Last Time ...

- Map Reduce
  - Overview
  - Matrix multiplication
  - Complexity theory
Today …

- Assignment 1 – deadline extended – due Oct 6
- Complexity theory for MapoReduce
- Page Rank
Complexity Theory for mapreduce
Reducer size & Replication rate

- **Reducer size** ($q$)
  - Upper bound on the number of values that are allowed to appear in the list associated with a single key
    - By making the reducer size small, we can force there to be many reducers
      - High parallelism $\rightarrow$ low wall-clock time
    - By choosing a small $q$ we can perform the computation associated with a single reducer entirely in the main memory of the compute node
      - Low synchronization (Comm/IO) $\rightarrow$ low wall clock time

- **Replication rate** ($r$)
  - Number of $(k, v)$ pairs produced by all the Map tasks on all the inputs, divided by the number of inputs
  - $r$ is the average communication from Map tasks to Reduce tasks
Graph model for mapreduce problems

- Set of inputs
- Set of outputs
- many-many relationship between the inputs and outputs, which describes which inputs are necessary to produce which outputs.

- Mapping schema
  - Given a reducer size $q$
  - No reducer is assigned more than $q$ inputs
  - For every output, there is at least one reducer that is assigned all input related to that output
Grouping for Similarity Joins

- Generalize the problem to $p$ images
- $g$ equal sized groups of $\frac{p}{g}$ images
- Number of outputs is $\binom{p}{2} \approx \frac{p^2}{2}$
- Each reducer receives $\frac{2p}{g}$ inputs ($q$)
- Replication rate $r = g - 1$

\[ r = \frac{2p}{q} \]

- The smaller the reducer size, the larger the replication rate, and therefore higher the communication
  - communication $\leftrightarrow$ reducer size
  - communication $\leftrightarrow$ parallelism
1. Prove an upper bound on how many outputs a reducer with $q$ inputs can cover. Call this bound $g(q)$

2. Determine the total number of outputs produced by the problem

3. Suppose that there are $k$ reducers, and the $i^{th}$ reducer has $q_i < q$ inputs. Observe that $\sum_{i=1}^{k} g(q_i)$ must be no less than the number of outputs computed in step 2

4. Manipulate inequality in 3 to get a lower bound on $\sum_{i=1}^{k} q_i$

5. 4 is the total communication from Map tasks to reduce tasks. Divide by number of inputs to get the replication rate

$$r \geq \frac{p}{q}$$
Matrix Multiplication

- Consider the one-pass algorithm → extreme case
- Lets group rows/columns into bands → \( g \) groups → \( n/g \) columns/rows
Matrix Multiplication

- Map:
  - for each element of $M, N$ generate $g \ (k, v)$ pairs
  - Key is group paired with all groups
  - Value is $(i, j, m_{ij})$ or $(i, j, n_{ij})$

- Reduce:
  - Reducer corresponds to key $(i, j)$
  - All the elements in the $i^{th}$ band of $M$ and $j^{th}$ band of $N$
  - Each reducer gets $n \binom{n}{g}$ elements from 2 matrices

$$q = \frac{2n^2}{g}, \quad r = g \quad \Rightarrow \quad r = \frac{2n^2}{q}$$
Lower bounds on Replication rate

1. Prove an upper bound on how many outputs a reducer with \( q \) inputs can cover. Call this bound \( g(q) \).

2. Determine the total number of outputs produced by the problem.

3. Suppose that there are \( k \) reducers, and the \( i^{th} \) reducer has \( q_i < q \) inputs. Observe that \( \sum_{i=1}^{k} g(q_i) \) must be no less than the number of outputs computed in step 2.

4. Manipulate inequality in 3 to get a lower bound on \( \sum_{i=1}^{k} q_i \).

5. 4 is the total communication from Map tasks to reduce tasks. Divide by number of inputs to get the replication rate.

- Each reducer receives \( k \) rows from \( M \) and \( N \rightarrow q = 2nk \) and produces \( k^2 \) outputs \( \rightarrow g(q) = \frac{q^2}{4n^2} \).

