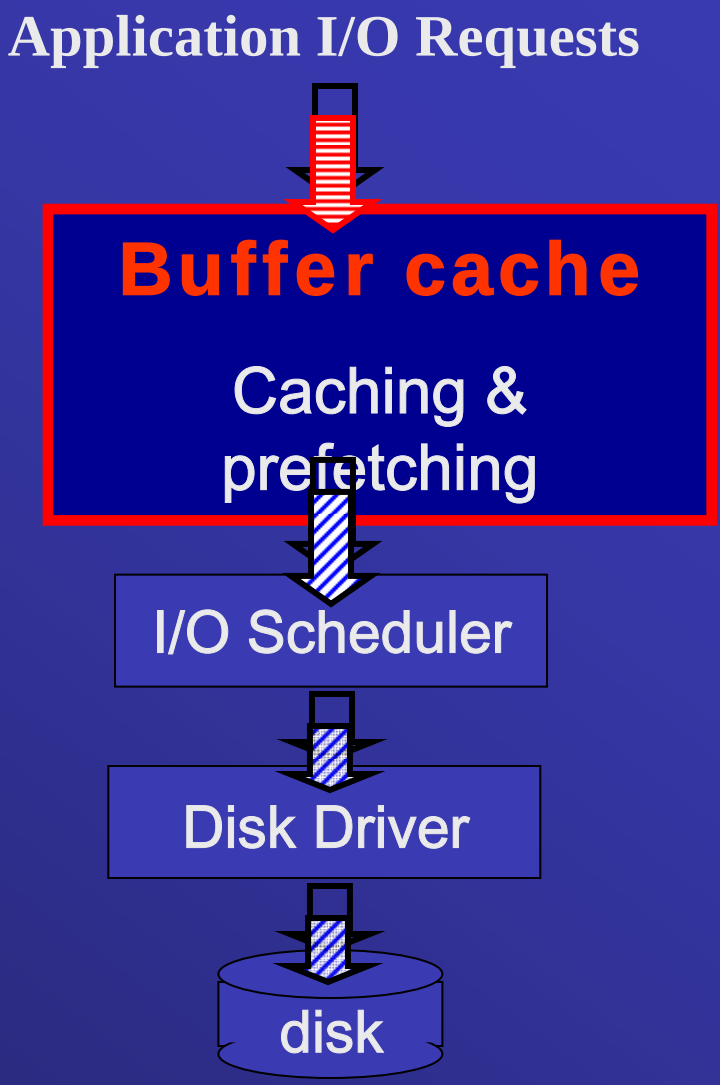
- Programming for Disk Performance
  - Hiding disk latency by overlapping computing
  - Sorting large data sets (SIGMOD'97)
  - Application dependent and programming burden
- Transparent and Informed Prefetching (TIP)
  - Applications issue hints on their future I/O patterns to guide prefetching/caching (SOSP'99)
  - Not a general enough to cover all applications
- Collective I/O: gather multiple I/O requests
  - make contiguous disk accesses for parallel programs

