WS 2007/2008

Fundamental Algorithms

Dmytro Chibisov, Jens Ernst

Fakultät für Informatik
TU München

http://www14.in.tum.de/lehre/2007WS/fa-cse/

Fall Semester 2007

1. Depth First Search

1.1 Application of DFS: Topological Sorting

Definition 1

Given a directed acyclic graph (*dag*) $G = (V, E)$, a topological sort of G is a linear ordering of all its vertices such that if G contains an edge (u, v), then u appears before v in the ordering.

Computation problem: assign the unique number $f(v) \in \{1, \ldots, |V|\}$ to every $v \in V$, such that for every $(u, v) \in E$ $f(u) < f(v)$.

Example 2

$$V = \{shirt, belt, tie, jacket, watch, pants, underwear, shoes, socks\}$$

$$E = \{(shirt, tie), (shirt, belt), (tie, jacket), (belt, jacket),$$
$$(pants, shoes), (pants, belt), (socks, shoes), (underwear, pants)\}$$

Topological Sorting:

void TopSort(vertex v){
  initialize the empty stack; // global variable
  foreach ($v \in V$) do $v.dfsnum := 0$; od
  while $\exists v_0 \in V : v_0.dfsnum = 0$ do modified-DFS(v_0) od
  od }

Modified DFS:

void modified-DFS(vertex v){
  $v.dfsnum$:= counter++;
  foreach ($w|(v,w) \in E$) do
    if ($w.dfsnum$=0) then modified-DFS(w); fi
  od
  push(v) }

Topological Sorting:

```
void TopSort(vertex v){
  initialize the empty stack; // global variable
  foreach (v ∈ V ) do v.dfsnum := 0; od
  while ∃v0 ∈ V : v0.dfsnum = 0 do modified-DFS(v0) od
  od }
```

Modified DFS:

```
void modified-DFS(vertex v){
  v.dfsnum:= counter++;
  foreach (w|(v, w) ∈ E) do
    if (w.dfsnum=0) then modified-DFS(w); fi
  od
  push(v) }
```

1.2 Classification of edges:

DFS performs the partition of edges into four classes:

- **Tree edges** – edge (u, v) is a tree edge if v was first discovered by exploring edge (u, v) $(v.dfsnum = 0)$.
- **Back edges** – edge (u, v) connecting a vertex u to an ancestor v in a depth-first tree $(v.dfsnum < u.dfsnum$, and DFS(v) is not finished).
- **Forward edges** – non-tree edges (u, v) connecting a vertex u to a descendant v in a depth-first tree $(v.dfsnum > u.dfsnum)$.
- **Cross edges** – are all other edges $(u.dfsnum > v.dfsnum$, and DFS(v) is finished).

Lemma 3

In a depth first search of an undirected graph G, every edge of G is either a tree edge, or a back edge.

Proof.

Let $\{u, v\}$ be an arbitrary edge of G, and suppose without loss of generality that $u.dfsnum < v.dfsnum$. Then, v must be finished before we finish u, since v is on u's adjacency list. If the edge $\{u, v\}$ is explored first in the direction from u to v, then $\{u, v\}$ becomes a tree edge. If $\{u, v\}$ is explored first in the direction from v to u, then $\{u, v\}$ is a back edge. □