Data Structures And Algorithms - Problem Sheet 1

Dom Hutchinson

October 5, 2018

Question 1 - Teams
Prove using strong induction that for all $n \geq 18$, a group of n people can be divided into teams so that each team either has size 4 or size 7.

My Solution 1
Base Case

$$\begin{array}{ll} 18 = 4+7+7 & 20 = 4+4+4+4+4 \\ 19 = 4+4+7 & 21 = 7+7+7 \end{array}$$

Inductive Assumption
Let $m > 21, m \in \mathbb{N}$ and assume all integers less that it can be split in groups of 4s & 7s.

Inductive Case
Consider $m - 4$.
Since this is less than m then, by the inductive assumption, it can be produced by a combination of 4s & 7s.
Then $\exists\ x, y \in \mathbb{N}\ st\ m - 4 = 4x + 7y \implies m = 4(x + 1) + 7y$.
So m is a can be produced by a combinations of 4s & 7s.
Thus, by the principle of strong induction, all integers greater than, or equal to, 18 can be produced by a sum of 4s & 7s.

Question 2 - Ningi
On the isle of Magarathea, which you are visiting to offer your services as a contract programmer, the unit of currency is the Ningi. Your Magarathean customers will pay you 3 Ningi for each programme you write in an imperative language and 8 Ningi for each programme you write in a functional language. The Ningi is not exchangable for sterling, so you would like ot spend all the currency that you earn before leaving the island. What is the least number n such that for all prices $m \geq n$ you can earn exactly m Ningi?
Prove you answer.

My Solution 2
13 cannot be produced by a combination of 3s & 8s.

Base Case

$$14 = 3+3+8 \quad 15 = 3+3+3+3+3 \quad 16 = 8+8$$

Inductive Assumption
Let $m > 16, m \in \mathbb{N}$ and assume all integers less than it can be split into groups of 3s & 8s.

Inductive Case
Consider $m - 3$.
Since this is less than m then, by the inductive assumption, it can be produced by a combination

of 3s & 8s .
Then $\exists\ x, y \in \mathbb{N}\ st\ m-3 = 3x+8y \implies m = 3(x+1)+8y$.
So m is a can be produced by a combinations of 4s & 7s.
Thus, by the principle of strong induction, all integers greater than, or equal to, 14 can be produced by a sum of 3s & 8s.

Question 3 - Bubblesort Correctness
Bubblesort is an algorithm for sorting an array A. Its pretty, although not very efficient. It works by swapping adjacent elements that are out of order. Here is pseudocode for Bubblesort:

```
BUBBLESORT(A)
n=A.setlength
for i=1 to n-1
  for j=n downto i+1
    if A[j] < A[j-1]
      exchange A[j] with A[j-1]
```

Prove the correctness of the Bubblesort algorithm. Structure your proof in steps of initialization, maintenance and termination for the inner for-loop starting at line 3, and then initialization, maintenance and termination for the outer for-loop starting at line 2.
Hint: as part of your proof, show that the following is a preserved invariant for the inner for-loop:
- $A[j] = min\{A[k] : j \leq k \leq n\}$
and use this result to prove that the following is a preserved invariant for the outer for-loop:
- For $1 \leq j \leq i-1$, $A[j]$ is the j^{th} smallest element of A.

My Solution 3
Inner Loop
To prove the invariant "$A[j] = min\{A[k] : j \leq k \leq n\}$" is preserved.
Initialisation
Set $j = 1$. Since the array contains only one item then $A[1]$ is the lowest value be default.

Maintenance
This loop moves $A[j]$ to the left of the array until no elements to its right are less than it. Assuming this holds for all prior passes of the loop then $A[j] = min\{A[k] : j \leq k \leq n\}$. Thus invariance holds.

Termination
Since the loop has a definite termination value then termination is quaranteed.

Since the invariant, "$A[j] = min\{A[k] : j \leq k \leq n\}$", holds for initialisation, maintenance & termination it is a preserved invariant of the inner loop.

Outer Loop
To prove the invariant *"for $1 \leq j \leq i-1$, $A[j]$ is the j^{th} smallest element of A"* is preserved.
Initialisation
Set $i = 2$. Since the list has only one element, its first, & only, element is the smallest element of A.

Maintenance
By the invariant proved for the inner loop we know the first $j-1$ elements of A are the $j-1$ smallest elements, thus j is the j^{th} smallest element of $A[1, \dots, j]$.

Termination
As this loop has a definite termination value so it definetly terminants.

Since the invariant, "for $1 \leq j \leq i-1$, $A[j]$ is the j^{th} smallest element of A.", holds for initialisation, maintenance & termination it is a preserved invariant of the outer loop.

Since both invariants have been proved, the correctness of bubblesort producing a sorted list is proved.

Question 4 - Cicle of Discs
There are seven discs, labelled with the numbers 1 to 7, arranged in seven positions that form a circle; we will label the positions with the numbers 1 to 7 too, and write p(i) for the position of the disc with label i. You can move the discs one of two ways. One way is to cycle the circle: in pseudocode,

```
for i=1 to 7
  if p(i) < 7
    p(i) = p(i) + 1
  else
    p(i)=1
```

The other way is to cycle the set of three discs in positions 1, 2, 3: in pseudocode,

```
for i=1 to 3
  if p(i) < 3
    p(i) = p(i) + 1
  else
    p(i)=1
```

Given a set of positions of the discs, a transposition for this position is a pair (i, j) such that $1 \leq i < j \leq 7$ and $p(j) < p(i)$. The discs are initially arranged so that each disc is at the position with the corresponding label, i.e. $p(i) = i$ for all $1 \leq i \leq 7$.

