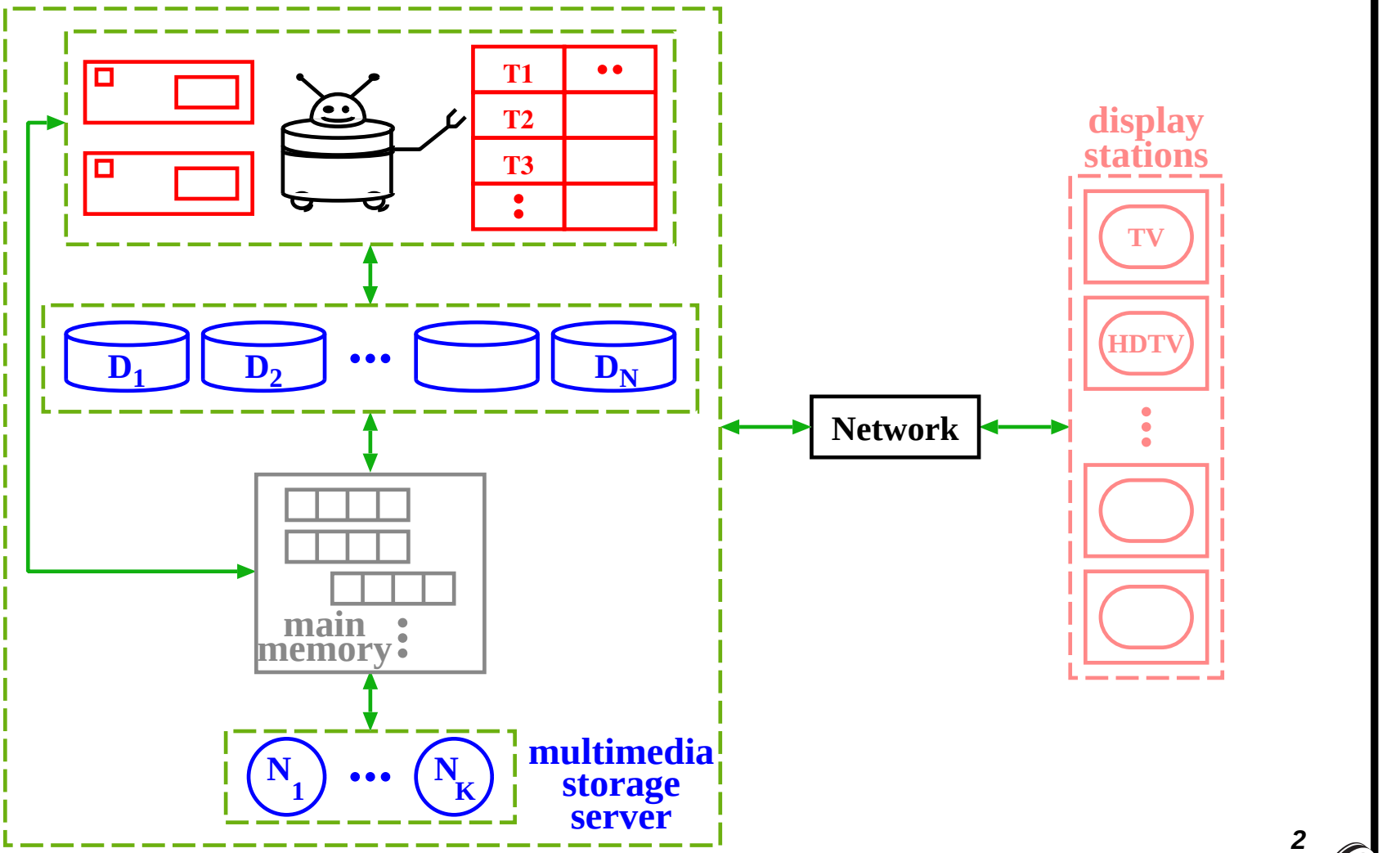
CUHK

Reducing I/O Demand in Video-On-Demand Storage Servers*

John C.-S. Lui
Department of Computer Science and Engineering
The Chinese University of Hong Kong

