Map Reduce

Last Time …

- Parallel Algorithms
  - Work/Depth Model
- Spark
- Map Reduce
- Assignment 1

- **Questions ?**

Today …

- Map Reduce
  - Matrix multiplication
  - Similarity Join
  - Complexity theory

MapReduce – word counting

- Input → set of documents
- Map:
  - reads a document and breaks it into a sequence of words
    $$w_1, w_2, \ldots, w_n$$
  - Generates (k, v) pairs,
    $$(w_1, 1), (w_2, 1), \ldots, (w_n, 1)$$
- System:
  - group all (k, v) by key
  - Given r reduce tasks, assign keys to reduce tasks using a hash function
- Reduce:
  - Combine the values associated with a given key
  - Add up all the values associated with the word → total count for that word

Matrix-vector multiplication

- $n \times n$ matrix M with entries m_{ij}
- Vector $\boldsymbol{v}$ of length n with values v_j
- We wish to compute

$$x_i = \sum_{j=1}^{n} m_{ij} v_j$$

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

Matrix-Matrix Multiplication

- $P = MN \rightarrow p_{ik} = \sum_j m_{ij} n_{jk}$
- 2 mapreduce operations
  - Map 1: produce (k, v), $\left(j, (M, i, m_{ij})\right)$ and $\left(j, (N, k, n_{jk})\right)$
  - Reduce 1: for each $j \rightarrow \left(i, k, m_{ij} \times n_{jk}\right)$
  - Map 2: identity
  - Reduce 2: sum all values associated with key (i, k)

Matrix-Matrix multiplication

- In one mapreduce step
- Map:
  - generate $(k, v) \rightarrow \big((i, k), (M, j, m_{ij})\big)$ & $\big((i, k), (N, j, n_{jk})\big)$
- Reduce:
  - each key (i, k) will have values $\big((i, k), (M, j, m_{ij})\big)$ & $\big((i, k), (N, j, n_{jk})\big)$ $\forall j$
  - Sort all values by j
  - Extract m_{ij} & n_{jk} and multiply, accumulate the sum

Complexity Theory for mapreduce

Communication cost

- Communication cost of a task is the size of the input to the task
- We do not consider the amount of time it takes each task to execute when estimating the running time of an algorithm
- The algorithm output is rarely large compared with the input or the intermediate data produced by the algorithm

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 → 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) → 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

Example: one-pass matrix mult

- Assume matrices are $n \times n$
- r – replication rate
  - Each element m_{ij} produces n keys
  - Similarly each n_{jk} produces n keys
  - Each input produces exactly n keys → load balance
- q – reducer size
  - Each key has n values from M and n values from N
  - $2n$

Example: two-pass matrix mult

- Assume matrices are $n \times n$
- r – replication rate
  - Each element m_{ij} produces 1 key
  - Similarly each n_{jk} produces 1 key
  - Each input produces exactly 1 key (2nd pass)
- q – reducer size
  - Each key has n values from M and n values from N
  - $2n$ (1st pass), n (2nd pass)

Real world example: Similarity Joins

- Given a large set of elements X and a similarity measure $s(x, y)$
- Output: pairs whose similarity exceeds a given threshold t
- Example: given a database of 10^6 images of size 1MB each, find pairs of images that are similar
- Input: (i, P_i), where i is an ID for the picture and P_i is the image
- Output: (P_i, P_j) or simply (i, j) for those pairs where $s(P_i, P_j) > t$

Approach 1

- Map: generate (k, v)

$$\left((i,j),(P_i,P_j)\right)$$

- Reduce:
  - Apply similarity function to each value (image pair)
  - Output pair if similarity above threshold t

- Reducer size – q → 2 (2MB)
- Replication rate – r → $10^6 - 1$
- Total communication from map→reduce tasks?
  - $10^6 \times 10^6 \times 10^6$ bytes → 10^{18} bytes → 1 Exabyte (kB MB GB TB PB EB)
  - Communicate over GigE → 10^{10} sec → 300 years

