IDMA 2026
Problem set 1

Abdulkareem Al-Rifai, kbt185

16-02-26

Contents

4.a Theorem. All swans have the same colour. Proof. We prove by induction over n that any set of n swans have the same colour. For the base case, if we have a set of n 1 swan, then this swan vacuously has the same colour as itself. For the induction step, assume as our induction hypothesis that all sets of n swans have the same colour, and consider a set of n + 1 swans. Fix some swan Sl in this set. If we remove Sl, then we have n swans left, and by the induction hypothesis they all have the same colour. Let C be this colour. Now consider another swan S2 in the set. If we remove S2 from our set of n + 1 swans instead of Sl, then we again have n swans left, and they all have the same colour by the induction hypothesis. Since Sl is one of the swans in this set, it must have the same colour C as all the others. Hence, all n + 1 swans have the same colour. It follows by the principle of mathematical induction that any set of n swans must always have the same colour. 4

4.b Theorem. All positive integers are equal. Proof. By induction over n. For the base case, the positive integer 1 is equal to itself. For the induction step, assume as our induction hypothesis that n — 1 n. Adding 1 to both sides of this equality, we derive that n n + 1. It follows from this by the principle of mathematical induction that for all integers n the equality n n + 1 holds. But then by transitivity we obtain that all integers are equal, and the claim in the theorem statement has thus been established. 4

1 Provide formal proofs for the following claims

1.a For the sequence $a_1 = 1, a_n = 2 \cdot a_{n-1} + 1, \forall n \geq 2$ prove that $a_n = 2^n - 1, \forall n \in \mathbb{Z}^+$

Using principles of mathematical induction, we start by showing that the first proposition holds : Let $P(n)$ be the predicate $a_n = 2^n - 1$ Base case P(1) :

$$1 = a_1 = 2^1 - 1 = 1(HOLDS) \tag{1}$$

Now we try to prove that $P(k) \implies P(k+1)$ is a tautology

$$P(k) : a_k = 2^k - 1 \tag{2}$$

$$P(k+1) : a_{k+1} = 2^{k+1} - 1 \tag{3}$$

The next term in the sequence (left hand side) is

$$a_{k+1} = 2 \cdot a_{k-1+1} - 1 = 2 \cdot a_k - 1$$

substituting (2) in , we get

$$a_{k+1} = 2 \cdot (2^k - 1) + 1 = 2^{k+1} - 1 \tag{4}$$

We have shown that $a_n = 2^n - 1$ holds for all values $n \in \mathbb{N}^+$

1.b Prove that $\forall n \in \mathbb{N}^+$ it holds that $3|4^n + 5$

Base case : $P(n_0)$ We start with proving the base case $n_0 = 1$ is true

$$P(n_0) : 4^1 + 5 = 9 \tag{5}$$

$$3|9 : True \tag{6}$$

induction step :

$$P(k) : 3|4^k + 5 \tag{7}$$

let $f(k) = 4^k + 5$

$$P(k+1) : 3|4^{k+1} + 5 \tag{8}$$

The main point here is to manipulate $P(k+1)$ into an expression in terms of $P(k)$

$$P(k+1) : 4^{k+1} + 5 = 4 * 4^{k+1} + 5 = 4^k + 4^k + 4^k + 4^k + 5 = 3 \cdot 4^k + (4^k + 5) \tag{9}$$

We can write now that $P(k+1) = f(k) + s, s \in \mathbb{N}$. We have already proved that 3 divides $f(k)$, if we can prove that 3 also divides s, we can use use the properties of integers and division and say that if $a|b \wedge a|c$ then $a|b+c : T$ $s = 3 \cdot 4^k$, since s is a multiple of 3, we can easily conclude that s is divisible by 3, hence $3|f(k) + s : T$

2 For each of the propositional logic formulas below, determine whether it is a tautology or not. If the formula is not a tautology, show how to add a single connective to make it into a tautology. Please make sure to justify your answers (e.g., by presenting truth tables, or by using rules for rewriting logic formulas that we have learned in class).

2.a $\neg((p \to q) \lor r) \to ((\neg q \land \neg r) \land p)$

2.b $((p \land q) \to r) \leftrightarrow ((q \lor r) \lor \neg p)$

3 One annoying feature of merge sort is that it ignores long runs of already sorted elements. Your task in this problem is to analyse a modified algorithm by Jakob intended to fix this. The merge part of the algorithm would be essentially the same as before:

3.a Is the pseudocode algorithm above correct in that for any input array A it will be the case that mergerunssort (A) returns the same array but sorted in increasing order?

3.b Regardless of whether the sorting algorithm is correct or not, does it always terminate (assuming that the input is an array of elements that can be compared with ¡=) and, if so, what is the worst-case time complexity?

3.c Can you give any example of a family of input arrays of growing size for which Jakob's merge runs sort algorithm will output a correctly sorted array and be asymptotically faster than the merge sort algorithm that we have covered in class? Can you give any example of a family of input arrays of growing size for which merge runs sort will be asymptotically slower than standard merge sort?

4 When constructing this problem set, Jakob ran into some very unfortunate problems. When he wanted to add a couple of more examples illustrating the power of inductive proofs, the proofs turned out to be a little bit too powerful. Specifically, Jakob was able to use mathematical induction to show 1. that all swans have the same colour (presumably white, so that there are no black swans after all), and 2. that all positive integers are in fact equal. Both of these claims are fairly disturbing from a mathematical point of view. Please help Jakob by pointing out clearly what goes wrong in his induction proofs below.

4.a Theorem. All swans have the same colour. Proof. We prove by induction over n that any set of n swans have the same colour. For