BİL465 BİLGİSAYAR AĞ YÖNETİMİ LABORATUVARI DERSİ
ARASINAV-II PROJESİ

Projenin veriliş tarihi: 10.12.2019-12:00
Projenin son teslim tarihi: 25.12.2019-17:00
Proje sunumuna gelirken uygulamanızın çalışan halini göstermeniz yeterli olacaktır.

The Classical DFS Algorithm

Our first distributed DFS algorithm called *Classic_DFS*. Tarry's algorithm had two rules (Rule 1 and Rule 2): a process never forwards the token twice through the same channel, and a noninitiator forwards the token to its parent if it cannot forward it to any other neighbor with the first rule. This algorithm is formed by the addition of the following rule:

- Rule 3: When a process receives the token, it sends it back through the same channel if this is allowed by Rules 1 and 2.

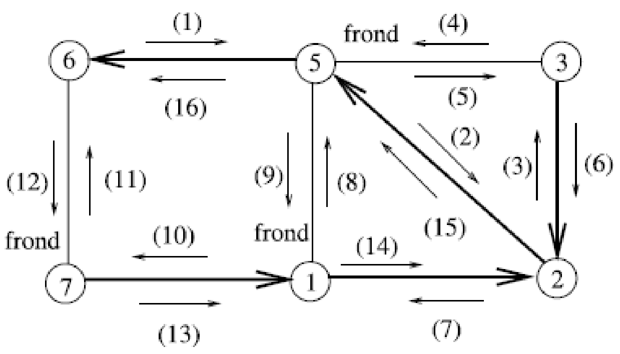
Rule 1: A process never forwards the token twice through the same channel.
Rule 2: A noninitiator forwards the token to its parent, the node from which it received the token for the first time, only if there is no other channel left according to Rule 1.

In order to implement Rule 1, node *i* uses an array *used* to monitor status of its neighbors. Upon reception of the token, node *i* forwards the token to an unsearched neighbor *j*, assigns true value to *used*[*j*], and when all of the neighbors have true values in the *used* array meaning that all neighbors have been searched, it forwards the token back to its parent to implement Rule 2.

Algorithm 2 displays the operation of the Classical DFS Algorithm. Lines 23–24 implement R3 so that the token received via the frond edge is sent back to the sender.

Figure 2 shows the operation of *Classic_DFS* in a sample network with six nodes, where the traversal of the token is displayed by the time frame it occurs along the edges. Node 6 starts the algorithm, and after reaching node 2 in time 2, the token is forwarded to node 3 in time 3, which forwards the token to its neighbor 5. Node 5, however, applies Rule 3, finds edge {5,3} that is a frond edge, and returns token immediately back to 3. Similarly, token is returned to senders along the frond edges {1,5} and {6,7}. It can also be seen that the construction of the whole DFS tree is completed in 16 steps, which is twice the number of edges for this graph, as there are two traversals for each edge.

Fig. 2. The classical DFS algorithm execution example



Algorithm 2. *Classic_DFS*

```
int $parent \leftarrow \perp$
boolean $visited[d_i] \leftarrow \{false\}$
message types $token$

if $i = root$ then                                        ▷ root starts the algorithm
    $parent \leftarrow i$, choose $j \in \Gamma(i)$
    $visited[j] \leftarrow true$, send $token(i)$ to $j$
end if

while $true$ do
    receive $token(j)$
    if $parent = \perp$ then                                      ▷ token first time
        $parent \leftarrow j$
    end if
    if $\forall q \in \{\Gamma(i) \setminus \{parent\}\}: visited[q]$ then
        if $i = root$ then                        ▷ if root and all searched, terminate
            exit
        elsesend $token$ to $parent$
            $visited$[parent] $\leftarrow true$
            exit                                  ▷ all nodes except root terminate
        end if
    else
        if $j \neq parent \wedge \neg visited[j]$ then ▷ check to send token back from same channel
            $q \leftarrow j$
        else
            choose $q \in \{\Gamma(i) \setminus \{parent\}\} : \neg visited[q]$      ▷ send token to unsearched
                                                                            neighbor
        end if
        $visited[q] \leftarrow true$
        send $token$ to $q$
    end if
end while
```