Approach 2: group images

- Group images into g groups with $\frac{10^6}{g}$ images each
- Map: Take input element (i, P_i) and generate
  - $(g-1)$ keys $(u, v) \mid P_i \in \mathcal{G}(u),\ \ v \in \{1, \dots, g\} \setminus \{u\}$
  - Associated value is (i, P_i)
- Reduce: consider key (u, v)
  - Associated list will have $2 \times \frac{10^6}{g}$ elements (j, P_j)
  - Take each (i, P_i) and (j, P_j) where i, j belong to different groups and compute $s(P_i, P_j)$
  - Compare pictures belonging to the same group
    - heuristic for who does this, say reducer for key $(u, u+1)$

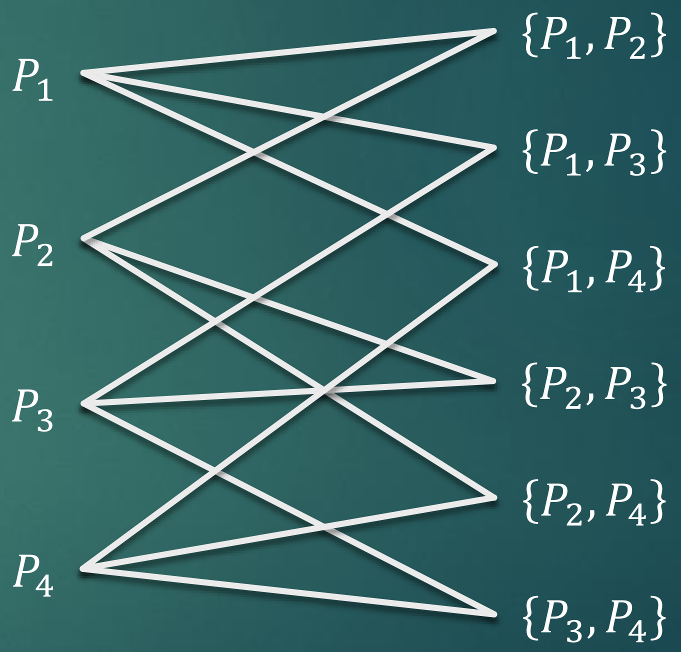
Approach 2: group images

- Replication rate: $r = g - 1$
- Reducer size: $q = 2 \times 10^6 / g$
- Input size: $2 \times 10^{12} / g$ bytes
- Say $g = 1000$,
  - Input is 2GB
  - Total communication: $10^6 \times 999 \times 10^6 = 10^{15}$ bytes → 1 petabyte

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 ↔ reducer size
  - communication ↔ parallelism

