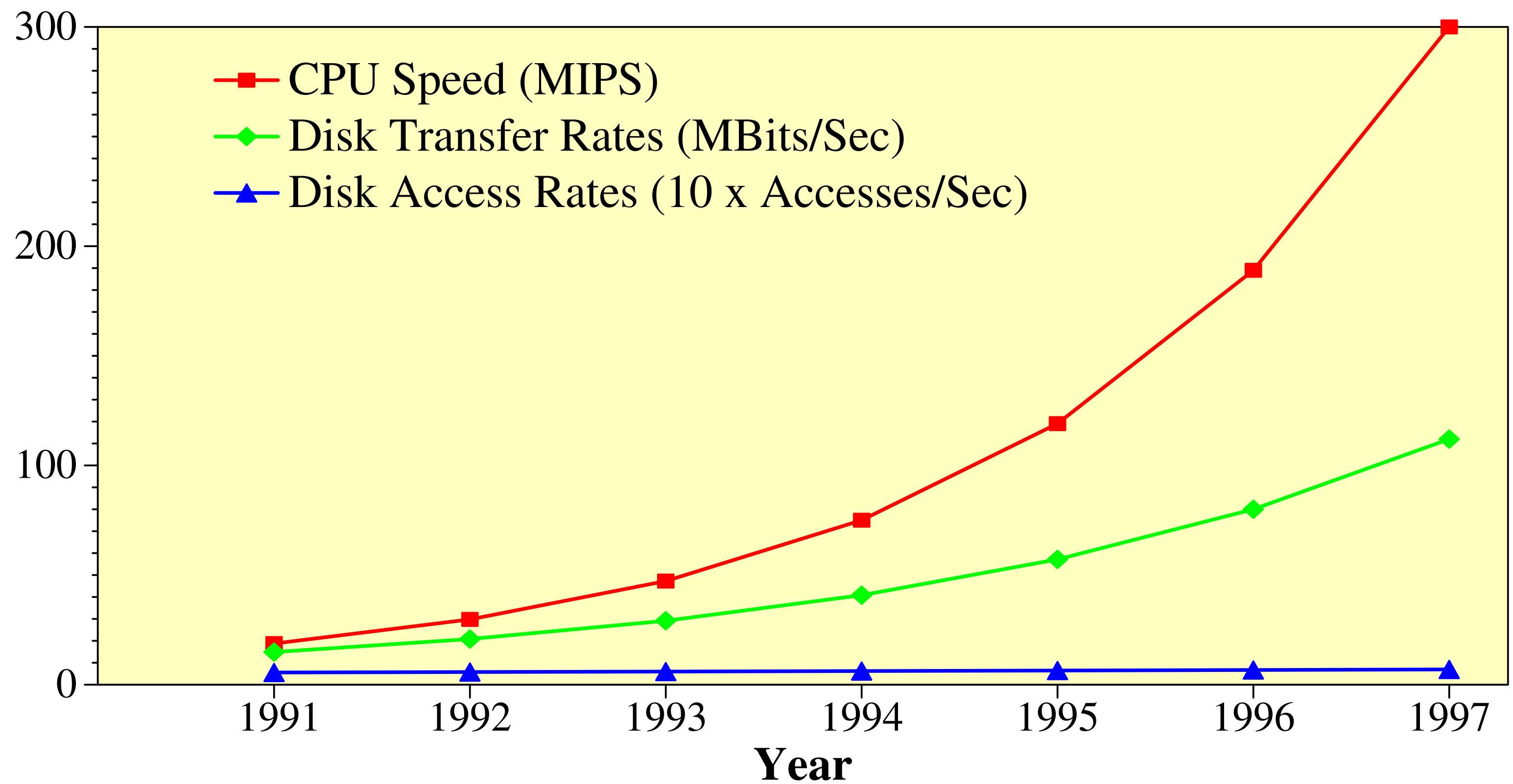
Informed Multi-Process Prefetching and Caching

Andrew Tomkins
R. Hugo Patterson
Garth Gibson

Carnegie Mellon University
Parallel Data Lab

The I/O Bottleneck



- **Processor Speeds** **Double Every 18 Months**
- **Disk Transfer Rates** **Increase 40% Each Year**
- **Disk Access Times** **Increased 30% over 7 years**

Carnegie Mellon

Parallel Data Laboratory

PDL Approach

Storage Parallelism (disk arrays, RAID layouts) provides high bandwidth to secondary storage

but how do applications access this parallelism?

