stichting
mathematisch
centrum

AFDELING MATHEMATISCHE BESLISKUNDE BN 9/71 OCTOBER

JAC.M. ANTHONISSE

# THE RUSH IN A DIRECTED GRAPH



2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

## 1. Introduction

In several applications of graph theory it is of interest to compare the 'importance' of the vertices of a graph. Evidently, the definition of 'importance' should depend upon the specific application.
The centrality of a vertex, i.e. the sum of the distances from a vertex to the other vertices, is a well-known example.
In this note the concept of 'rush' is introduced, applicable to the vertices as well as to the arcs of a directed graph.
The rush in an element is the total flow through that element, resulting from a flow between each pair of vertices.

## 2. Definitions

Only finite directed graphs without loops are consired, undirected graphs can be included by interpreting them as symmetric digraphs.

A path from vertex u to vertex z is a sequence

$$u,\ (u,v),\ v,\ (v,w),\ w,\ \ldots,\ y,\ (y,z),\ z$$

where

u, v, w, ... denote vertices

and

(u,v), (v,w), ... denote arcs, directed from u to v, from v to w, etc.

If a path from u to z exists then z is a descendant of u, whereas u is an ascendant of z.

The length of a path is defined as the number of arcs constituting the path. If a path from u to z exists there also exist one or more minpaths, i.e. paths of minimal length. The length of a minpath from u to z is the distance from u to z.

From the matrix $(a_{ij})$ associated with a graph, i.e.

$$a_{ij} = \begin{cases} 1 & \text{if an arc from } x_i \text{ to } x_j \text{ exists,} \\ 0 & \text{otherwise,} \end{cases}$$

the matrix $(d_{ij})$ of distances, i.e.

$$d_{ii} = 0$$

$$d_{ij} = \left.\begin{cases} \text{distance from } x_i \text{ to } x_j\text{, if } x_j \text{ is a descendant of } x_i\text{,} \\ \infty \quad \text{otherwise,} \end{cases}\right\} \qquad i \neq j$$

is easily found using Floyds algorithm [1].

It is not difficult to construct the minpaths from $(d_{ij})$.

To define the rush, one unit of flow is sent from each vertex $x_i$ to each vertex $x_j$, provided $0 < d_{ij} < \infty$. The flow is sent along minpaths only, if $e_{ij}$ minpaths from $x_i$ to $x_j$ exist then $1/e_{ij}$ units are sent along each path.

The rush in an element (vertex or arc) is defined as the total flow through that element, resulting from the flows defined above.

It should be noted that the flow originating from $x_i$ and the flow with destination $x_i$ do not belong to the rush in $x_i$.

The definition of rush can be extended in at least two directions. Instead of one unit, $f_{ij}$ units of flow can be sent from $x_i$ to $x_j$. Instead of length one, length $l_{ij}$ can be assigned to arc $(x_i, x_j)$. The length of a path is then defined at the total length (sum) of its constituent arcs.

As an example, consider the graph depicted in figure 1.

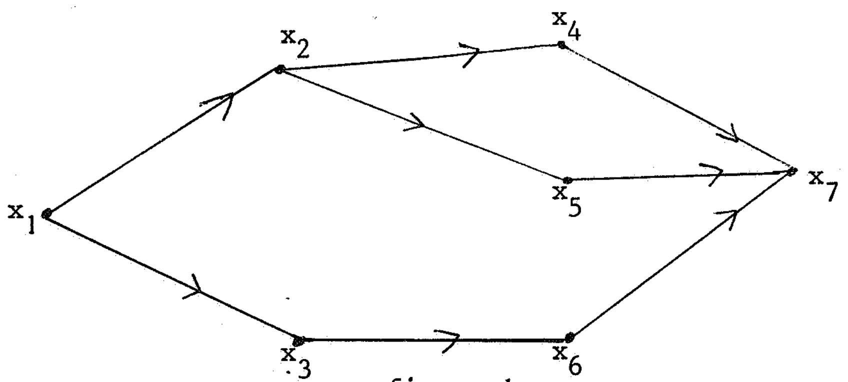


figure 1

Table 1 gives the minpaths with length > 1, only the constituent vertices are listed.

| from | to | minpaths |
|---|---|---|
| 1 | 4 | 1, 2, 4 |
| 1 | 5 | 1, 2, 5 |
| 1 | 6 | 1, 3, 6 |
| 1 | 7 | 1, 2, 4, 7 ; 1, 2, 5, 7 ; 1, 3, 6, 7 |
| 2 | 7 | 2, 4, 7 ; 2, 5, 7 |
| 3 | 7 | 3, 6, 7 |

table 1

Table 2 contains the rush in each element of the graph.

| vertex | rush | arc | rush |
|---|---|---|---|
| 1 | 0 | 1 2 | $3\frac{2}{3}$ |
| 2 | $2\frac{2}{3}$ | 1 3 | $2\frac{1}{3}$ |
| 3 | $1\frac{1}{3}$ | 2 4 | $2\frac{5}{6}$ |
| 4 | $\frac{5}{6}$ | 2 5 | $2\frac{5}{6}$ |
| 5 | $\frac{5}{6}$ | 3 6 | $3\frac{1}{3}$ |
| 6 | $1\frac{1}{3}$ | 4 7 | $1\frac{5}{6}$ |
| 7 | 0 | 5 7 | $1\frac{5}{6}$ |
| | | 6 7 | $2\frac{1}{3}$ |