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

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

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

$$\binom{q}{2} \approx \frac{q^2}{2}$$

$$\binom{p}{2} \approx \frac{p^2}{2}$$

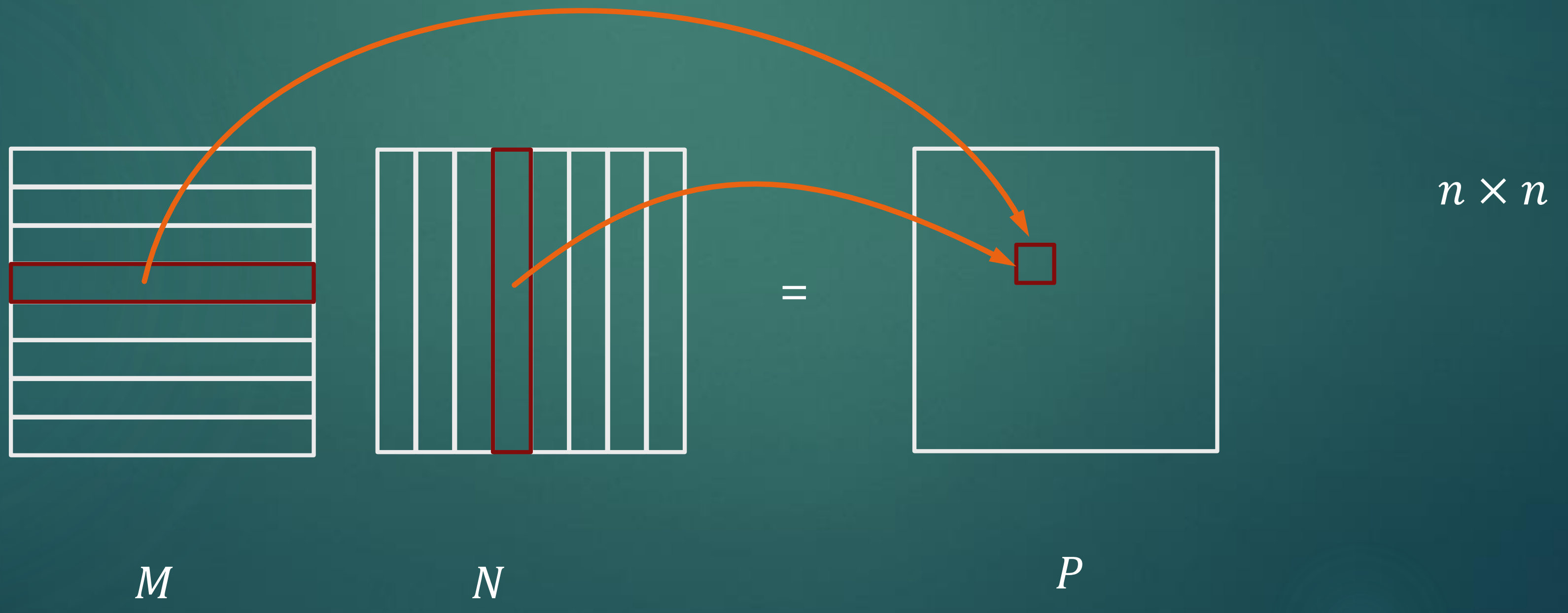
$$\sum_{i=1}^{k} \frac{q_i^2}{2} \geq \frac{p^2}{2}$$

$$q \sum_{i=1}^{k} q_i \geq p^2$$

$$\sum_{i=1}^{k} q_i \geq \frac{p^2}{q}$$

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 $n\left(\frac{n}{g}\right)$ elements from 2 matrices
  - $q = \frac{2n^2}{g}$, $r = g$ → $r = \frac{2n^2}{q}$

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 → $q = 2nk$ and produces k^2 outputs → $g(q) = \frac{q^2}{4n^2}$
- n^2
- $\sum_{i=1}^{k} \frac{q_i^2}{4n^2} \geq n^2$

