Topological sort

Goldsmiths Computing

January 13, 2019

Motivation

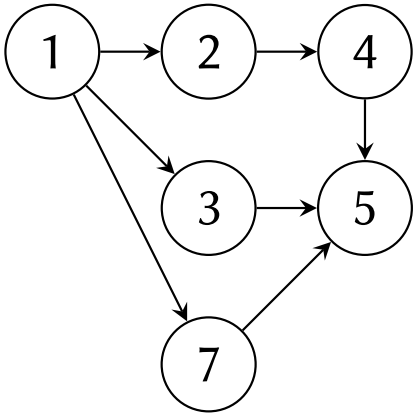
Given dependency information, generate a set of tasks in order so that dependent tasks are done after dependencies:

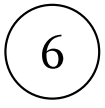
- spreadsheet recalculation
- Makefile target building
- database foreign key loading order
- serialization of data

Definition

A topological sort of a directed graph yields a linear collection of vertices such that if u and v are vertices and there is a edge from u to v, then u precedes v in the ordering.

Example





- 1, 2, 3, 7, 4, 5, 6
- 1, 6, 7, 2, 4, 3, 5
- 6, 1, 7, 3, 2, 4, 5

Kahn's topological sort

```
function KahnTS(G)
    L ← new DynamicArray(); S ← new Collection()
    for v ∈ vertices(G) ∧ ∄ e ∈ edges(G) : to(e) = v do
        insert(S,v)                          ▷ S: set of vertices with no incoming edges
    end for
    while ¬empty?(S) do
        v ← select!(S); push(L,v)                          ▷ add v to the end of L
        for e ∈ edges(G) ∧ from(e) = v do
            z ← to(e)
            remove-edge!(G,e)
            if ∄ f ∈ edges(G) : to(f) = z then
                insert(S,z)
            end if
        end for
    end while
    return L                              ▷ if G still has edges, then G was not a DAG
end function
```

Depth-first topological sort

```
function DFTS(G)
    L ← new List()
    UM ← new Set(VERTICES(G))
    TM ← new Set(); PM ← new Set()
    function VISIT(v)
        if v ∈ PM then
            return
        end if                                  ▷ if v ∈ TM then we have found a cycle
        DELETE!(UM,v); INSERT(TM,v)
        for e ∈ EDGES(G) ∧ FROM(e) = v do
            VISIT(TO(e))
        end for
        DELETE!(TM,v); INSERT(PM,v)
        L ← CONS(v,L)
    end function
    while ∃ v ∈ UM do
        v ← SELECT!(UM)
        VISIT(v)
    end while
    return L
end function
```

Relation to relations

Consider a relation R such that R is irreflexive, antisymmetric and transitive (a strict partial order). A topological sort of the graph induced by that relation will convert the partial order into a total order.
The transitive closure of any directed acyclic graph corresponds to a strict partial order.

Work

1. Reading
   - CLRS, sections 22.3, 22.4
   - DPV, section 3.3
2. Exercises and problems
   - CLRS, exercises 22.3-2, 22.4-1, 22.4-5
   - DPV, exercises 3.3, 3.14