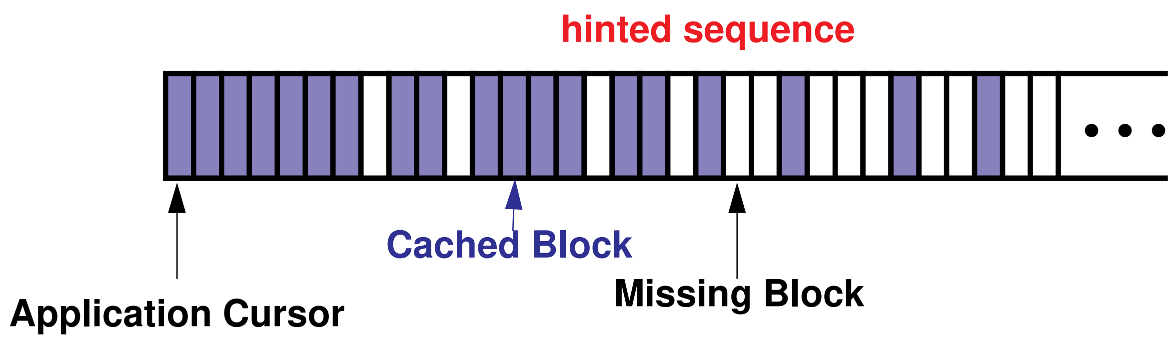
With knowledge of future accesses ("hints") multiple I/O's can proceed in parallel

but this solution requires software support...

Need Prefetching and Caching Algorithms That Incorporate Hints about Future I/O

[PDIS94, SOSP95: Patterson et al; OSDI94, Sigmetrics 95: Cao et al]

Prior Work: Single-Stream Prefetching



A Prefetching Algorithm Must:

1) Be aggressive: prefetch missing blocks in time

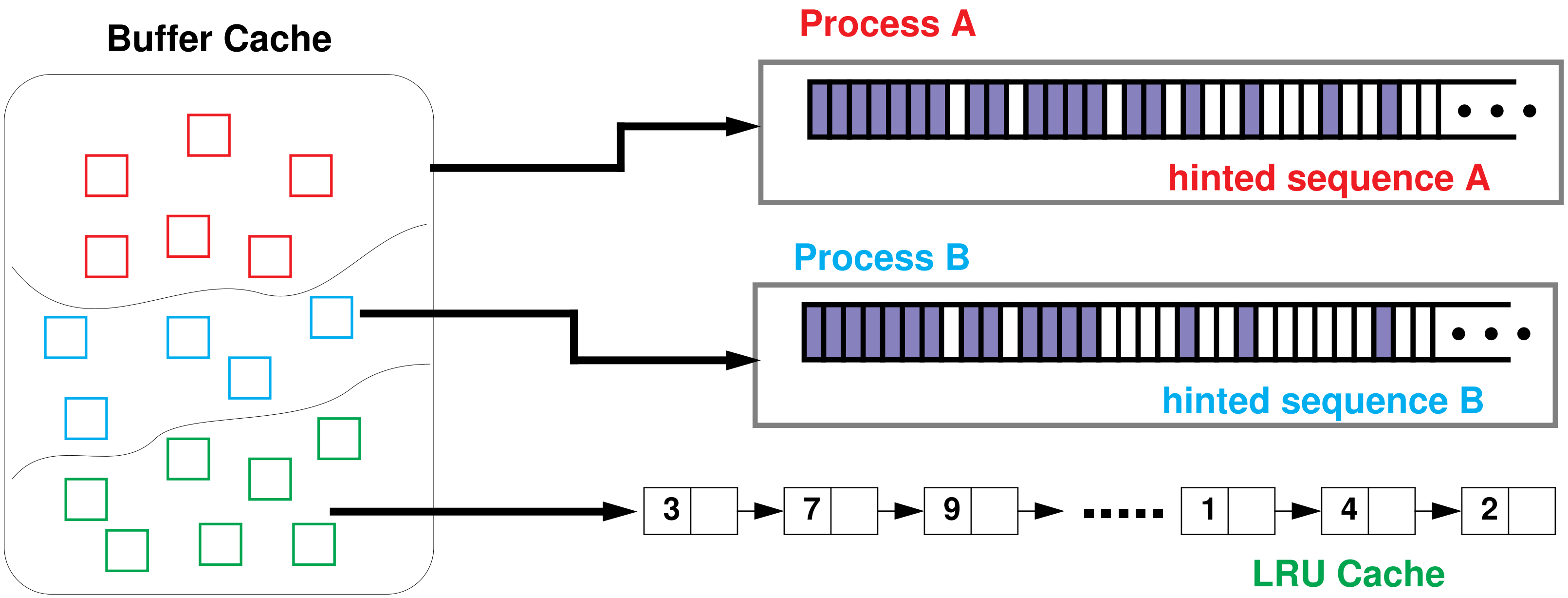
2) Be conservative: delay prefetching when no good eviction choices exist

