This post was published to Yifan&#039;s Blog at 4:00:27 PM 3/1/2016

HITS Algorithm in R

This is a rewrite of HITS ( <https://en.wikipedia.org/wiki/HITS_algorithm> ) in R. Indeed, the adjacent matrix *adjmat* is often considered as a sparse matrix, which could be replaced with **Matrix**::sparseMatrix instead.

Length = 100

## definition

## adjmat[i,j] i->j

adjmat = matrix(sample(0:1,Length\*Length,replace = T, prob = c(.9,.1)),Length,Length)

iter\_times = 100

tol = 1e-12 # tollerance

# clean: row col == all zero

diag(adjmat) = 1

# initial

p.auth = rep(1,Length) # p.auth is the authority score of the page p

p.hub = rep(1,Length) # p.hub is the hub score of the page p

#' incomingNeighbors is the set of pages that link to p

#' @param vet is a index (not vector)

#' @param adh is the adjacent matrix

incomingNeighbors <- function(vet, adjmat){

which(adjmat[,vet]>0)

}

#' outcomingNeighbors is the set of pages that link to p

#' @param vet is a index (not vector)

#' @param adh is the adjacent matrix

outcomingNeighbors <- function(vet, adjmat){

which(adjmat[,vet]>0)

}

# start for loop: run the algorithm for k steps

for( k in 1:iter\_times){ ## TODO: update all authority values first

p.hubold = p.hub

norm\_factor = 0;

for (p in 1:Length){

p.auth[p] = sum(p.hub[incomingNeighbors(p,adjmat)]); # use inner for loop

norm\_factor = norm\_factor + (p.auth[p])^2; # calculate the sum of the squared auth values to normalise

}

p.auth = p.auth/sqrt(norm\_factor) # update the auth scores

norm\_factor=0; # reset normal factor

for (p in 1:Length){ # then update all hub values

p.hub[p] = sum(p.auth[outcomingNeighbors(p,adjmat)]);

norm\_factor = norm\_factor + (p.hub[p])^2;

}

p.hub = p.hub/sqrt(norm\_factor) # normalise the hub values

err = sqrt(sum( (p.hub-p.hubold)^2))

cat(k,"-change:",err,'\n')

if (err < tol) break;

}

plot(x=1:Length,p.auth,type='h',col=2)