***These viewgraphs have been edited from the originals to fit within the presentation guidelines of UGC's visit.**

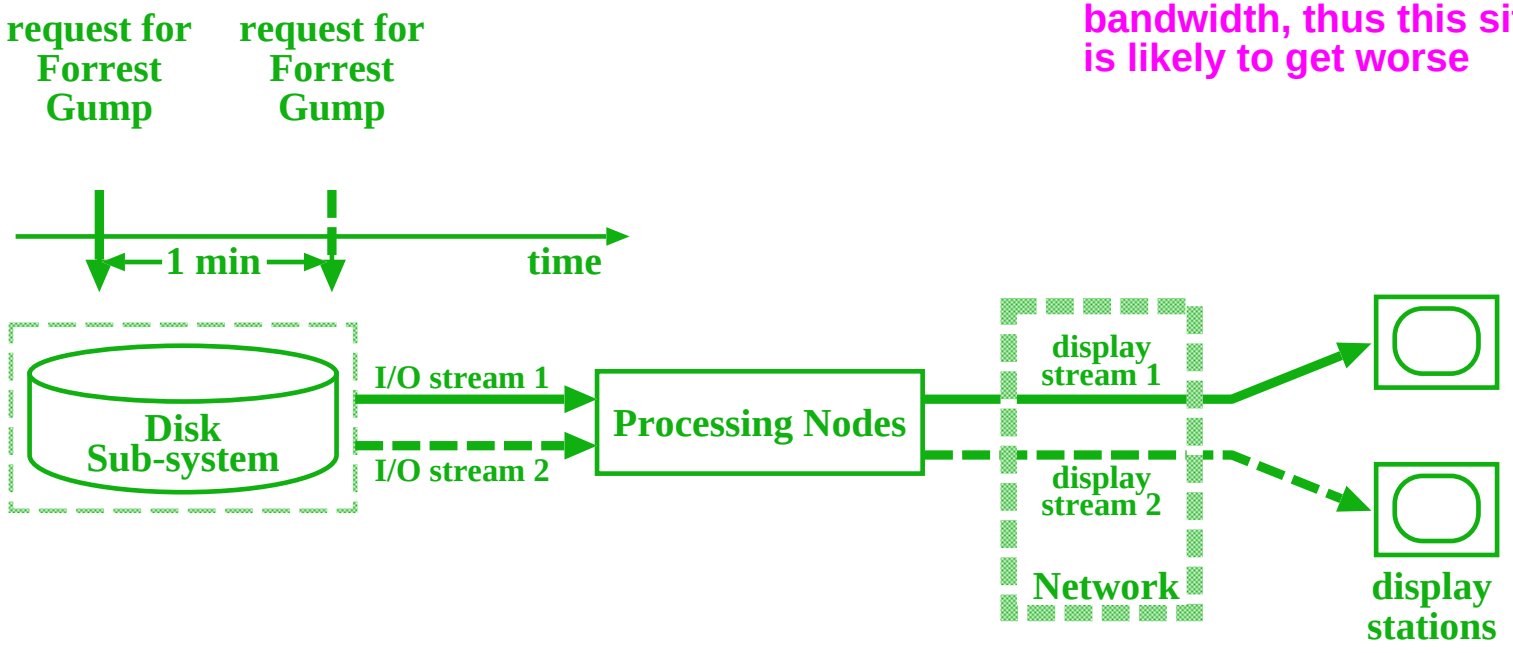
System


T1
T2
T3
D1
D2
DN
main memory
N1
NK
multimedia storage server
Network
display stations
TV
HDTV

