HPdgraph – Distributed algorithms for graph analytics

User Guide

Contents

HPdgraph – Distributed algorithms for graph analytics

User Guide

1. Introduction

The HPdgraph package provides a few distributed algorithms for graph analytics. It is written based on the infrastructure created in HP-Labs for distributed computing in R. The main functions of the this package are:

- hpdpagerank: It is a distributed function for computing pagerank vector of a graph.
- hpdwhich.max: It finds and returns the index of the maximum value stored in a darray.

2. hpdpagerank

hpdpagerank function is a distributed implementation of pagerank algorithm for directed graphs. Power Method is used to calculate the pagerank vector of a graph. Self-loops are allowed, but multiple edges are not considered. The input graph should be in the adjacency matrix format and stored in a darray. Moreover, it must be partitioned column-wise. It should be noticed that values of the non-zero elements of the darray representing graph will be altered after running hpdpagerank function. Nevertheless, it would be easy to retrieve the initial graph as it will be shown here through an example.

More detail about the interface of the function and its input arguments can be found in the manual of the package.

Example

As an example, let's use the real-world graph of Live-Journal social network which can be downloaded from http://snap.stanford.edu/data/soc-LiveJournal1.html. This graph has 4,847,571 vertices and 68,993,773 edges. The downloaded file should be uncompressed to its text file (using gunzip) before it can be used in our example. Let's assume that the text file is stored in this path "/graph/soc-LiveJournal1.txt" and we want to run hpdpagerank on two node with 12 cores each. Therefore, in this example we need to split the input graph to 24 parts. The procedure is displayed in Figure 1.

```
> library(HPdata)

> distributed_start(cluster="cluster2.xml")

> ret <- splitGraphFile(inputFile="/graph/soc-LiveJournal1.txt", outputPath="/graph/temp",
nSplits=24)
Sending files to node1
soc-LiveJournal1.txt0
soc-LiveJournal1.txt1
…
Sending files to node2
soc-LiveJournal1.txt0
soc-LiveJournal1.txt1
…

> dg <- file2graph(ret$pathPrefix, ret$nVertices, ret$verticesInSplit, ret$isWeighted)
Openning files:
/graph/temp/soc-LiveJournal1.txt0
... until ...
/graph/temp/soc-LiveJournal1.txt23
Progress: 100%

> dim(dg$X)
[1] 4847571 4847571

> sum(dg$X)
[1] 68993773

> library(HPdgraph)

> pr <- hpdpagerank(dg$X)

> dim(pr)
[1] 1 4847571

> hpdwhich.max(pr)
[1] 214407

> sum(dg$X) # the graph is altered
[1] 3662184

> # retrieving the initial graph

> foreach(I, 1:npartitions(dg$X), function(x=splits(dg$X,i)) {
+ x <- abs(sign(x))
+ update(x)
+ })
Progress: 100%
[1] TRUE

> sum(dg$X)
[1] 68993773
```

Figure 1. An example for running hpdpagerank for Live-Journal social network

3. hpdwhich.max

hpdwhich.max function is a distributed version of which.max function for a 1D-array which has darray as its input argument. As it is shown in Figure 1, it is a useful function for finding the vertex with highest ranke in the pagerank vector returned by hpdpagerank function.