Netzwerkalgorithmen

April 11, 2016

1 Zusätzliches blabla

Makros in C/C++: #define alias replace, wobei replace auch Code sein kann.

2 Datentypen für Graphen und Netzwerke (LEDA)

Definition eines Datentyps

Definition der Objekte des Typs: $stack < T >$
Konstruktion: $stack < int >$ S(100) (max Größe)
Operationen: $s.push(Tx)$, $Ts.pop()$
Bemerkung zu Implementierung

Graph-Datentyp in LEDA

Der Typ graph repräsentiert gerichtete Graphen.
Ein Graph g besteht aus zwei Typen von Objekten: node und edge
Mit jedem Knoten v sind zwei Listen von Kanten ($list < edge >$) verbunden (eingehend und ausgehend)
Mit jeder Kante e werden 2 Knoten source und target gespeichert.

Operationen auf G

Update:
node G.new_node(), erzeugt einen neuen Knoten in G und gibt ihn zurück. edge G.new_edge(node v, node w)
void G.del_edge(edge)
Access:
$list < edge >$ G.out_edges(node v);
int G.outdeg(node v);
node G.source(edge);
node G.target(edge);
Iteration:
forall_nodes(v,G)
forall_edges(e,G)
forall_out_edges(e,v)
forall_in_edges(e,v)

1. Problem

Gegeben: Graph G=(V,E)
Frage: Ist G azyklisch?

Algorithmus siehe Topologisches Sortieren: Entferne jeweils einen Knoten v mit indeg(v)=0 bis der Graph leer ist. Falls wir keinen solchen Knoten finden dann ist der Graph zyklisch, falls G am Ende leer, ist er azyklisch.
C++:

```
bool ACYCLIC(graph G){            //Call by value damit G nicht zerstört
        list<node> zero;
        node v;
        forall_nodes(v,G){
                if (G.indeg(v)==0) zero.append(v);
        }
        while (!zero.empty()){
                node u = zero.pop();
                edge e;
                forall_out_edges(e,u){
                        node w=G.target(e);
                        G.del_edge(e);
                        if (G.indeg(w) == 0){
                                zero.append(w);
                        }
                }
        }
        return G.empty();
}
```