Assumptions

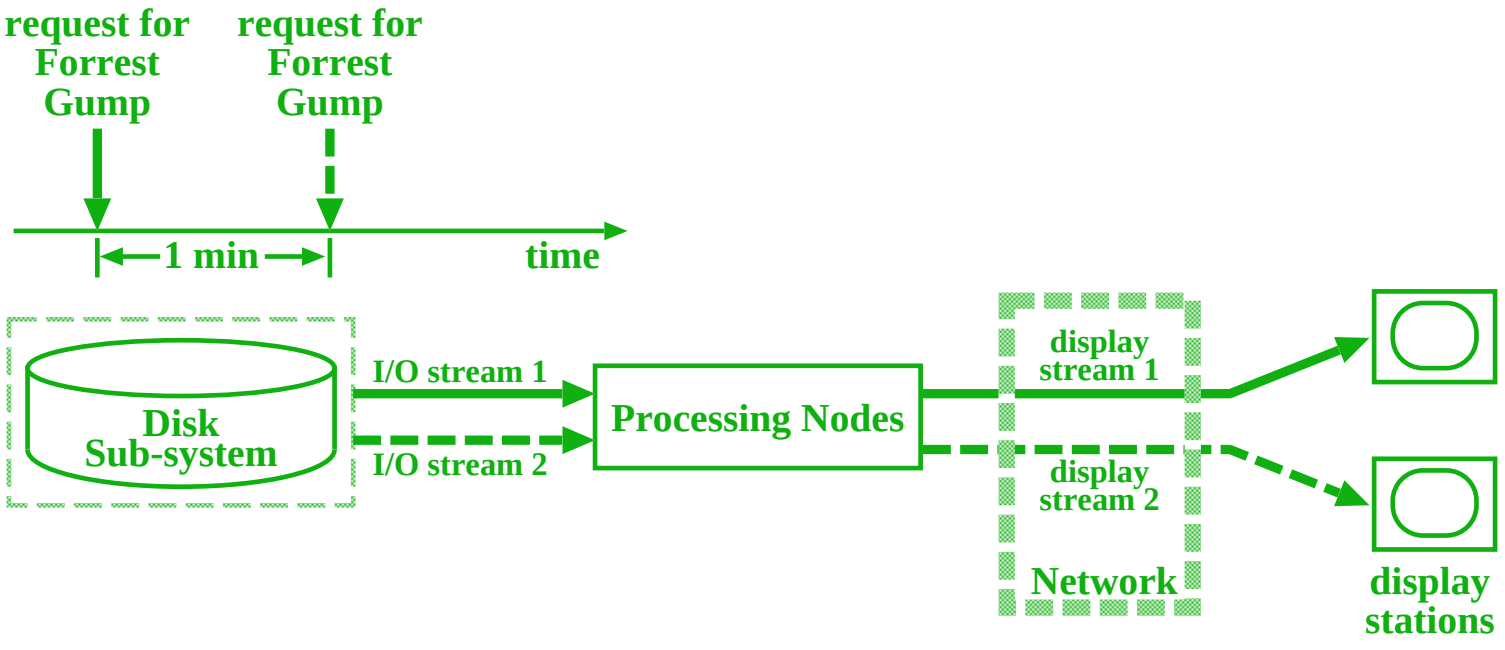
- Have a small fraction of objects that are frequently accessed
- All "popular" objects reside on secondary storage
- I/O bandwidth is the critical resource that contributes to increases in latency

disk capacity is increasing more rapidly than disk bandwidth, thus this situation is likely to get worse