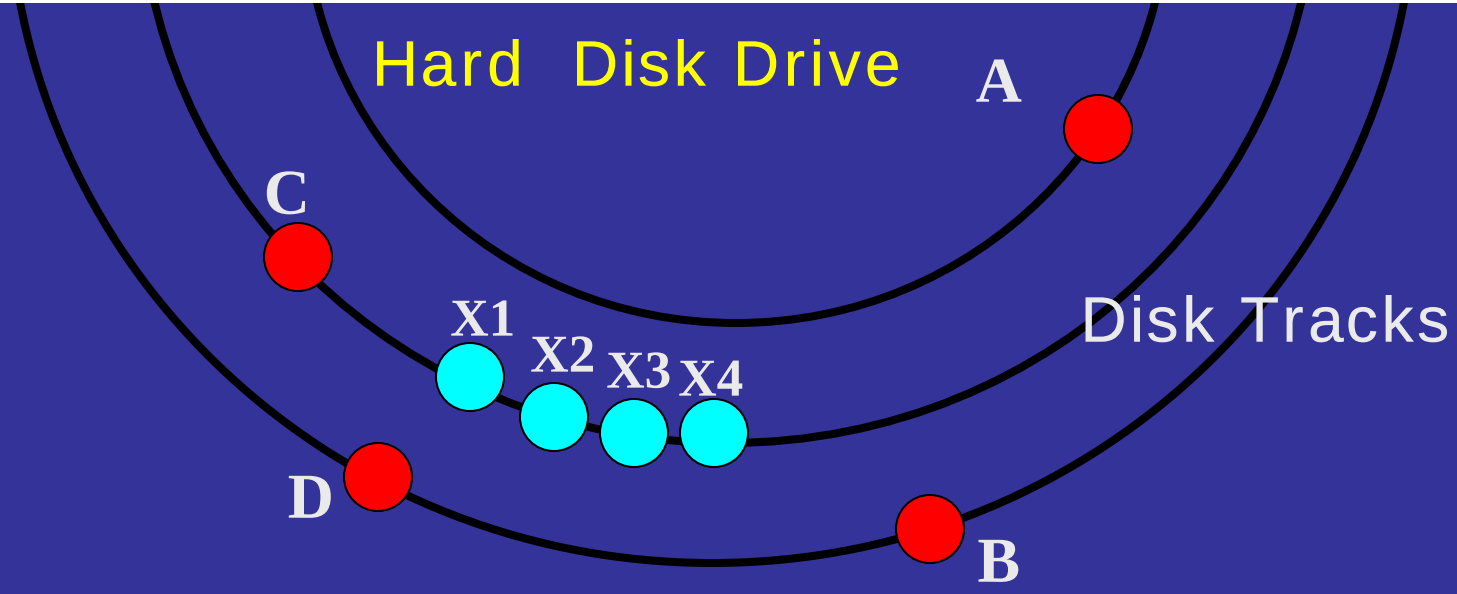
Our Objectives

- Reducing buffer cache miss penalty
  - by minimizing random disk accesses
  - making disk-aware caching and prefetching
- Application independent approach
  - putting disk access information on OS map
  - Exploiting DUal LOcalities (DULO):
    - temporal locality of program execution
    - spatial locality of disk accesses

