MS&E 317/CS 263: Algorithms for Modern Data Models, Spring 2014
`http://msande317.stanford.edu`.
Instructors: Ashish Goel and Reza Zadeh, Stanford University.

Lecture 3, 4/7/2014. Scribed by Susan Tu.

Lecture topics:

- Computing A^TA for tall/skinny and sparse A
- Computing SVD for A
- Combiners

4 Naive approach to computing A^TA

Let A be an $m \times n$ matrix, where $m \geq n$. Each row of A has at most l nonzero values. As a motivating example, suppose each column of A represented a movie, and each row a user. If the user likes a movie, a 1 is recorded in that column, otherwise it's a 0.

$$A = \begin{bmatrix} | & | & | & & | \\ a_1 & a_2 & a_3 & \dots & a_n \\ | & | & | & & | \end{bmatrix} \tag{1}$$

Then in order to find similar movies, we need to look for similar a_i, a_j. We can use cosine similarity, $cos(a_i, a_j)$, to measure this. (Recall the definition of cosine similarity: $cos(x, y) = \frac{x^T y}{|x|_2 |y|_2}$.)

With MapReduce, we probably want to slice A across machines at row boundaries. Let r_1, r_2, ..., r_m be the rows of A.

```
Map (r_i)
  for all (a_ij, a_ik) ∈ r_i
    Emit ((j,k) → (a_ij * a_ik)) // only emit if nonzero
Reduce ((j,k), <v_2, v_2, ..., v_R>)
  Output (Σ_{i=1}^{R} v_i)
```

The shuffle size is $O(ml^2)$ since there are m rows and each row has at most l^2 nonzero pairs. The reduce-key operation takes $O(m)$ time and space (consider two extremely similar movies) .

5 Probabilistic approach to computing A^TA

Let's modify the previous algorithm:

```
Map (r_i)
  for all (a_ij, a_ik) ∈ r_i
    With probability min(γ/(|a_j|_2 |a_k|_2), 1)} Emit ((j,k) → (a_ij * a_ik))
Reduce ((j,k), < v_2, v_2, ..., v_R >)
  if (γ/(||a_j|| ||a_k||) > 1)
    Output( 1/(||a_j|| ||a_k||) Σ_{i=1}^{R} v_i )
  else
    Output( 1/γ Σ_{i=1}^{R} v_i )
```

Intuitively, the more popular a column is, the less likely it is to be sampled.

Claim 5.1 $E[output] = \frac{1}{||a_j|| ||a_k||} a_j^T a_k$.

We're going to show that this adaptive sampling approach will remove the dependence on m.

Claim 5.2 *Expected shuffle size is* $O(\gamma n L)$

Let $\#(x, y)$ *be the number of dimensions in which both* x *and* y *are both nonzero.*

Let's compute the shuffle size (note that the innermost of the three summations is summing over all the mappers):

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{m} Pr[Emit((i,j))] &= \sum_{i=1}^{n}\sum_{j=1}^{n} \#(a_i, a_j)\frac{\gamma}{||a_i|| ||a_j||} \\
&\leq \sum_{i=1}^{n}\sum_{j=1}^{n} \#(a_i, a_j)\frac{\gamma}{\sqrt{\#(a_i)}\sqrt{\#(a_j)}} \\
&\leq \frac{\gamma}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \#(a_i, a_j)(\frac{1}{\#(a_i)} + \frac{1}{\#(a_j)}) && \textit{Using AM-GM} \\
&= \gamma \sum_{i=1}^{n}\sum_{j=1}^{n} \#(a_i, a_j)\frac{1}{\#(a_j)} \\
&= \gamma \sum_{i=1}^{n} \frac{1}{\#(a_i)} \sum_{j=1}^{n} \#(a_i, a_j) \\
&\leq \gamma \sum_{i=1}^{n} \frac{1}{\#(a_i)} \#(a_i) l && \textit{Using the sparsity assumption} \\
&= \gamma n l
\end{aligned}$$

Expected reduce-key space and time complexity is $O(\gamma)$.

6 Computing the SVD of A

How do we leverage the previous algorithm to compute the singular value decomposition of A? Recall that this means that we want to find U, Σ, and V where $A = U\Sigma V^T$. Since $A^T A =$

$V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T$, we can first compute V and Σ on a single machine. Both are reasonably small ($O(n^2)$ and $O(n)$). Now we can compute $U = AV\Sigma^{-1}$.

7 Setting γ

For how to set γ, see `http://stanford.edu/~rezab/papers/dimsum.pdf`. (For example, to preserve cosine similarities that are $\geq S$, set $\gamma = \Omega(\frac{log(n)}{S})$. To obtain the singular values of A within relative error ϵ, set $\gamma = \Omega(n^2/\epsilon)$.

8 Combiners

Combiners are an optimization that can be used when the reduce operation is commutative and associative (e.g., sum, min, max). They allow mappers to reduce as far as possible before communcating with other machines. For our $A^T A$ computation (done the naive way), for example, they allow us to emit a "partial" $n \times n$-dimensional $A^T A$. If there are k mappers, then, the shuffle size is $O(n^2 k)$.

References

[1] R. Zadeh and G. Carlsson "Dimension Independent Matrix Square using MapReduce." `http://stanford.edu/~rezab/papers/dimsum.pdf`