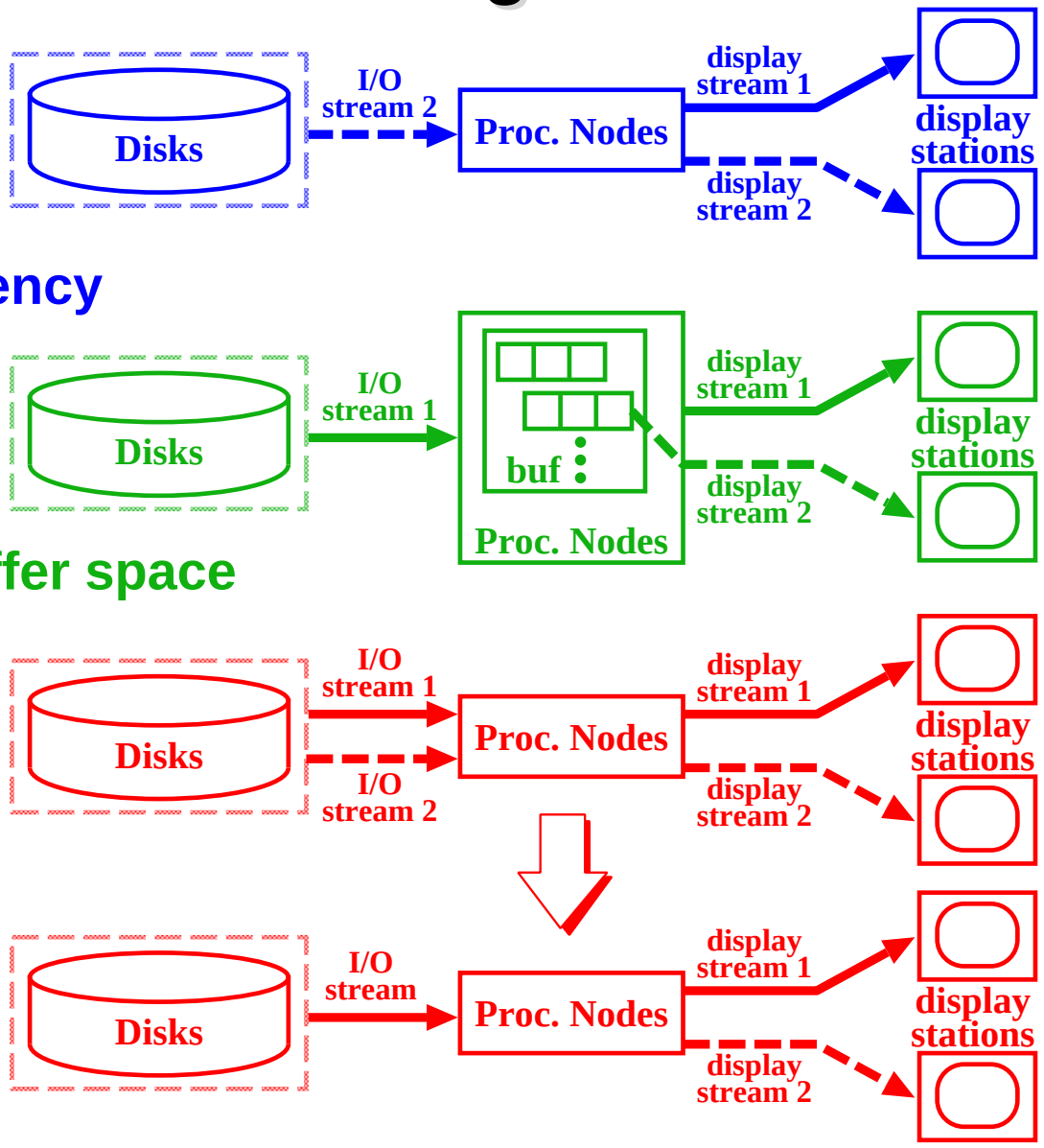
Choices

- Simply purchase more disk
- Improve data layout & scheduling techniques
- Reduce I/O demand of each request in service through "data sharing"



Data Sharing

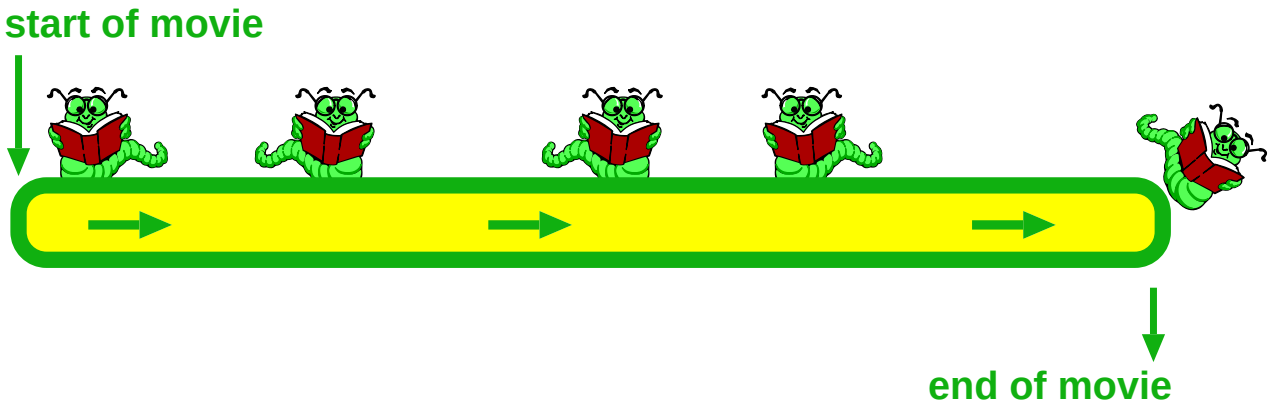
- **Batching**
  [Dan, et. al]
  - drawback: additional latency
- **Buffering**
  [Kamath, et. al]
  - drawback: additional buffer space
- **Our approach**