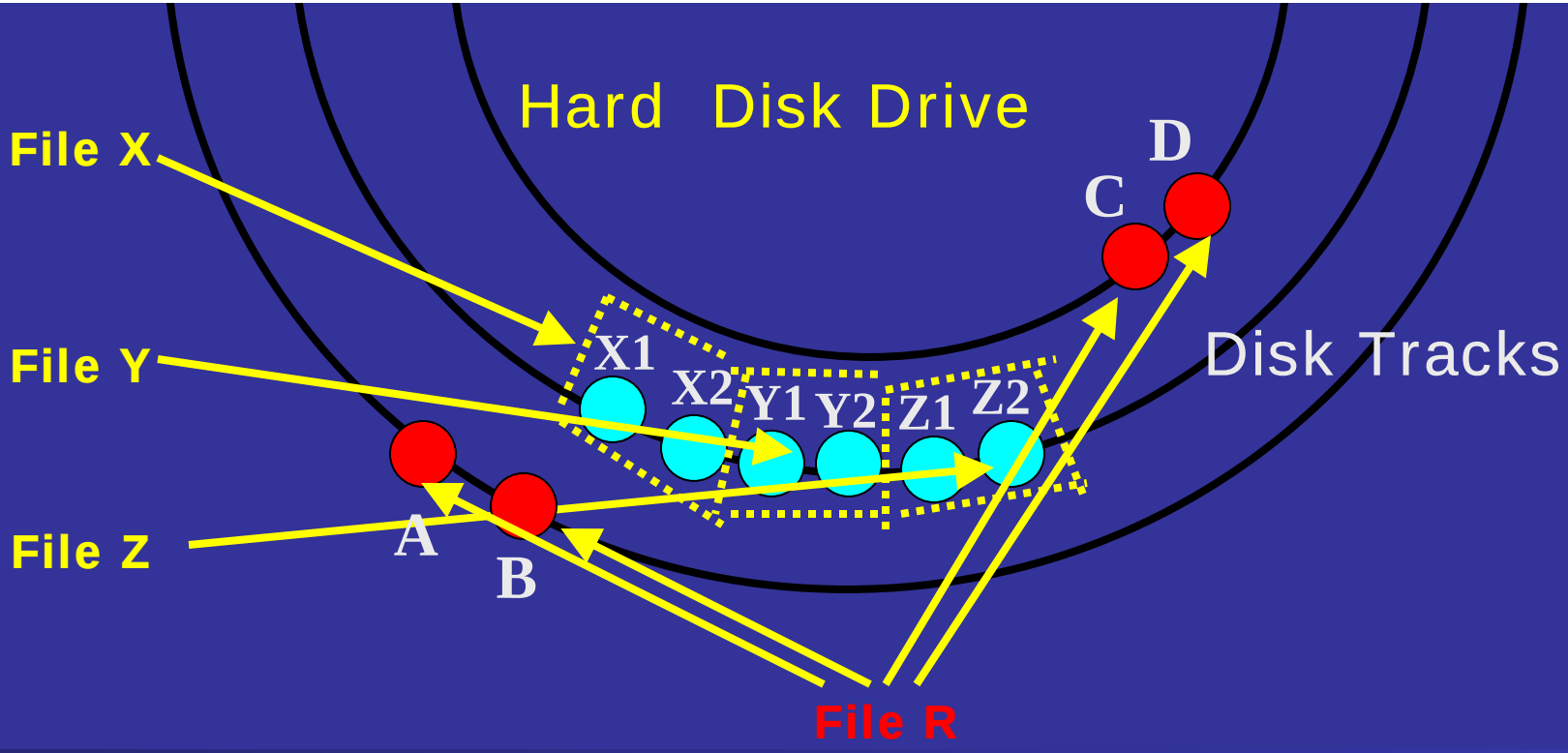
What is Buffer Cache Aware and Unaware?

- Buffer is an agent between I/O requests and disks.
  - aware access patterns in time sequence (in a good position to exploit temporal locality)
  - unaware physical layout (no effort to exploit spatial locality in disks)
- Existing functions
  - send unsatisfied requests to disks
  - **LRU replacement by temporal locality**
  - generate prefetching requests to disks.
- I/O scheduler is not effective: spatial locality is not aware in buffer cache.





- Unique and critical roles of buffer cache
- Buffer cache can influence request stream patterns in disks
- If buffer cache is disk-layout-aware, OS is able to
  - Distinguish sequentially and randomly accessed blocks
  - Give “expensive” random blocks a high caching priority
  - replace long sequential data blocks timely to disks
  - Disk accesses become more sequential.



- Performance is low by logic file abstraction based prefetch.
  - Sequential accesses spanning multiple small files cannot be detected.
  - Sequential layout in logic abstraction may not be sequential layout on physical disk.
  - Detected sequences of blocks are not remembered.

Opportunities and Challenges

- With Disk Spatial Locality (Disk-Seen)
  - Buffer cache exploits Dual Localities (DULO)
  - Address the limits of both caching and prefetching
- A Disk-Seen System Infrastructure
  - analyze and utilize disk-layout Information
  - accurately and timely identify long disk sequences
  - consider trade-offs of temporal and spatial locality (buffer cache hit ratio vs miss penalty)
  - manage its data structures with low overhead
  - Implement it in OS kernel for practical usage