- \( n^2 \)

- \( \sum_{i=1}^{k} q_i^2 \geq 4n^4 \)

- \( \sum_{i=1}^{k} q_i \geq \frac{4n^4}{q} \)

\[
 r = \frac{1}{2n^2} \sum_{i=1}^{k} q_i = \frac{2n^2}{q}
\]
Matrix Multiplication

LET US REVISIT THE TWO-PASS APPROACH
Matrix-vector multiplication

- $n \times n$ matrix $M$ with entries $m_{ij}$
- Vector $\mathbf{v}$ of length $n$ with values $v_j$
- We wish to compute
  \[ x_i = \sum_{j=1}^{n} m_{ij} v_j \]

- If $\mathbf{v}$ can fit in memory
  - Map: generate $(i, m_{ij} v_j)$
  - Reduce: sum all values of $i$ to produce $(i, x_i)$
- If $\mathbf{v}$ is too large to fit in memory? Stripes? Blocks?
- What if we need to do this iteratively?
Grouped two-pass approach

\[ g^2 \text{ groups of } \frac{n^2}{g^2} \text{ elements each} \]

First pass: compute products of square \((I,J)\) of \(M\) with square \((J,K)\) of \(N\)

Second pass: \(\forall I, K\) sum over all \(J\)
Grouped two-pass approach

- Replication rate for map1 is $g \rightarrow 2gn^2$ total communication
- Each reducer gets $\frac{2n^2}{g^2} \rightarrow q = \frac{2n^2}{g^2} \rightarrow g = n\sqrt{\frac{2}{q}}$
- Total communication $\rightarrow 2\frac{\sqrt{2n^3}}{\sqrt{q}}$
- Assume map2 runs on same nodes as reduce1 $\rightarrow$ no communication
- Communication $\rightarrow gn^2 \rightarrow \frac{\sqrt{2n^3}}{\sqrt{q}}$
- Total communication $\rightarrow 3\frac{\sqrt{2n^3}}{\sqrt{q}}$
Comparison

\[
\frac{n^4}{q} < \frac{n^3}{\sqrt{q}}
\]

If \( q \) is closer to the minimum of \( 2n \), two pass is better by a factor of \( \Theta(\sqrt{n}) \).
Page Rank
Webpage quality ranking

- Inverted web indexes help locate matching pages of search words
  - But there are too many matches and humans can’t read all
- Both relevance and quality are important in web search
- What is a high-quality web page?
- How to identify a high-quality web page?
  - Hard to spam
- Related to identifying high-quality scientific publications
  - But much bigger dataset
Page Rank

Transition matrix

\[
M = \begin{bmatrix}
0 & 1/2 & 1 & 0 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 0 & 0 & 1/2 \\
1/3 & 1/2 & 0 & 0
\end{bmatrix}
\]

\( \nu \rightarrow \) probability distribution for the location of a random surfer

\( \nu \leftarrow \frac{1}{n} \)

Iterate on \( \nu \leftarrow M\nu \)
Page Rank

- Markov process
  - Limiting distribution
  - will converge if
    - Strongly connected
    - No dead ends
- Limiting \( \mathbf{v} \) is an eigenvector of \( M \)
  - \( \mathbf{v} = \lambda M \mathbf{v} \)
  - \( \mathbf{v} \) is also the primary eigenvector
- Iterate a few times on \( \mathbf{v} \leftarrow M \mathbf{v} \) until \( \|v_{i+1} - v_i\| < \epsilon \)
Solving Linear Systems

- $Mx = y \Rightarrow x = M^{-1}y$

- Gaussian Elimination $\Rightarrow O(n^3)$
- Iterative approaches $\Rightarrow O(kn^2)$
  - For sparse systems $\Rightarrow O(kn)$
  - Use optimal solvers $\Rightarrow k$ independent of $n$
Structure of the Web

- Strongly Connected Component
- In Component
- Out Component
- Dead Ends
- Spider Traps
Dead Ends

- Remove dead ends from the graph
  - And incoming links
- Compute page-rank on strongly connected component
- Restore graph, retaining page ranks
- Use existing page ranks to compute ranks for dead-end nodes
Spider traps & Taxation

- modify the calculation of PageRank by allowing each random surfer a small probability of teleporting to a random page

\[ v' = \beta M v + \frac{(1 - \beta)e}{n} \]

- \( \beta \) is a constant that represents the probability that the surfer follows a link on the page
- Approach will still be biased towards spider traps