Our Approach

("adaptive piggybacking")

- **Adjusting display rates of requests *in progress* until their corresponding I/O streams can be "merged" into one**



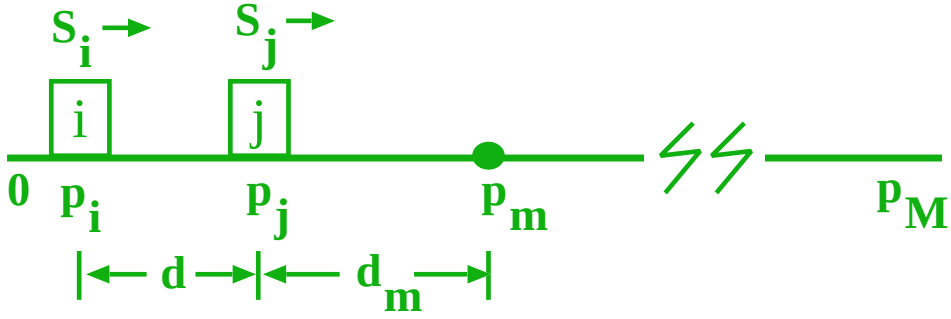
- **Adaptive Piggybacking**

Altering Display Rates

- Done every day!
- Audio is not a problem either!
- Video display rates can be altered by ± 5% without sacrificing video quality
  - for example, Zeus™ (an Ampex product)
- On-the-fly rate alteration
  - computationally intensive
  - could complicate data layout and scheduling
- Pre-canned versions: fast, normal, slow
  - requires additional storage space

Adaptive Piggybacking (cont.)

- **When can streams *i* and *j* be merged in such a way as to reduce the total I/O demand?**



- **Constraints:**
  - **cost**

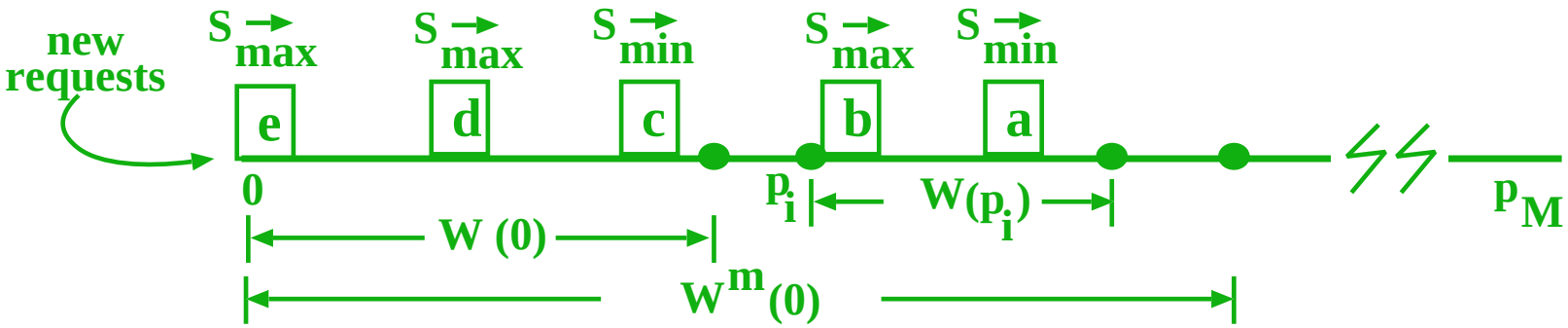
$$\frac{dC_i}{S_i}+\frac{d_mC_i}{S_i}+\frac{d_mC_j}{S_j}+\frac{(p_M-d-d_m-p_i)C_j^*}{S_j^*} \leq \frac{(p_M-p_i)C_i'}{S_i'}+\frac{(p_M-p_i-d)C_j'}{S_j'}$$