Disk-Seen Task 1: Make Disk Layout Info. Available

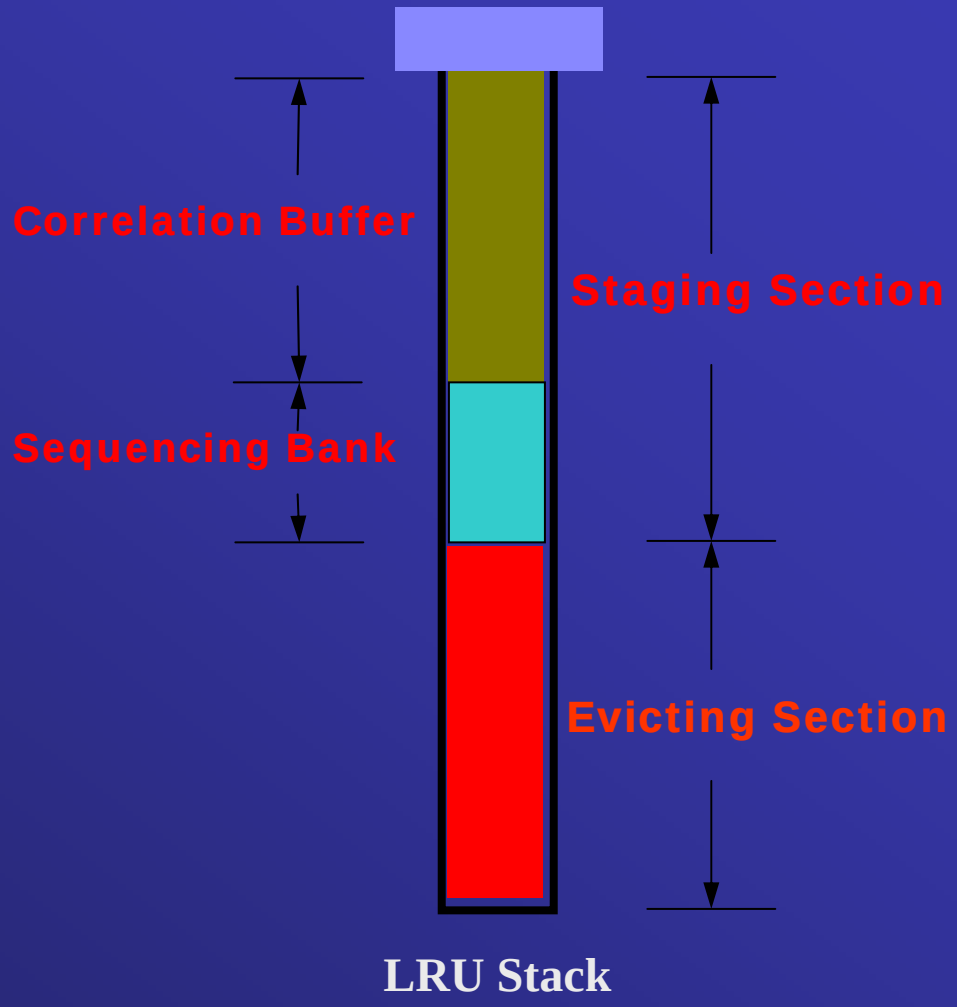
- Which disk layout information to use?
  - Logical block number (LBN): location mapping provided by firmware. (each block is given a sequence number)
  - Accesses of contiguous LBNs have a performance close to accesses of contiguous blocks on disk. (except bad blocks occur)
  - The LBN interface is highly portable across platforms.
- How to efficiently manage the disk layout information?
  - LBN is only used to identify disk locations for read/write;
  - We want to track access times of disk blocks and search for access sequences via LBNs;
  - Disk block table: a data structure for efficient disk blocks tracking.

Disk-Seen TASK 2: Exploiting Dual Localities (DULO)

- **Sequence Forming**

  Sequence ---- a number of blocks whose disk locations are adjacent and have been accessed during a limited time period.

