Tutorial 1 Exercise Solutions

COMP526: Efficient Algorithms

7–8 October, 2024

Exercise 1. Suppose A, B, and C are logical propositions. Which of the following expressions are logically equivalent to one another?

1. $A \implies B$
2. $(A \implies C) \land (B \implies C)$
3. $B \implies A$
4. $(A \land C) \lor (B \land C)$
5. $((A \land C) \implies B) \land ((A \land \neg C) \implies B)$
6. $(A \lor C) \land (B \lor C)$
7. $\neg A \lor B$
8. $(A \land B) \lor C$
9. $(A \lor B) \land C$
10. $(A \lor B) \implies C$
11. $\neg(B \land \neg A)$

Solution.

- $1 \iff 5 \iff 7$
- $2 \iff 10$
- $3 \iff 11$
- $4 \iff 9$
- $6 \iff 8$

□

Exercise 2. Consider a society consisting of a set S of people. We say that a person $p \in S$ is a ***dictator*** if for every $q \in S$, q obeys p. We say that S is a ***dictatorship*** if S contains a dictator.

1. Write the condition of S being a dictatorship in logical notation using the quantifiers $\forall$ and $\exists$ and the predicate $P(p,q)$ indicating that p obeys q.
2. Negate your expression from part 1 to obtain an expression for S not being a dictatorship.
3. How can you interpret the expression you devised for part 2 in plain English?

Proof.

1. $(\exists p)(\forall q)[P(q,p)]$
2. $(\forall p)(\exists q)[\neg P(q,p)]$

3. This expression indicates that for every person p, there is some person q that does not obey p (implying that p is not a dictator). Note that the expression $(\exists p)[\neg P(q,p)]$ is the negation of the expression for q to be a dictator: someone disobeys q.

$\square$

Exercise 3. Consider the following SELECTIONSORT algorithm.

```
1: procedure MININDEX(A, i, k)    ▷ Find and return the index of the minimum value
   in the array A between indices i and k, inclusive.
2:     m ← i
3:     for j = i, i+1, ..., k do
4:         if A[j] < A[m] then
5:             m ← j
6:         end if
7:     end for
8: return m
9: end procedure
10: procedure SELECTIONSORT(A, n)                    ▷ Sort the array A of size n
11:     for i = 1, 2, ..., n do
12:         j ← MININDEX(A, i, n)
13:         SWAP(A, i, j)
14:     end for
15: end procedure
```

Assume that the procedure MININDEX(A,i,j) correctly returns the index of the minimum value stored in A between the indices i and j (inclusive). The procedure SWAP(A,i,j) swaps the values of A at indices i and j. Prove that SELECTIONSORT correctly sorts every array A of size n. More specifically:

1. Identify a *loop invariant* that is satisfied at the end of each iteration of the loop in lines 11–14 of SELECTIONSORT.

2. Use mathematical induction to argue that your loop invariant holds.

3. Conclude that after the final iteration, the array is sorted (i.e., $A[1] \le A[2] \le \cdots \le A[n]$).

Solution. Consider the following invariant:

- After iteration i, $A[1..i]$ is sorted, and $A[i] \le A[j]$ for all $j \ge i$.

We argue that this invariant holds by induction on i.

Base case $i = 1$. In the first iteration, the index j stores the smallest value in the array in line 12. In line 13, this value is swapped into $A[1]$. Since the sub-array $A[1..1]$ has size 1, it is sorted, and $A[1] \le A[j]$ for all j after the swap.

Inductive Step. Suppose the loop invariant holds after iteration i. Consider iteration $i+1$ of the loop in lines 11–14. By the inductive hypothesis,

1. $A[1..i]$ is sorted, and

2. $A[i] \le A[j]$ for all $j \ge i$.

After lines 12–13, the smallest value in $A[i+1..n]$ is swapped to index $A[i+1]$. By 1 and 2 above, $A[1..i+1]$ is sorted, and by the minimality of $A[i+1]$, we have $A[i+1] \le A[j]$ for $j \ge i+1$. Therefore, the loop invariant holds after iteration $i+1$.

By induction, the loop invariant holds after every iteration of the loop. In particular, after iteration n, $A = A[1..n]$ is sorted, as desired. □