1. Intro
Map Reduce Reminder

Performance measures

Triangle Counting Sequentially

Triangle Counting on M.R.

Analysis: Shuffle Size, Redirecting complexity

2. Map Reduce
Mappers take in data and emit pairs

Reducers get all pairs with the same key

3. MR Bottlenecks, Performance Measures
Reduce-key complexity: traditional single machine work mode.

Shuffle size: (not used to this one). Total number of pairs emitted in the map phase.
Shuffling happens between the map and the reduce phase.

Given graph $G(V, E)$, $n$ nodes and $m$ edges, this graph is sparse. $m = O(n) = cn$.

Let “clustering coefficient” = number of triangles/$\binom{n}{3}$
where $\binom{n}{3}$ = number of possible triangles.

4. Counting triangles on a single machine: Node iterator algorithm
$T = 0$.
For each $v \in V$, for $u, w \in \Gamma^*(v)$, i.e. “pairs of nodes in neighborhood of $v$”
if $(u, w) \in E$ and $\deg(u) \leq \deg(v) \leq \deg(w)$
$T = T + 1$
Number of computations = $\sum_{v \in V} \binom{\deg(v)}{2}$
If highly connected node exists, then this is at least $\Omega(n^2)$.

Every triangle will be counted by the node with the lowest degree.

Want: $\deg(v) \leq \deg(w)$ and $\deg(v) \leq \deg(u)$

Define: $\Gamma^*$ as neighborhood of $v$ consisting of only higher degree nodes.

So now, don’t need this:

$$\deg(u) \leq \deg(v) \leq \deg(w) .$$

And do:

$$T = T + 1/2$$

Now, # of computations is:

$$\sum_{v \in V} \left( \frac{\deg^*(v)}{2} \right) .$$

Use threshold $t$:

$$\sum_{\deg(v) \leq t} \left( \frac{\deg^*(v)}{2} \right) \leq \sum_{\deg(v) \leq t} \deg^*(v)^2$$

$$\leq \sum_{\deg(v) \leq t} t \deg^*(v) \leq 2mt$$

There are at most $2m/t$ nodes with $\deg \geq t$.

$$\sum_{v \in V, \deg(v) > t} \left( \frac{\deg^*(v)}{2} \right) \leq \left( \frac{2m}{t} \right) ^3 .$$

Note: handshake lemma from graph theory $\sum_v \deg(v) = 2m$, and that $t$ is arbitrary.

$$\sum \left( \frac{\deg^*(v)}{2} \right) \leq \left( \frac{2m}{t} \right) ^3 + 2mt = O(m^{3/2})$$

So, runtime went from $O(n^2)$ to $O(m^{3/2})$ which is great for a sparse graph.

Let “high degree node” be a node with degree $> \sqrt{m}$.

This algorithm can be used to list all triangles.

$$m = O(n)$$

$O(m^{3/2})$
\[
m = \frac{\sqrt{n}}{2} + n + \sqrt{n} = O(n)
\]
so \( T = \Omega\left(\frac{\sqrt{n}}{3}\right) \)

5. **Edgeless Format**

\((u, v) \in E\) is the input to mappers

6. **Map Reduce for Computing Neighborhoods**

\[
\text{map}((u, v)) \quad \text{emit}(u, v) \quad \text{emit}(v, u)
\]

\[
\text{reduce}(v, \Gamma(v)) \quad \text{for } (u, w) \in \Gamma(v) \quad \text{output}((u, w) \rightarrow v)
\]

Use ordering:

\[
\text{map}((u, v)) \quad \text{if } \deg(u) \leq \deg(v) : \quad \text{emit}(u, v) \quad \text{else:} \quad \text{emit}(v, u)
\]

\[
\text{reduce}(v, \Gamma^*(v)) \quad \text{for } (u, w) \in \Gamma^*(v) \quad \text{output}((u, w) \rightarrow v)
\]

Now, number of operations in reduce is \( O(\sqrt{m}) \).

If node \( v \) is of low degree:

the reduce key complexity is at most \( \left(\frac{\sqrt{m}}{2}\right) \rightarrow O(\sqrt{m}) \).

Else if \( v \) is high degree:

Reduce key complexity is \( \left(\frac{\sqrt{m}}{2}\right) \rightarrow O(\sqrt{m}) \).

And, shuffle size is number of edges \( \rightarrow O(m) \). Output of MR gives two hop paths.

7. **But, what if the graph is not sparse?**

Let \( A_{ij} \) be an adjacency matrix.

\[
A_{ij}^3 = \text{number of paths of length 3 between.}
\]
We can do matrix multiplication $O(n^\gamma)$ where $\gamma = 2.374$.

$A_{ij}^3/6$ counts number of triangles.

Before, we had algorithm $O(m^{1.5})$ and now it’s $O(m^{1.4})$. See also Alon et al. 1997.

**8. Next class**

Compute cosine similarities

Generalize to squaring a matrix

This Friday is Spark Workshop