MapReduce

M = Massive Data

Mapper(M) -> {(key,value)}

Shuffle({(key,value)}) -> group by "key"

Reducer ("key,value_i") -> ("key, f(value_i))

Can repeat, constant # of rounds

------------------------

Today: Simulate EREW PRAM in MR
       Simulate CRCW PRAM in MR
       Simulate BSP in MR
       + algorithms...

------------------------

MUD (Feldman, Muthukrishnan, Sidiropoulos, Stein, Svitkina 2008)

M = O(log^c n)

Linear sketch streaming algorithms can be simulated in MR

------------------------

Karloff, Suri, Vassilvistskii 2010

M = O(n^{1-eps})
P = O(n^{2-eps})

Simulate EREW PRAM with MR
   in MR P = O(n^{1-eps})
R = O(log^c n)

----

MST in MR
Minimum spanning tree of graph G=(V,E)
   works with E=O(V^2)

   - Partition V into sets V_i s.t. |V_i| = N/k
- on each pair $V_i \cup V_j$,
  consider all edges $(v_1,v_2)=e$ in $E$ s.t. $v_1,v_2$ in $V_i \cup V_j$
- Return MSF on each $V_i \cup V_j$, discard other edges.

"filter" (preview)

---------------------
Goodrich, (Sitchinava, Zhang) 2011

Simulate CRCR PRAM and BSP with MR

$R = \# \text{ rounds}$

$n_{r,i} =$ size I/O of mapper/reducer $i$ in round $r$

$C_r = \sum_i n_{r,i}$

$C = \sum_{r=0}^{R-1} C_r \quad \text{== communication complexity}$

$t_r =$ internal running time for round $r$

$\geq \max_i \{n_{r,i}\}$

$t = \sum_{r=0}^{R-1} t_r \quad \text{== total running time}$

$L =$ latency of shuffle (number of steps mapper or reducer waits for shuffle)

$B =$ bandwidth of shuffle network

$\quad \# \text{ elements delivered in unit of time (like block in I/O)}$

Total time $T = \Omega(t + RL + C/B)$

word count has $R=1$, $C=\Theta(n)$, $t=\Theta(n)$

"the" occurs $7\%$ of time $= \Theta(n)$

$M =$ I/O buffer memory size: require $n_{r,i} \leq M$

$T = \Omega(R(M+L) + C/B)$

rounds + work in PRAM

Let $M = \Theta(n^{\epsilon})$ for $\epsilon > 0$

then algorithms can run in $O(\log_M N)$ rounds, a constant!

------

Any BSP algorithm in $R$ super-steps, with memory size of $N$ and $P \leq N$ processors
$\rightarrow$ simulated in MR in $R$ rounds with $C = O(RN)$ with $M = O(N/P)$

Any CRCW PRAM (including sum on concurrent write)
with T steps w/ P processors, memory size N
-> simulated in MR in R = \( O(T \log_M P) \) rounds
    \( C = O(T(N+P)\log_M(N+P)) \) comm. complex.

Key idea: think of computation in the (dynamic) DAG model.
... edges defined based on data.

--------
Prefix sum in 2 \( \log_M N \) rounds with \( N \log_M N \) communication
    each element has (a_i, i) a_i=value, i=order
   return (i, \( \sum_{j=1}^i a_i \))

Just like PRAM/BSP algorithm, but with M-way split tree
   stage 1 (\( \log_M N \) rounds) : sum of all items
   stage 2 (\( \log_M N \) rounds) : filter down using partial prefix sums

key trick is to split indexes into chunks of size M each round

Can be extended when index values i are not consecutive and N not known whp.

-----------------
MultiSearch in R=\( O(\log_M N) \) and CC=\( O(N \log_M N) \)
   N searches on N data items

Sorting in R=\( O(\log_M N) \) and CC=\( O(N \log_M N) \)