UNIVERSITY OF ILLINOIS AT CHICAGO, INFORMATION AND DECISION SCIENCES

MapReduce-based PageRank Algorithm

1 ALGORITHM DESCRIPTION

In this assignment, you will implement a MapReduce-based commonly-used web link analysis algorithm: PageRank.
The input: Graph G and parameter β. The Graph G is directed without having dead end nodes. You can either write codes to generate such a graph or manually create it. Common values for β are in the range of 0.8 to 0.9. In the case of spider traps, the random surfer follows a link at random with probability β.

The output: PageRank vector $\vec{r}$, each component would be the ranking value of a node in the graph G.

The algorithm: The stopping criteria is that the change of PageRank vectors between current iteration and the previous iteration is less than a very small value ϵ (for example, $\epsilon = 0.05$. The sequential version of PageRank implementation (pseudo code) is shown below.

2 PSEUDO ALGORITHM FOR SEQUENTIAL VERSION

PageRank algorithm on a directed graph G
begin
$set: r_j^{(0)} = \frac{1}{N}, t = 1$

N is the number of nodes in the graph G

do
(1) $\forall j : r_j^{'(t)} = \sum_{i \to j} \beta \frac{r_i^{(t-1)}}{d_i}$

$r_j^{'(t)} = 0$ if in-degree of j is 0

(2) Now re-insert the leaked PageRank:

$\forall j : r_j^{(t)} = r_j^{'(t)} + \frac{1-S}{N}$ where: $S = \sum_j r_j^{'(t)}$

(3) $t = t+1$ od

while $\sum_j \mid r_j^{(t)} - r_j^{(t-1)} \mid > \epsilon$ od

end

3 Pseudo Algorithm for MapReduce Version

/ ⋆ The Mapper is to invert the input ⋆ /

$Mapper$:

$\forall page_j \in (page_1, page_2, \cdots, page_k)$

output $page_j \rightarrow \langle page_i, \frac{rank_i}{d_i} \rangle$ // d_i is degree of node i.

output $page_i \rightarrow page_1, page_2, \cdots, page_k$

/ ⋆ The Reducer is to update the ranking using the in-links ⋆ /

$Reducer$:

Input is in a format of △. The key: $page_k$

$\forall$ in-link $page_i \in (page_1, page_2, \cdots, page_n)$

$rank_k += \frac{rank_i}{d_i} * \beta$

$output \langle page_k, rank_k \rangle \rightarrow \langle page_1, page_2, \cdots, page_n \rangle$

// $page_1$, $page_2$, $\cdots$, $page_n$ are out-links of $page_k$.

After map function, we have temporary files in the following structure (△):

$page_k \rightarrow \langle page_1, rank_1 \rangle$,

$\langle page_2, rank_2 \rangle$,

.....,

$\langle page_n, rank_n \rangle$,

$\langle page_{k1}, page_{k2}, \cdots, page_{kn} \rangle$

where $page_1$, $page_2$, $\cdots$, $page_n$ are the in-links of $page_k$,

and $page_{k1}$, $page_{k2}$, $\cdots$, $page_{kn}$ are the out-links.

4 Submission Instruction

You need to submit one zip file, which contain all source codes (*.java files) and README file. The name of your zip file would be following the format like: Firstname_Lastname_Assignment_6.zip. Please follow this format and submit it through the blackboard. Please try to comment your codes for critical sections and make your codes as readable as possible.