Result: Adaptive Prefetching

Depth of Prefetching Regulated by Load on Disks

[OSDI96: Kimbrel et al]

This Talk: Multi-Process Prefetching



Each Cache Section Wants More Buffers

1) Deeper Prefetching for Load Imbalances

2) Caching for Larger Working Sets

Our Result: Adaptive Allocation

Allocation Regulated by Re-Use and Disk Load

Outline

Compare Two Allocation Approaches:

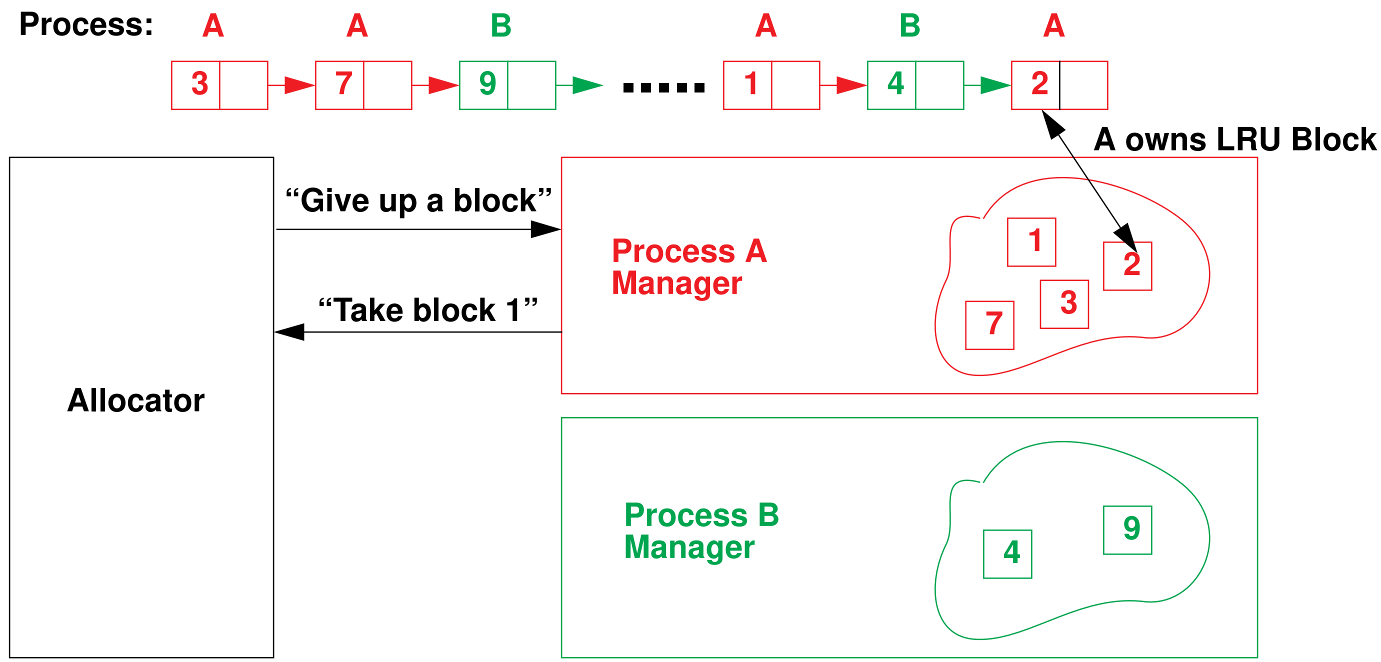
1) Allocation Based on Data Rate

2) Allocation Based on Cost-Benefit Analysis

Experimental Approach: Trace-Driven Simulation

Results: Cost-Benefit Analysis Performs Better When Re-Use Is Not Tracked by Data Rate

