Introduction to Programming – Lecture 21

Topological Sort
Part 1: Algorithm

Last revised 24 January 2006

1

The problem

From a given partial order, produce a compatible total order

Partial order: ordering constraints between elements of a set, e.g.
- "Dishes dishes before Politicising politics"
- "Walk to Uetliberg before lunch"
- "Take your medicine before lunch"
- "Finish lunch before removing dishes"

Total order: sequence including all elements of set

Compatible: the sequence respects all ordering constraints
- Uetliberg, Medicine, Lunch, Dishes, Politics: OK
- Medicine, Uetliberg, Lunch, Dishes, Politics: OK
- Politics, Medicine, Lunch, Dishes, Uetliberg: not OK

2

"Topological sort"

From a given partial order, produce a compatible total order

3

Why we are doing this!

- Very common problem in many different areas
- Interesting, efficient, non-trivial algorithm
- Illustration of many algorithmic techniques
- Illustration of software engineering techniques: from algorithm to component with useful API
- Opportunity to learn or rediscover important mathematical concepts: binary relations (order relations in particular) and their properties
- It’s just beautiful!

Today: problem and math basis

Next time: detailed algorithm and component

4
The problem

From a given partial order, produce a compatible total order

Partial order: ordering constraints between elements of a set, e.g.

- "Dishes before Politics"
- "Walk to Uetliberg before lunch"
- "Take your medicine before lunch"
- "Finish lunch before removing dishes"

Compatible: the sequence respects all ordering constraints

- Uetliberg, Medicine, Lunch, Dishes, Politics: OK
- Medicine, Uetliberg, Lunch, Dishes, Politics: OK
- Politics, Medicine, Lunch, Dishes, Uetliberg: not OK

Pictured as a graph

Rectangles with overlap constraints

Constraints: [B, A], [D, A], [A, C], [B, D], [D, C]

Possible solution: B D E A C

Rectangles with overlap constraints

Constraints: [B, A], [D, A], [A, C], [B, D], [D, C]

Possible solution: B D E A C

Rectangles with overlap constraints

Constraints: [B, A], [D, A], [A, C], [B, D], [D, C]

Possible solution: B D E A C

Rectangles with overlap constraints

Constraints: [B, A], [D, A], [A, C], [B, D], [D, C]

Possible solution: B D E A C
Rectangles with overlap constraints

Constraints: \([B, A], [D, A], [A, C], [B, D], [D, C]\)

Possible solution: \(B\ D\ E\ A\ C\)

Sometimes there is no solution

- “Introducing recursion requires that students know about stacks”
- “You must Politics abstract data types before introducing stacks”
- “Abstract data types rely on recursion”

The constraints introduce a cycle

Topological sort: example uses

From a dictionary, produce a list of definitions such that no word occurs prior to its definition

Produce a complete schedule for carrying out a set of tasks, subject to ordering constraints  
(This is done for scheduling maintenance tasks for industrial tasks, often with thousands of constraints)

Produce a version of a class with the features reordered so that no feature call appears before the feature’s declaration

Overall structure (1)

Given:
- A type \(G\)
- A set of elements of type \(G\)
- A set of constraints between these elements

Required:
- An enumeration of the elements, in an order compatible with the constraints

Required:
- An enumeration of the elements, in an order compatible with the constraints

Class ORDERABLE[\(G\)]

- feature:
  - elements: LIST[\(G\)]
  - constraints: LIST[TUPLE[\(G, G\)]]
  - topsort: LIST[\(G\)] is
  - ensure:
    - compatible(Result, constraints)
  - end
Introduction to Programming – Lecture 21

Chair of Software Engineering

Binary relation on a set

Any property that either holds or doesn’t hold between two elements of a set.

On a set PERSON of persons, example relations are:
- mother: a mother b holds if and only if a is the mother of b.
- father
- child
- sister
- sibling (brother or sister)

Notation: a r b to express that r holds of a and b.

Example: the before relation

The set of interest:
- Tasks = {Politics, Lunch, Medicine, Dishes, Üetliberg}

The constraining relation:
- Dishes before Politics
- Üetliberg before Lunch
- Medicine before Lunch
- Lunch before Dishes

Some special relations on a set X

universal [X]: holds between any two elements of X
id [X]: holds between every element of X and itself
empty [X]: holds between no elements of X

Relations: a more precise mathematical view

We consider a relation r on a set P as:

A set of pairs in P x P, containing all the pairs [x, y] such that x r y.

Then x r y simply means that [x, y] ∈ r.

Example: the before relation

The set of interest:
- elements = {Politics, Lunch, Medicine, Dishes, Üetliberg}

The constraining relation:
- before =
  - {Dishes, Politics}, {Üetliberg, Lunch}, {Medicine, Lunch}, {Lunch, Dishes}

Using ordinary set operators

- spouse = wife ⋃ husband
- sibling = sister ⋃ brother ⋃ id [Person]
- father ⋑ ancestor

universal [X] = X x X (cartesian product)
empty [X] = ∅
Possible properties of a relation

(On a set $X$ All definitions must hold for every $a, b, c \in X$)

- **Total**: $(a \neq b) \Rightarrow ((a \leq b) \vee (b \leq a))$
- Reflexive: $a \leq a$
- Irreflexive: not $(a \leq a)$
- Symmetric: $a \leq b \Rightarrow b \leq a$
- Antisymmetric: $(a \leq b) \land (b \leq a) \Rightarrow a = b$
- Asymmetric: not $(a \leq b) \land (b \leq a)$
- Transitive: $(a \leq b) \land (b \leq c) \Rightarrow a \leq c$

Definition of "total" is specific to the discussion (there is no standard definition). The other terms are standard.