- **Sequence Sorting** based on its recency (temporal locality) and size (spatial locality)



Disk-Seen TASK 3: DULO-Caching

Adapted GreedyDual Algorithm

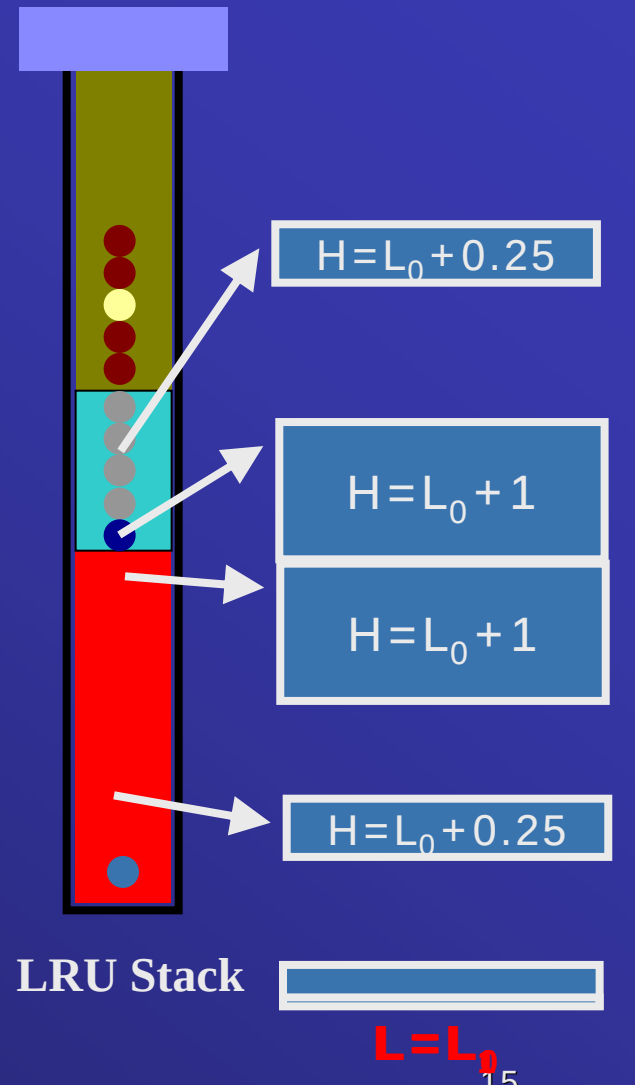
- a global inflation value L , and a value H for each sequence
- Calculate H values for sequences in sequencing bank:

  H = L + 1 / Length(sequence)

  Random blocks have larger H values
- When a sequence (s) is replaced,

  L = H value of s .

  L increases monotonically and make future sequences have larger H values
- Sequences with smaller H values are placed closer to the bottom of LRU stack