$$\sum_{i=1}^{k} q_i^2 \geq 4n^4$$

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

$$r = \frac{1}{2n^2} \sum_{i=1}^{k} q_i = \frac{2n^2}{q}$$

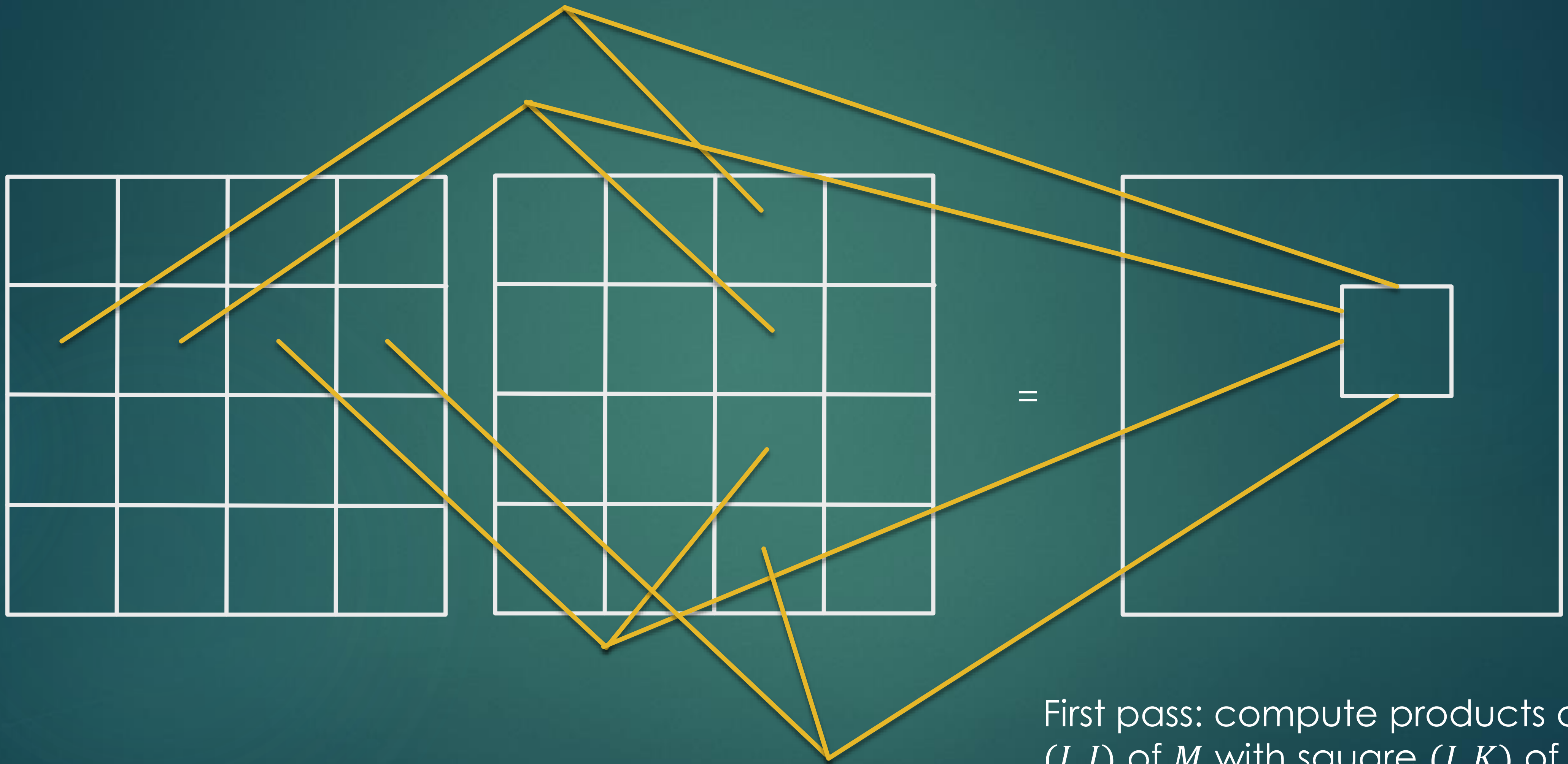
Matrix Multiplication

LET US REVISIT THE TWO-PASS APPROACH

Matrix-Matrix Multiplication

- $P = MN \rightarrow p_{ik} = \sum_j m_{ij} n_{jk}$
- 2 mapreduce operations
  - Map 1: produce (k, v), $\left(j, (M, i, m_{ij})\right)$ and $\left(j, (N, k, n_{jk})\right)$
  - Reduce 1: for each $j \rightarrow \left(i, k, m_{ij} \times n_{jk}\right)$
  - Map 2: identity
  - Reduce 2: sum all values associated with key (i, k)

Grouped two-pass approach



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

g^2 groups of $\frac{n^2}{g^2}$ elements each

Grouped two-pass approach

- Replication rate for map1 is g → $2gn^2$ total communication
- Each reducer gets $\frac{2n^2}{g^2}$ → $q = \frac{2n^2}{g^2}$ → $g = n\sqrt{\frac{2}{q}}$
- Total communication → $2\frac{\sqrt{2}n^3}{\sqrt{q}}$

- Assume map2 runs on same nodes as reduce1
  → no communication
- Communication → gn^2 → $\frac{\sqrt{2}n^3}{\sqrt{q}}$
- Total communication → $3\frac{\sqrt{2}n^3}{\sqrt{q}}$

Comparison

$$\frac{n^4}{q} < \frac{n^3}{\sqrt{q}}$$

$q < n^2$

If q is closer to the minimum of $2n$, two pass is better by a factor of $O(\sqrt{n})$