Algorithms for Big Data Fall 2017

Lecture 15— November 27, 2017

Prof. Xiaotie Deng *Scribes: Zhenxing Gu*

1 Sorting Bigdata

1. All dirrernt sorting algorithm
   - bubble sort
   - insertion sort
   - merge sort
   - quick osrt
2. Quik sort
   (a) Randomly pick one item P, split the data into two part according to $A_0 \leq P < A_1$.
   (b) Quick sort A_0,A_1
3. Extension to partition sort
   (a) Suppose we have 4 computers. Each computer randomly pick 4 items, then distribute to all other computer.Each computer partition local data into sorted pieces delimited by the selected items

$$A_{11} < A_{12} <... < A_{116}$$
$$\vdots$$
$$A_{41} < A_{42} <... < A_{416}$$

   (b) Distribute $A_{i1}, A_{i2}, A_{i3}, A_{i4}$ to computer i.
   (c) Each computer sort data locally.
4. Locality efficiency in computation
   - Trade off between local computation and communication
   - Weakness: Data are moved around

2 Memory I/O wrt Disk access

1. Ideal-cache model
   - Cache size: Z words, Lines of length: L
   - Tall Memory: Z=$\Omega(L^2)$

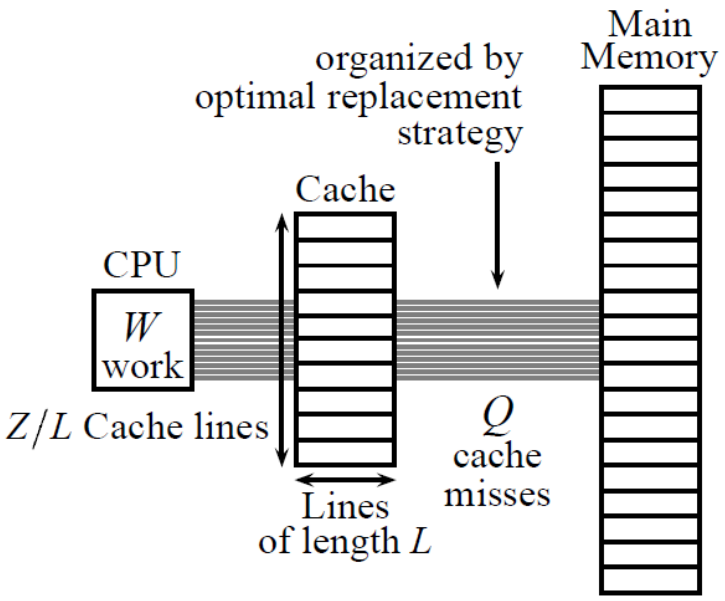

Figure 1: The ideal-cache Model

2. Funnel Sort

   (a) Split the input into $n^{1/3}$ contiguous arrays of size $n^{2/3}$, sort these arrays recursively.

   (b) Merge $n^{1/3}$ sorted subsequence using a $n^{1/3}$-merger,which is described below.

3. K-Merger

   (a) A k-merger is built recursively out of $\sqrt{k}$-mergers.

   (b) The k inputs sequence are partitioned into $\sqrt{k}$ set of $\sqrt{k}$ elements, which from the input to the $\sqrt{k}$ $\sqrt{k}$-mergers $L_1, L_2, ..., L_{\sqrt{k}}$ in the left of the figure 3.

   (c) The outputs of these mergers are connected to the inputs of $\sqrt{k}$ buffers. Each buffer can hold $2k^{3/2}$ elements.

   (d) The outputs of the buffers are connected to the $\sqrt{k}$ inputs of the $\sqrt{k}$-merger R in the right part of the figure. The output of this final $\sqrt{k}$-merger becomes the output of the whole k-merger.

   (e) The intermediate buffers are maintained half-full which is necessary for the correct behavior of algorithm

4. Cache Misses

   - Each page fault leads to L data into cache,each page fault cause 1/L cost.
   - g(L) < k: Do recursion
   - g(L) < k: Load all data in cache of size Z
   - If $Z = \Omega(L^2)$,then a k-merger operates with at most

     $$Q_M(k) = O(1 + k + k^3/L + k^3 \log_z k/L$$

     cache misses

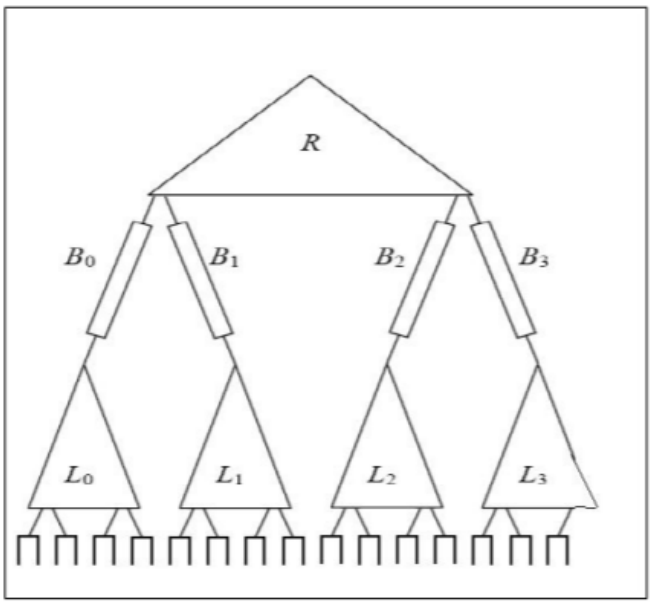

Figure 2: A 16-funnel with 16 input streams

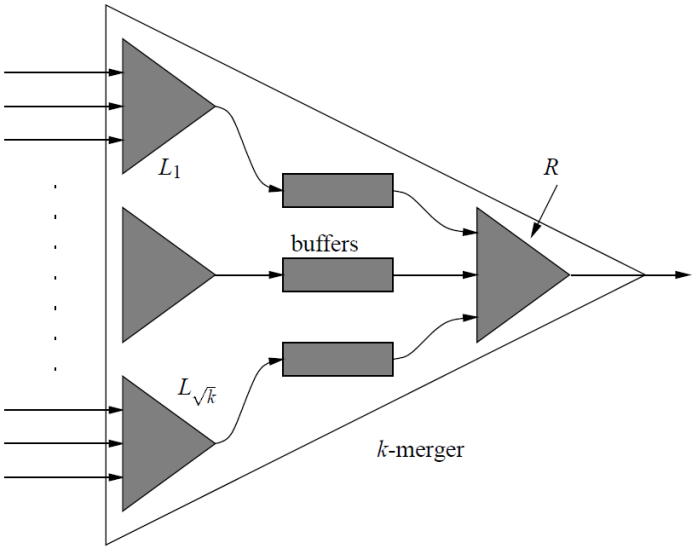

Figure 3: Illustration of a K-merger.

3 REFERENCE

- Matteo Frigo Charles E. Leiserson Harald Prokop Sridhar Ramachandran, Cache-Oblivious Algorithms.