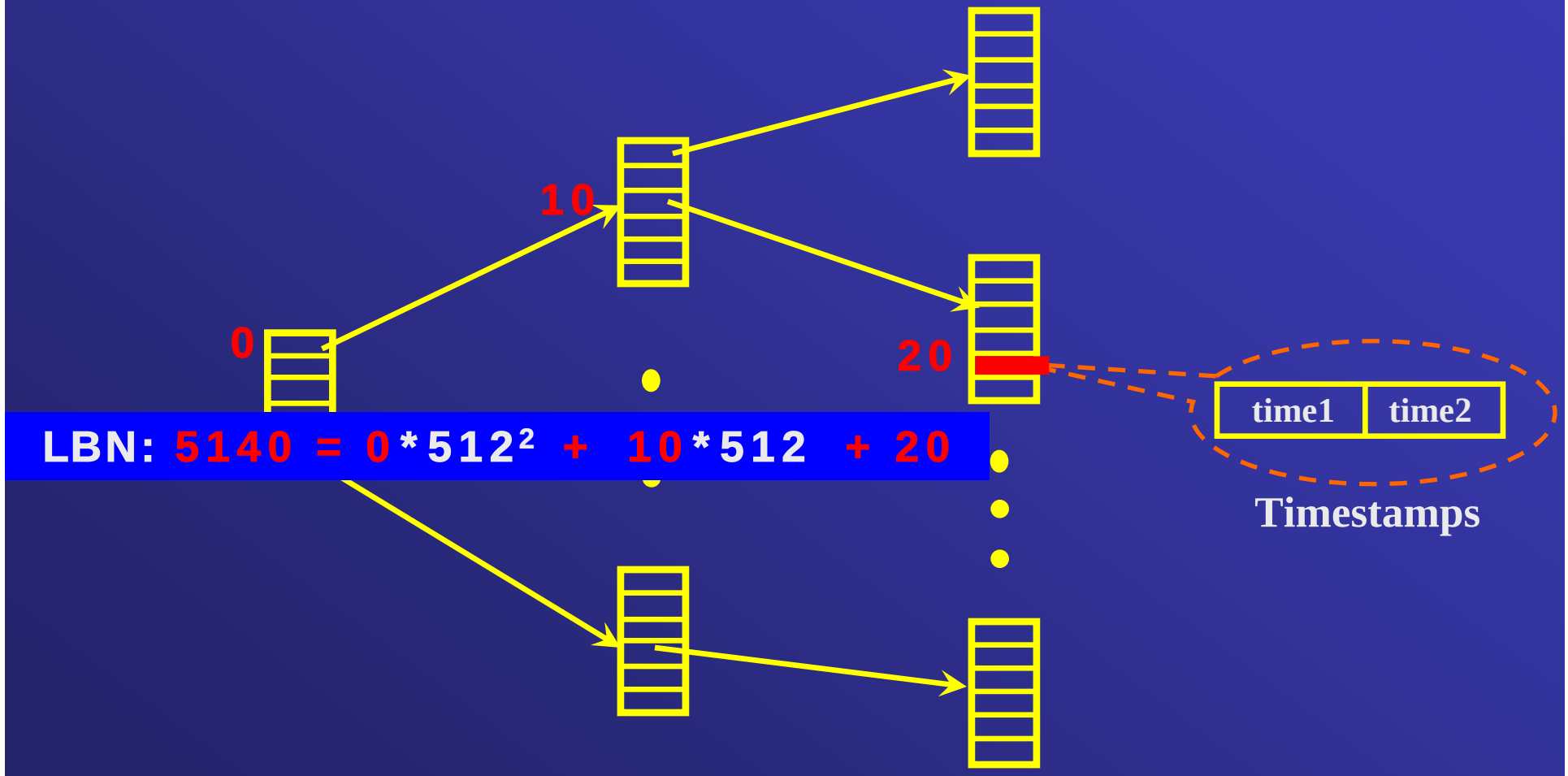


DULO-Caching Principles

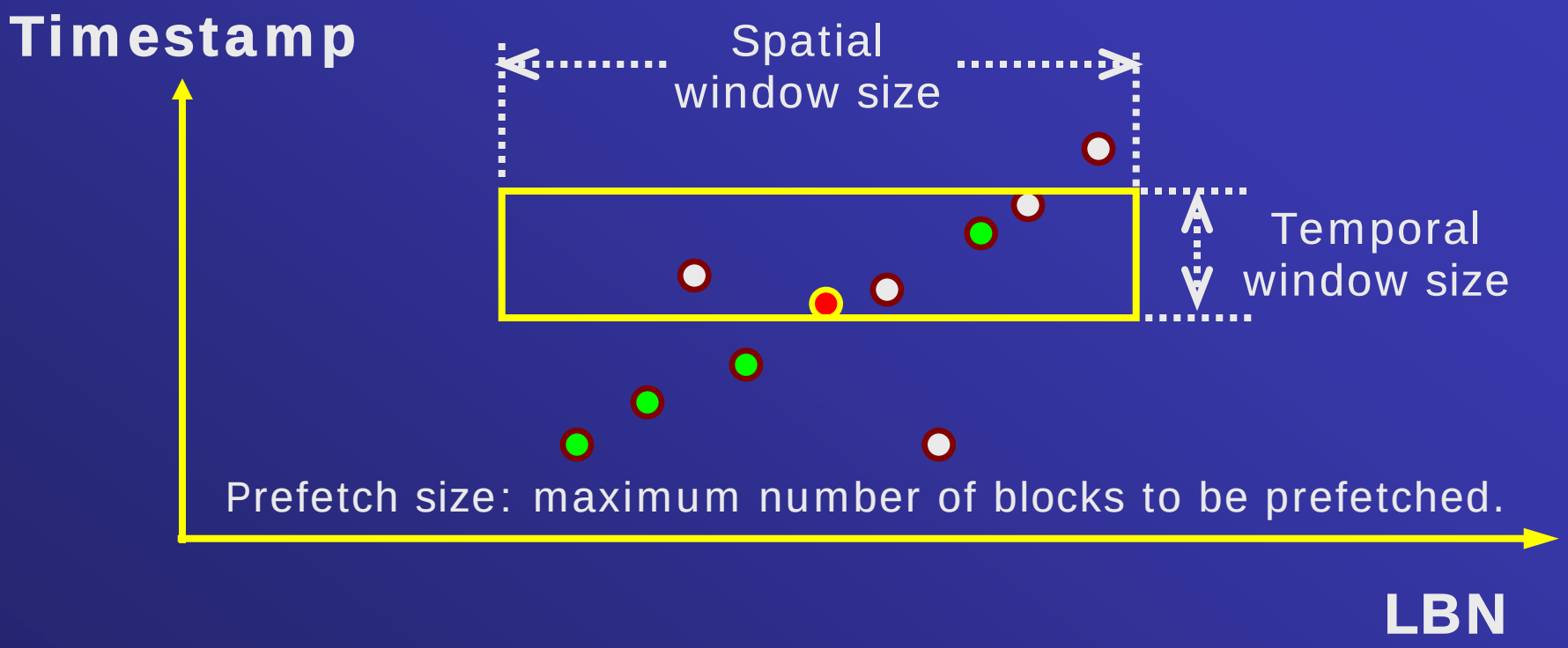
- Moving long sequences to the bottom of stack
  - replace them early, get them back fast from disks
- Replacement priority is set by sequence length.
- Moving LRU sequences to the bottom of stack
  - exploiting temporal locality of data accesses
- Keeping random blocks in upper level stack
  - hold them: expensive to get back from disks.