  - **distance**
  - **time**

$$\frac{p_i+d+d_m}{S_i} \leq p_M \qquad \frac{d+d_m}{S_i} = \frac{d_m}{S_j}$$

Policies

- **Odd-even reduction:** pair up consecutive arrivals (for merging), whenever possible
- **Simple merging:** assign streams to ‘‘merging groups’’
  - *i* initiates the group, and all streams that arrive while *i* is in $W(0)$ eventually merge with *i*
- **Greedy:** start as in the odd-even reduction policy, but
  - after the initial merge, continue pairing up streams for merging whenever possible



Performance Analysis

Odd-even reduction:

$$BW_{oe}^{m} = (t_m + x)C_{min} + t_m C_{max} + t_f C_n$$

$$BW_{oe} = \frac{\left(\int_{T}^{\frac{W_{oe}(0)}{S_{min}}} \left(BW_{oe}^{m} * \frac{N}{2} * f_{t_a}(x)\right)\, dx + \int_{\frac{W_{oe}(0)}{S_{min}}}^{\infty} \left(BW_{oe}^{nm} * \frac{N}{2} * f_{t_a}(x)\right)\, dx\right)}{\frac{p_M}{S_n}}$$

Simple merging:

$$\beta = \max\left\{\lfloor \frac{W_{sm}(0)/S_{min}}{T + 1/\lambda} + 1 \rfloor, 2\right\}$$

$$BW_{sm} = \frac{\int_{T}^{\frac{W_{sm}(0)}{S_{min}}} \left(BW_{sm}^{m} * \frac{N}{\beta} * f_{t_a}(x)\right)\, dx + \int_{\frac{W_{sm}(0)}{S_{min}}}^{\infty} \left(BW_{sm}^{nm} * \frac{N}{\beta} * f_{t_a}(x)\right)\, dx}{\frac{p_M}{S_n}}$$

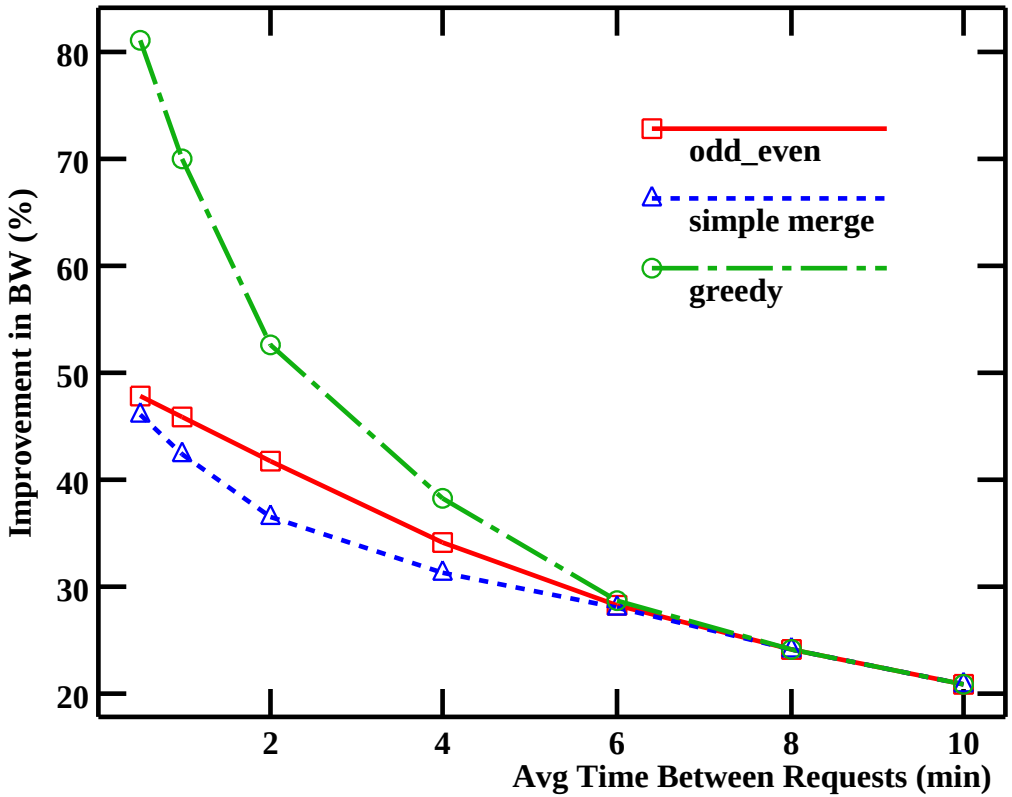
Greedy:

$$l = \max\left\{k : p_M - 2^{(k-1)}\tau \left[S_n + \left(\frac{S_{min}}{S_{max} - S_{min}}\right) S_{min}\right] > 0\right\}$$

$$\tau = \int_{T}^{\frac{W_g(0)}{S_{min}}} x f_{t_a}(x)\, dx = \left[T + \frac{1}{\lambda}\right] - \left[\frac{(S_{min} + \lambda W_g(0))e^{(-\lambda(W_g(0) - S_{min}T)/S_{min})}}{\lambda S_{min}}\right]$$

$$BW_g^m = \left[\frac{(t_m + x)C_{min} + t_m C_{max}}{p_M/S_n}\right]\left[\frac{N}{2}\right] + \sum_{j=2}^{l}\left[h(j)\left[\frac{S_n}{p_M}\right]\left[\frac{N}{2^j}\right]\right] + \left[\frac{t_f C_n}{p_M/S_n}\right]\left[\frac{N}{2^l}\right]$$

Benefits of Adaptive Piggybacking



Parameters:

S_{min} = 28.5 frames/sec

S_{n} = 30.0 frames/sec

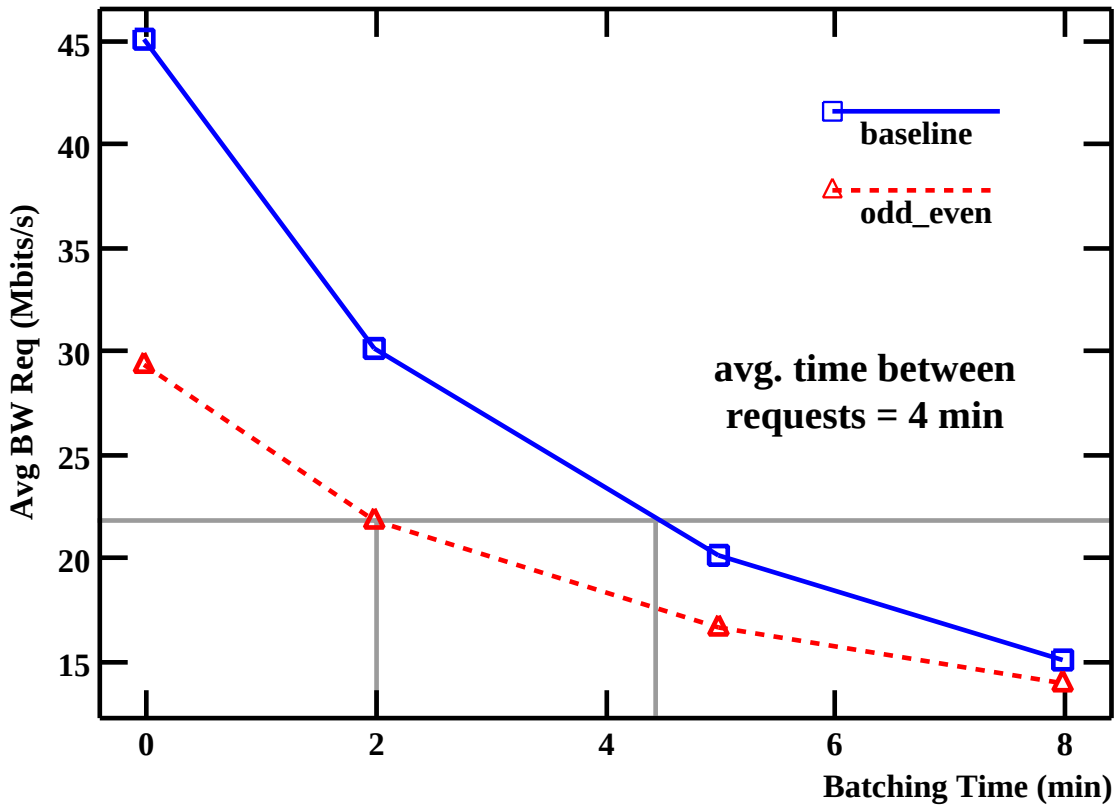
S_{max} = 31.5 frames/sec

C_{min} = 1.425 Mbits/sec

C_{n} = 1.500 Mbits/sec

C_{max} = 1.575 Mbits/sec

Including Batching



Parameters:

S_{min} = 28.5 frames/sec

