# DMA 9

Carl Dybdahl, Patrick Hartvigsen, Emil Søderblom

December 19, 2016

## Part 1

### (1)

We need to prove that a relation $\preceq$ defined on binary tuples of an ordered set $(A, \leq)$ is an ordering relation. To be specific, $\preceq$ is defined by:

$$(a_1, a_2) \preceq (b_1, b_2) \iff [(a_1 \neq b_1) \wedge (a_1 \leq b_1)] \vee [(a_1 = b_1) \wedge (a_2 \leq b_2)]$$

Note that there are two disjunct terms in this relation, so many proofs will proceed by case analysis.

**Theorem 1.** *$\preceq$ is reflexive.*

*Proof.* Given $(a_1, a_2)$, we need to show $(a_1, a_2) \preceq (a_1, a_2)$. We know that $a_1 = a_1$ and that $a_2 \leq a_2$, which makes the second disjunct term in the definition of $\preceq$ true and thus the proposition must hold. □

**Lemma 1.** *If $(a_1, a_2) \preceq (b_1, b_2)$ then $a_1 \leq b_1$.*

*Proof.* There are two cases to consider here: when $(a_1 \neq b_1) \wedge (a_1 \leq b_1)]$ holds and when $(a_1 = b_1) \wedge (a_2 \leq b_2)$ holds. In the first case, the second conjunct is exactly the proposition we wish to prove. In the second case, the first conjunct implies by reflexivity the proposition, i.e. $a_1 \leq a_1 = b_1$. □

**Theorem 2.** *$\preceq$ is antisymmetric.*

*Proof.* Assume $(a_1, a_2) \preceq (b_1, b_2)$ and $(a_1, a_2) \succeq (b_1, b_2)$. By lemma 1, we therefore have $a_1 \leq b_1$ and $a_1 \geq b_1$. This implies that $a_1 = b_1$, which means that only the second disjunct of $(a_1, a_2) \preceq (b_1, b_2)$ and $(a_1, a_2) \succeq (b_1, b_2)$ can be true. This means that $(a_1 = b_1) \wedge (a_2 \leq b_2)$ and $(a_1 = b_1) \wedge (a_2 \geq b_2)$. From this we can conclude that $a_2 = b_2$, which means that $(a_1, a_2) = (b_1, b_2)$. □

**Lemma 2.** *If $a_1 = b_1$ and $(a_1, a_2) \preceq (b_1, b_2)$ then $a_2 \leq b_2$.*

*Proof.* $\preceq$ is defined by a disjunction of two cases, and the first case is contradicted by $a_1 = b_1$. Therefore the second case must be correct, and it contains $a_2 \leq b_2$. □

**Theorem 3.** *$\preceq$ is transitive.*

*Proof.* Suppose $(a_1, a_2) \preceq (b_1, b_2) \preceq (c_1, c_2)$. Then either $a_1 \neq c_1$ or $a_1 = c_1$. In the first case, we apply lemma 1 twice to obtain $a_1 \leq b_1 \leq c_1$, which means that we have $(a_1, a_2) \preceq (c_1, c_2)$. In the second case, we have that $c_1 = a_1 \leq b_1 \leq c_1 = a_1$, so $b_1$ must be equal to $a_1$ and $c_1$. This lets us apply lemma 2 to obtain $a_2 \leq b_2 \leq c_2$, which leads us to conclude $[a_1 = c_1] \wedge [a_2 \leq c_2]$ and therefore $(a_1, a_2) \preceq (c_1, c_2)$. $\square$

## (2)

We were asked to topologically sort a set. We wrote the following algorithm to do it:

dma9.fsx - topSort

```
// first, define the relations on A and A*A
let aRel x y = (y % x = 0)
let aSqrRel (a1, a2) (b1, b2) =
   (a1 <> b1 && aRel a1 b1) || (a1 = b1 && aRel a2 b2)

// define the set to be sorted
let unsorted = Set.ofList [(2, 3); (4, 6); (2, 10); (10, 2); (30, 30); (2, 30)]

// define the strict version of the order relation
let bigger x y = aSqrRel y x && x <> y

// define a sorting algorithm
// NOTE: this is slow (like O(n^3)) and thus not suited for big inputs
let rec topSort set =
   if set = Set.empty then []
   else
      let found =
         set |> Set.toList
             |> List.tryFind (fun x -> not (Set.exists (bigger x) set))
      let elem = Option.get found
      elem :: topSort (Set.remove elem set)

// sort the set
let sorted = topSort unsorted

// output
printfn "%A" sorted
```

It yielded the result `[(2, 3); (2, 10); (2, 30); (4, 6); (10, 2); (30, 30)]`, which we verified is topologically sorted.

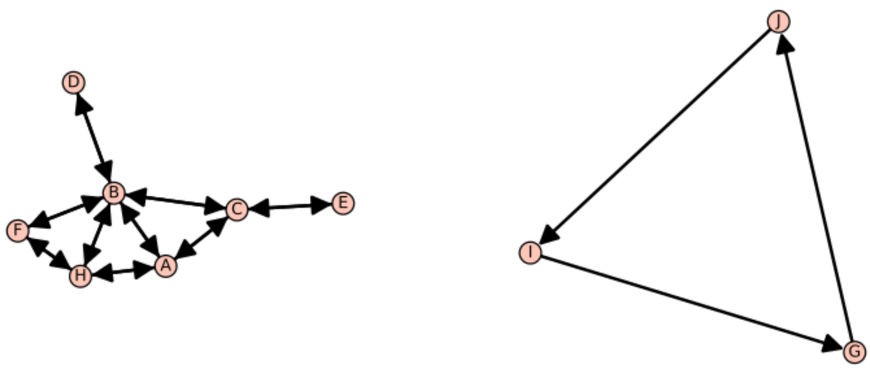
# Part 2

## a)

We define $R$ by the adjacency set:

$$\begin{aligned} R = \{&(A,B),(A,C),(A,H),\\ &(B,A),(B,C),(B,D),(B,F),(B,H),\\ &(C,A),(C,B),(C,E),\\ &(D,B),\\ &(E,C),\\ &(F,B),(F,H),\\ &(G,J),\\ &(H,A),(H,B),(H,F),\\ &(I,G),\\ &(J,I)\} \end{aligned}$$

This adjacency list works as a representation. Alternatively, we can represent $R$ as a directed graph:




## b)

$R^\infty$ is given by:

| | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $G$ | $H$ | $I$ | $J$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $A$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $B$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $C$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $D$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $E$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $F$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $G$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| $H$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| $I$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| $J$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |

### c)

As can be seen in the matrix in b), $R^\infty$ is already reflexive and so the reflexive closure of $R^\infty$ is the same as $R^\infty$.

We can easily read of from the matrix that $R^\infty$ relates all elements in the set $\{A, B, C, D, E, F, H\}$, that it relates all elements in the set $\{G, I, J\}$, and that it relates no other elements. From this it's easy to see that $R^\infty$ is an equivalence relation.