<u>table 2</u>

## 3. Computation

Two ALGOL-60 procedures are presented.
The procedure minpaths (d, e, n) calculates, from the n x n matrix d of distances, the n x n matrix e, where $e_{ij}$ = the number of minpaths from $x_i$ to $x_j$.
The procedure rush (d, e, r, n) calculates from the matrices d and e, the rush-matrix r, where $r_{ii}$ = the rush in $x_i$ and, for $i \neq j$, $r_{ij}$ = the rush in arc $(x_i, x_j)$ if this arc exists, zero otherwise.
Both procedures are straightforward, and no claim of efficiency is made.

The procedures are based upon the relation

$$e_{ij} = \begin{cases} 1 & \text{if} \quad d_{ij} = 1 \\ \sum\limits_h (e_{hj} \mid d_{ih} = 1,\ d_{ij} = d_{ih}+d_{hj}) & \text{if } 1 < d_{ij} < \infty. \end{cases}$$

```
procedure minpaths(d,e,n); value n; integer n;
integer array d,e;
begin   integer i,j;
        integer procedure paths(i,j); value i,j; integer i,j;
        begin
                if e[i,j] = -1 then
                begin   integer a,h;
                        a:= d[i,j] - 1; e[i,j]:= 0;
                        if a = 0 then e[i,j]:= 1 else
                        for h:= 1 step 1 until n do
                        if d[i,h] = 1 ∧ d[h,j] = a then
                        e[i,j]:= e[i,j] + paths(h,j)
                end;
                paths:= e[i,j]
        end paths;
        for i:=1 step 1 until n do
        for j:=1 step 1 until n do e[i,j]:=-1;
        for i:=1 step 1 until n do
        for j:=1 step 1 until n do
        if d[i,j] = 0 ∨ d[i,j] ≥ n then e[i,j]:= 0 else
        if e[i,j] = -1 then paths(i,j)
end minpaths;
```

```
procedure rush(d,e,r,n); value n; integer n;
integer array d,e; real array r;
begin   integer i,j;
        procedure add(b,i,j); value b,i,j;
        integer i,j; real b;
        begin   integer a,h,p;
                real c;
                a:= d[i,j] - 1; p:= e[i,j];
                if a = 0 then r[i,j]:= r[i,j] + b else
                for h:= 1 step 1 until n do
                if d[i,h] = 1 ∧ d[h,j] = a then
                begin
                        c:=b × e[h,j] / p;
                        r[i,h]:= r[i,h] + c;
                        r[h,h]:= r[h,h] + c;
                        add(c,h,j)
                end
        end add;

        for i:= 1 step 1 until n do
        for j:= 1 step 1 until n do r[i,j]:= 0;
        for i:= 1 step 1 until n do
        for j:= 1 step 1 until n do
        if d[i,j] > 0 ∧ d[i,j] < n
        then add(1,i,j)
end rush;
```

## 4. An Application

The concept of rush has been applied in a study of relations between committees. Table 3 describes the constitution of 9 committees.

| person \ committee | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| a | 1 | 1 | 1 | | | | | | |
| b | | | 1 | 1 | | | | | |
| c | | | | 1 | 1 | | | | |
| d | | | | 1 | | 1 | | | |
| e | | | | | | 1 | 1 | | |
| f | | | | | | | 1 | 1 | |
| g | | | | | | | 1 | | 1 |

table 3

In figure 2 a graph is depicted, which describes the relations between the committees.
Each vertex corresponds to a committee, two vertices are connected if the committees have a member in common.

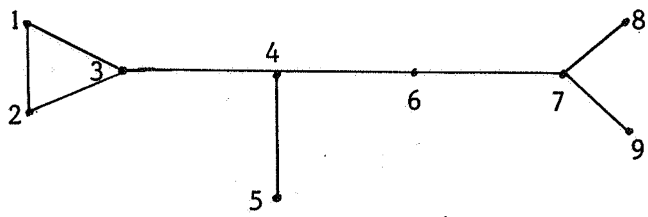


figure 2

It is very easy to compute the rush in this graph, table 4 contains the rush in each element.

| vertex | rush | edge | rush | | person |
|---|---|---|---|---|---|
| 1 | 0 | 1,2 | 2 | | |
| 2 | 0 | 1,3 | 14 | 30 | a |
| 3 | 24 | 2,3 | 14 | | |
| 4 | 38 | 3,4 | 36 | | b |
| 5 | 0 | 4,5 | 16 | | c |
| 6 | 30 | 4,6 | 40 | | d |
| 7 | 22 | 6,7 | 36 | | e |
| 8 | 0 | 7,8 | 14 | | f |
| 9 | 0 | 7,9 | 14 | | g |

table 4

From table 4 it might be concluded that committee 4 is very important for the exchange of information between the committees, and that person d has an important position.

## 5. Reference

R.W. Floyd,
Algorithm 97
Comm. ACM 5(1962) 345.