S_{n} = 30.0 frames/sec

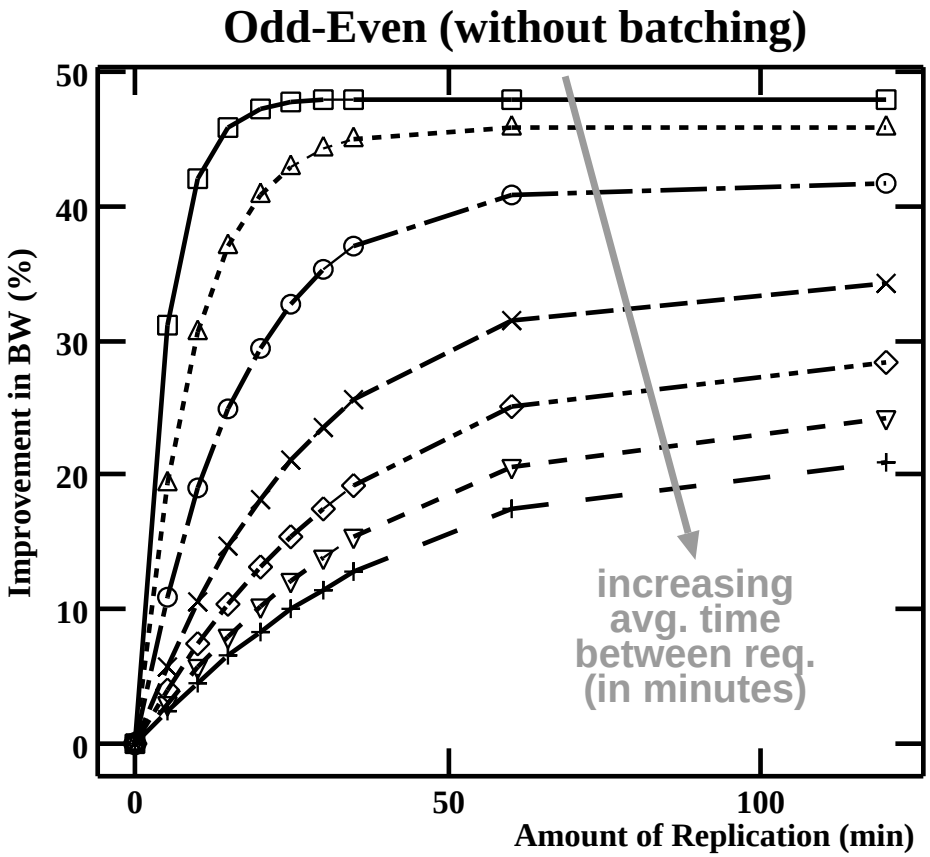
S_{max} = 31.5 frames/sec

C_{min} = 1.425 Mbits/sec

C_{n} = 1.500 Mbits/sec

C_{max} = 1.575 Mbits/sec

Limited Merging



Constrain merging to occur before a specified maximum merging time

Most of the reduction in I/O demand can be achieved using relatively small maximum merging times

Summary

- Presented a novel technique for data sharing (adaptive piggybacking) in on-demand multimedia storage servers, which results in significant reduction in I/O bandwidth requirements
- Current Research Work:
  - study of the tradeoff between the ‘‘optimal’’ merging and some other heuristic policies
  - study buffering policies (in conjunction with batching & adaptive piggybacking)

- Efficient storage servers will make a multitude of real-time & interactive storage and bandwidth intensive applications possible