Disk-Seen Task 4: Identifying Long Disk Sequence

a data structure for tracking disk blocks

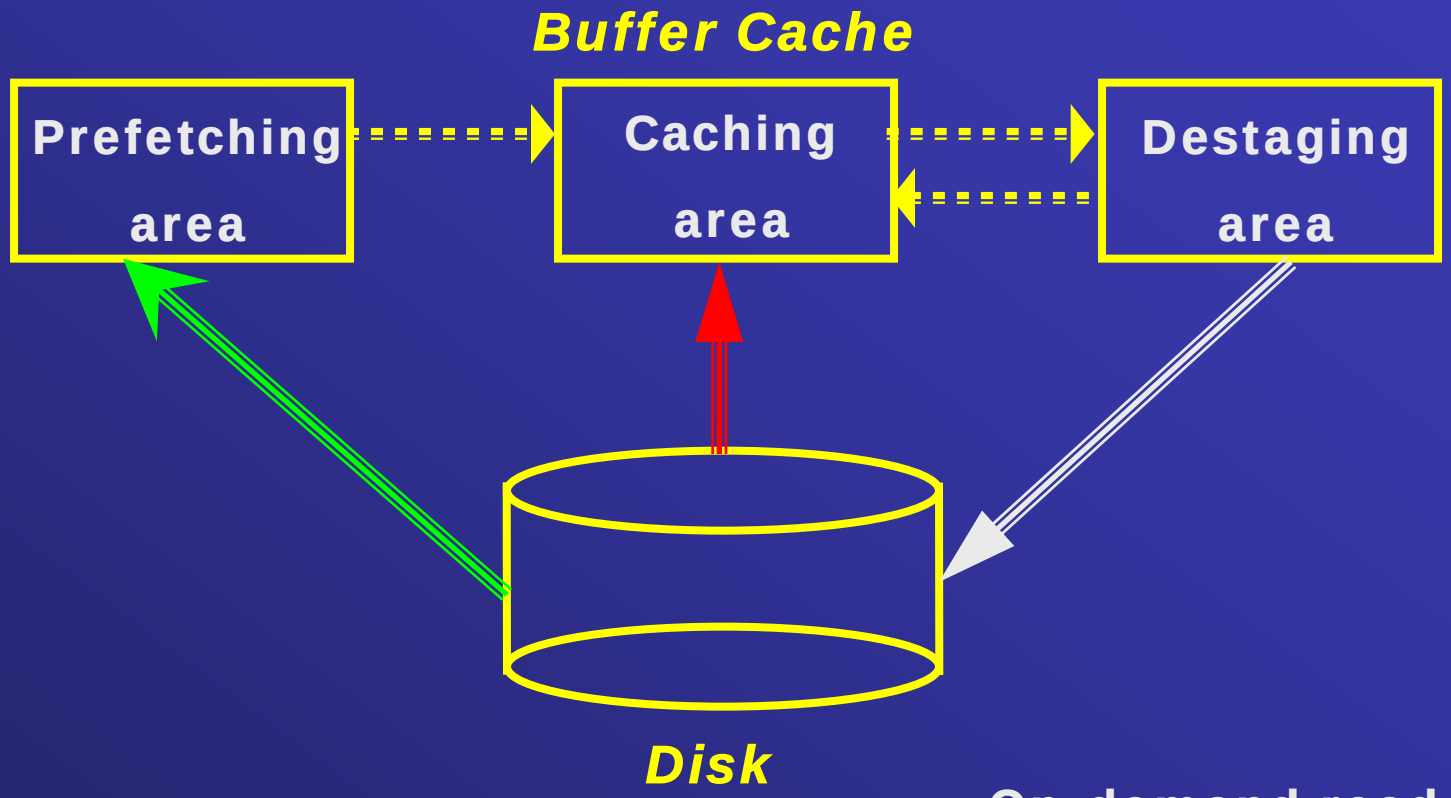


Disk-Seen Task 5: DULO-Prefetching



- Block initiating prefetching
- Resident block
- Non-resident block

DiskSeen: a System Infrastructure to Support DULO-Caching and DULO-Prefetching



Block transfers between areas

On-demand read: place in stack top

DULO-Prefetching: adj. window / stream

DULO-Caching: LRU blks and Long seqs.

Conclusions

- Disk performance is limited by
  - Non-uniform accesses: fast sequential, slow random
  - OS buffer management is Disk-layout unaware.
- The buffer cache is a critical component for I/O.
  - temporal locality of program execution is used only.
- Building a Disk-Seen system infrastructure for
  - DULO-Caching
  - DULO-Prefetching
- The size of the block table is 0.1% (4 K block) of disk capacity. Its working set can be in buffer cache.

References

- LIRS: buffer cache replacement, SIGMETRICS'02.
- ULC: multi-level storage caching, ICDCS'04.
- Clock-Pro: Linux VM page replacement, USENIX'05.
- DULO-caching: a prototype and its results, FAST'05.
- SmartSaver: Saving disk energy by Flash M, ISLPED'06
- Measurements of BitTorrent, SIGCOMM IMC'05.
- Measurements of streaming quality, SIGCOMM IMC'06.
- DULO-prefetching: OS kernel enhancement, USENIX'07.