Rate-Based Allocation



- Allocator chooses a process to lose a block
- Process decides which block

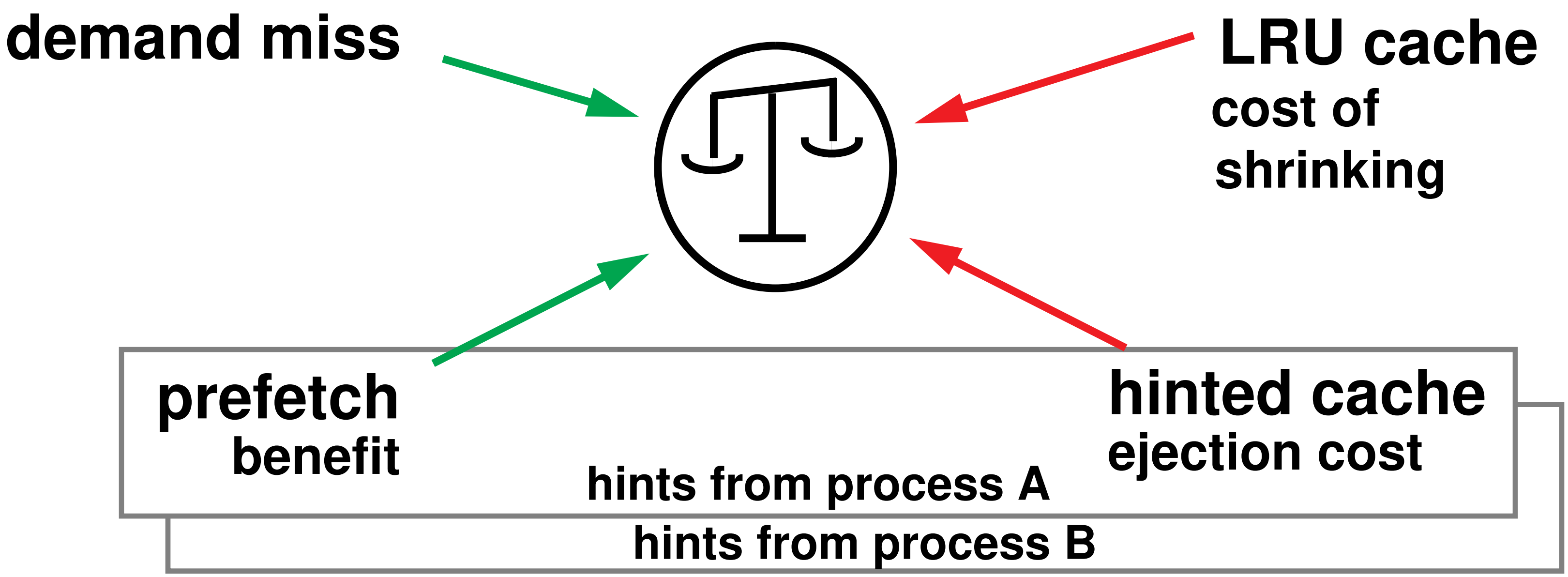
"LRU-SP" [Summer Usenix 94: Cao et al] with Adaptive Prefetching

Allocation Corresponds to Consumption Rate

Carnegie Mellon

Parallel Data Laboratory

Cost-Benefit Allocation



1) **Estimate:**
   - *benefit* of using a buffer to initiate I/O
   - *cost* of taking a buffer from a cache

2) **Reallocate when** *benefit* > *cost*

"TIP2" [SOSP95: Patterson et al] with Adaptive Prefetching

Allocation Corresponds to Value

Methodology: Trace-Driven Simulation

1) Suite of Six I/O-Intensive Applications drawn from diverse fields, modified to emit hints

[SOSP95: Patterson et al]

2) Example Applications

- *Davidson* computational physics sequentially reads 17 MB file 61 times
- *XDataSlice* 3-D scientific visualization reads arbitrary slices from 500 MB dataset
- *Postgres* relational database
  - sequential read of 3.2 MB relation
  - random reads of index and data of 32 MB relation

3) Traces

- Traced read, write, open, close, context switches and hints.