**Examples (on a set of persons)**

- **Father**
  - Reflexive, symmetric, transitive
- **Mother**
  - Symmetric, reflexive
- **Uncle**
  - Reflexive, antisymmetric
- **Grandparent**
  - Asymmetric, irreflexive

**Total order relation (strict)**

Relation is strict total order if:
- Total
- Irreflexive
- Transitive

Example: "less than" $<$ on integers (or reals)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0$</td>
<td>$1$</td>
<td>$2$</td>
<td>$3$</td>
<td>$4$</td>
<td>$5$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$0 &lt; 1$</td>
<td>$0 &lt; 2$</td>
<td>$0 &lt; 3$</td>
<td>$0 &lt; 4$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1 &lt; 2$</td>
<td>$1 &lt; 3$</td>
<td>$1 &lt; 4$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2 &lt; 3$</td>
<td>$2 &lt; 4$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Theorem**

A strict (total) order is **asymmetric**

**Total order relation (non-strict)**

Relation is non-strict total order if:
- Total
- Reflexive
- Transitive

Example: "less than or equal" $\leq$ on integers

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0 \leq 0$</td>
<td>$0 \leq 1$</td>
<td>$0 \leq 2$</td>
<td>$0 \leq 3$</td>
<td>$0 \leq 4$</td>
<td>$\ldots$</td>
<td></td>
</tr>
<tr>
<td>$1 \leq 1$</td>
<td>$1 \leq 2$</td>
<td>$1 \leq 3$</td>
<td>$1 \leq 4$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2 \leq 2$</td>
<td>$2 \leq 3$</td>
<td>$2 \leq 4$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reflexive: $a \leq a$
- Symmetric: $a \leq b \Rightarrow b \leq a$
- Antisymmetric: not $(a \leq b) \land (b \leq a)$
- Asymmetric: not $(a \leq b) \land (b \leq a)$
- Transitive: $(a \leq b) \land (b \leq c) \Rightarrow a \leq c$
Total order relation (strict)

Relation is strict total order if:
- Total: \((a \neq b) \Rightarrow ((a \leq b) \lor (b \leq a))\)
- Irreflexive: not \((a \leq a)\)
- Transitive

Partial order relation (strict)

Relation is strict partial order if:
- Irreflexive: not \((a \leq a)\)
- Symmetric: \(a \leq b \Rightarrow b \leq a\)
- Transitive: \((a \leq b) \land (b \leq c) \Rightarrow a \leq c\)

Example partial order

Here the following hold:
- No link between \(a\) and \(c\), \(b\) and \(c\), \(b\) and \(d\):
- e.g. neither \(a \leq c\) nor \(b \leq c\) nor \(b \leq d\)
- \(a \leq b\) \(c \leq d\)

Theorems

A strict (total) order is asymmetric

A total order is a partial order

("partial" order really means possibly partial)

Possible topological sorts

Here the relation \(\leq\) is:
- \(\{[a, b], [a, d], [c, d]\}\)

One of the solutions is:
- \(\{[a, b], [a, c], [a, d], [b, c], [b, d], [c, d]\}\)

We are looking for a total order relation \(\leq\) such that:
- \(a \leq f\)
Final statement of topological sort problem

From a given partial order, produce a compatible total order

where:

A partial order $p$ is compatible with a total order $t$ if and only if $p \subseteq t$

From constraints to partial orders

Is a relation defined by a set of constraints, such as

$\text{constraints} = \{[\text{Dishes, Politics}], [\text{Uetliberg, Lunch}], [\text{Medicine, Lunch}], [\text{Lunch, Dishes}]\}$

always a partial order?

Powers and transitive closure of a relation

$r^{i+1} = r^i \circ r$ where $\circ$ is composition

Transitive closure $r^* = r^1 \cup r^2 \cup ...$ always transitive

Reflexive transitive closure

$r^0 = \text{id}[X]$ where $X$ is the underlying set

$r^{i+1} = r^i \circ r$

Reflexive transitive closure: $r^* = r^0 \cup r^1 \cup r^2 \cup ...$ always reflexive and transitive

Acyclic relation

$r^* \cap \text{id}[X] = \emptyset$

Acyclic relations and partial orders

Theorems:

- Any (strict) order relation is acyclic.
- A relation is acyclic if and only if its transitive closure is a (strict) order.

(Also: if and only if its reflexive transitive closure is a nonstrict partial order)
The partial order of interest is **before**

\[
\text{before} = \{ \{ \text{Dishes, Politics} \}, \{ \text{Üetliberg, Lunch} \}, \\
\{ \text{Medicine, Lunch} \}, \{ \text{Lunch, Dishes} \} \}
\]

**Üetliberg**

**Medicine**

**Lunch**

**Dishes**

**Politics**

End of lecture 21