Methodology, Continued

4) Test Equipment

- DEC Alpha (21064), 175 MHz, 128MByte memory, One HP 2247 1GByte disk

5) Simulator

- Accurate Simulation of Multiple Disks (HP 97560) and Multiple Processes

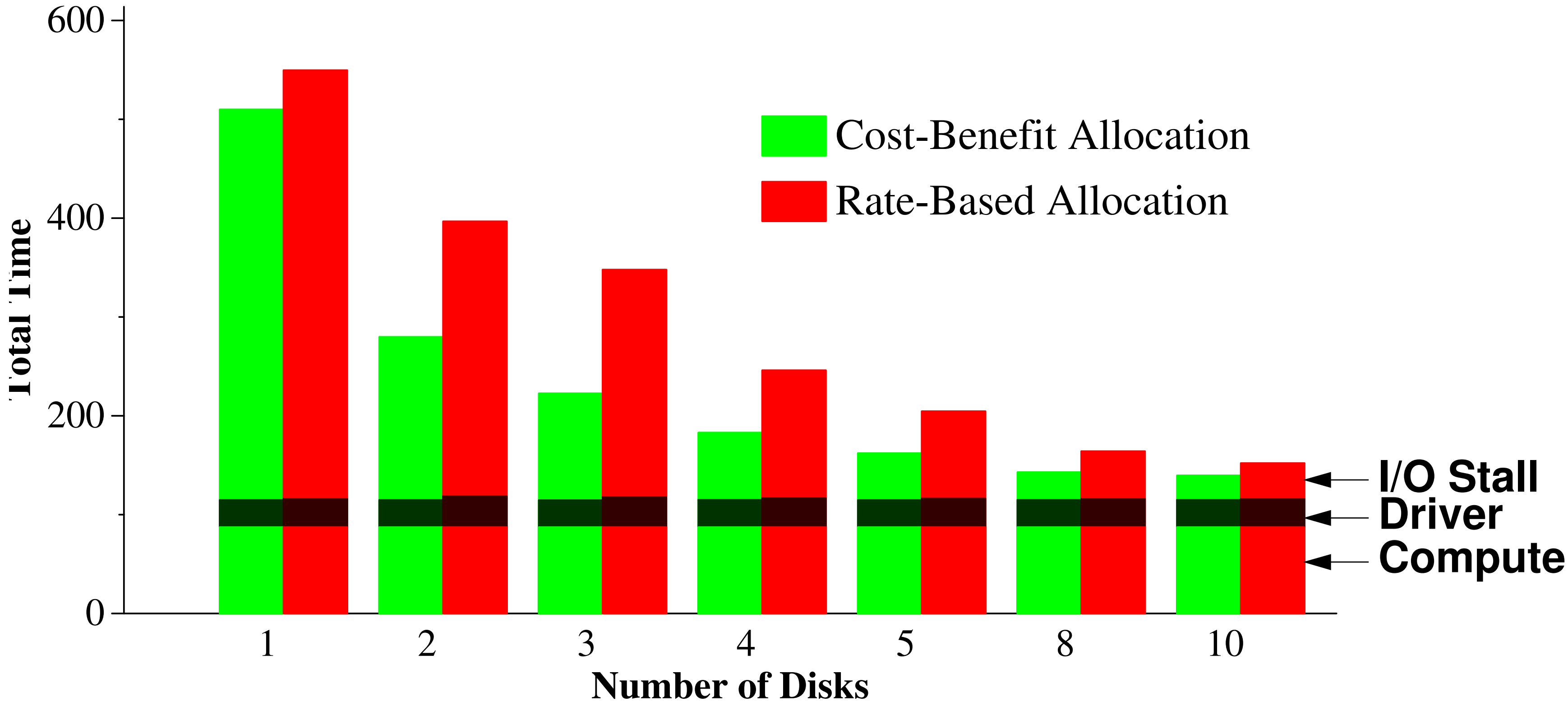
6) Experiments

- Pairs of Traces Chosen for Similarity in Duration

7) Metric

- Total Completion Time

LRU Cache Needs Buffers



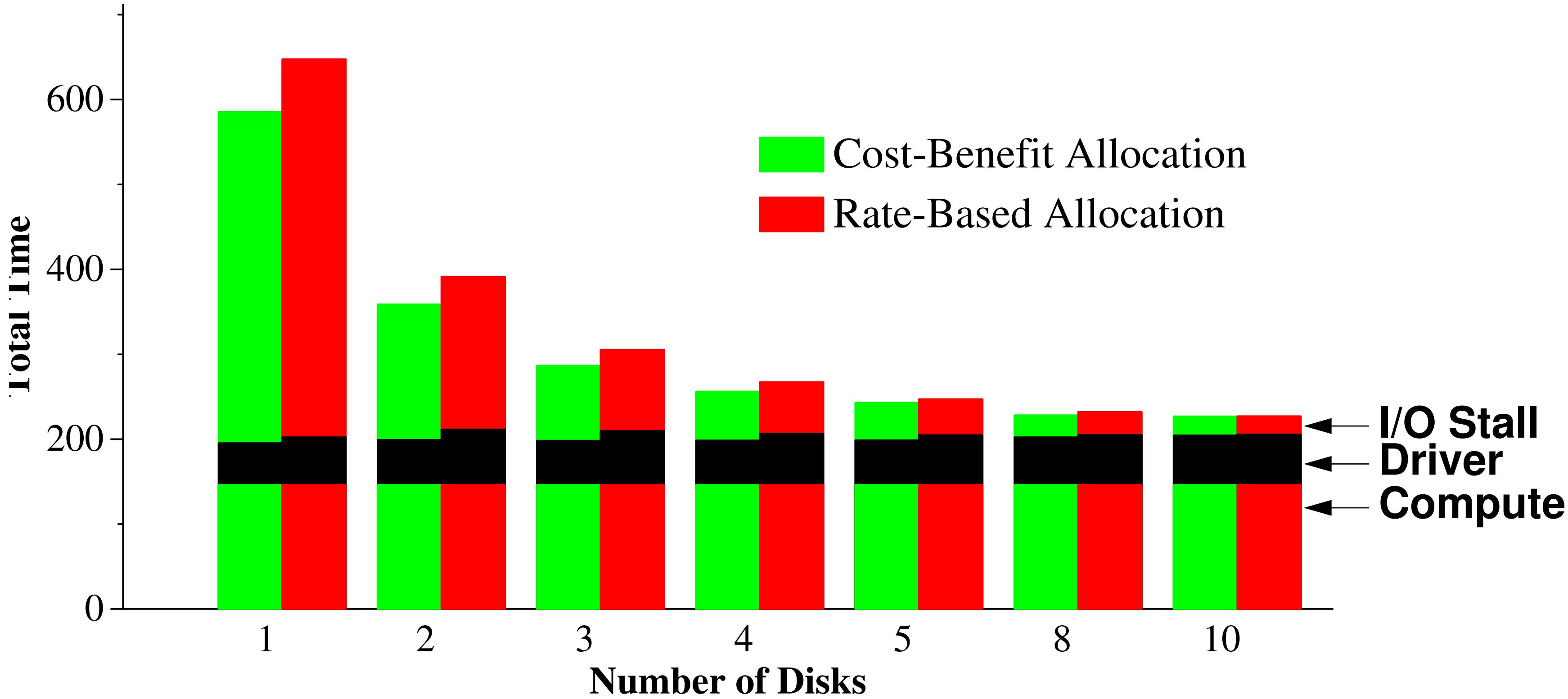
Postgres: Low Rate, High Unhinted Re-use

XDataSlice: High Rate, Low Re-use

Carnegie Mellon

Parallel Data Laboratory

Davidson Hinted Cache Needs Buffers



Davidson: High Rate, High Hinted Re-use

XDataSlice: High Rate, Low Re-use

Conclusions

1) Hint-Aware Allocation Algorithms for Multi-Process Prefetching and Caching Must Be “Adaptive” --- They Must Suit Allocation to Re-use Characteristics

2) Cost-Benefit Allocation Adapts to Different Re-use Situations Better than Rate-Based Allocation.