Question 4.1
Show that if $1 \leq p(i) < p(j) \leq 7$, (i, j) is a transposition immediately after the circle is cycled iff and only if : before the cycle was cycled EITHER (i, j) was a transposition and p(j) was less than 7, OR (i, j) was not a transposition and p(j) was equal to 7.

My Solution 4.1
Let $1 \leq p(i) < p(j) < 7,\ i, j \in \mathbb{N}$ then after cycling we have $2 \leq p(i) < p(j) \leq 7$. Since the order hasn't changed, and the labels are constant, this is a transposition iff it was before cycling.
Let $1 \leq p(i) < p(j) = 7,\ i, j \in \mathbb{N}$ then after cycling we have $1 \leq p(j) < p(i) \leq 7$. Since the order has changed, and the labels are constant, this is a transpition iff it was not before cycling.

Question 4.2
Prove that if the number of transpositions is even before the circle is cycled, it is even after the circle is cycled.

My Solution 4.2
Let $1 \leq i, \leq 7$ and set $p(i) = 7$, then a transposition (i, j) is formed with all $j > i$. There are $7 - i$ such transpositions before cycling.
After cycling, all previous transpositions are lost and new transpositions, (j, i), are formed with

all $j < i$, there are $i - 1$ such transpositions.
Thus there is a net change of $|(i-1)-(7-i)| = |2i-8|$. Since this value is always even, then the total number of transpositions remains even after cycling, if it was before cycling.

Question 4.3
By using a similar argment for cycling the set of three disks, show that it is a preserved invariant of the system that the number of transpositions is even.

My Solution 4.3
To prove the invariant *"the number of transposition is even"* is preserved.
Initialisation
When $p(i) = i\ \forall i \in \mathbb{N}, 1 \leq i \leq 7$ there are 0 transpositions. So the number of transpositions is even.

Maintenance
All transpositions $(i,j),\ p(i) \geq 4,\ p(j) \leq 3$ are unchanged
Let $p(j) = 3$ and $p(i) = \{1,2\}$ for $i \neq j$.
If (i,j) is a transposition before cycling three disks then it no longer is after, as the positions have changed but the labels are constant.
The inverse is true, if (i,j) is *not* a transposition before cycling, then it is afterwards.
This means either: two transpositions are gained; one transposition is gained and one is lost; or two transpositions are lost. Each of these outcomes has a net change in transpositions of zero or two.
Thus, if the number of transpositions is even before cycling three disks then it is afterwards.
The same was proved for cycling all disks in *Question 4.2*. The invariant holds for all actions.

Termination
As loop is over a finite number of elements, it always terminants.

As the invariant, *"the number of transposition is even"*, holds for initialisation, maintenance & termination it is a preserved invaraint.

Question 4.4
Deduce that it is not possible to reach the arrangement $p(1) = 2, p(2) = 1, p(i) = i$ for $i > 2$ by performing a sequence of cycles of the circle or of the three discs in positions $1, 2, 3$.

My Solution 4.4
Here the only transposition is $(1,2)$. Thus there are an odd number of transpositions, this breaks the preserved invariant so cannot be true. Thus this is an impossible state of the system.

Question 5 - Vegans
As stated in the lecture on stable matching, a matching is unstable if there is a pair who are not matched, but where both prefer eachother to their assigned matches.
An even number of students are trying to sort themselves into pairs for sharing 2-person rooms. Some of these students are vegan. The vegans would all rather share with another vegan than with any of the non-vegans. Each of the vegans is the first choice (as a room mate) of another vegan.

Question 5.1
Show that if the number of vegan students is odd, there is no stable roommate assignment.

My Solution 5.1
If the number of vegans is odd, then a vegan must share with a non-vegan. Call then v
By the question, v prefers all other vgeans over this non-vegan.
Since each vegan is the first choice of another vegan, and one cannot be the first choice of themselves, there is a vegan in another pair who would rather be with v than anyone else.
This includes there current partner, creating an unstable match.
This holds for all situtations with an odd number of vegans.

Question 5.2
Show that if the nunber of students is four, and exactly two of them are vegans, then the room assignment matching the two vegans together and the two non-vegans together is stable.

My Solution 5.2
Let v_1 & v_2 be vegans.
By the conditions of the question, v_1's first preference must be v_2 & v_2's must be v_1.
So when they are paired together, neither wish to form a pair with anyone else.
As there is only one other pair in this scenario, an unstable match cannot be formed.

Question 6 - Clients and Consultants
You are running a small computer consultancy, managing three employees (Alex, Brandon and Carmen). Three clients, Datadatadata, ElephantineLogs and FilesRUs (referred to as D, E and F, for short), would each like to use one of your consultants on a project to rationalize their storage systems. The clients and employees all have previous experience of eachother and have formed preferences as to who they would most like to work with.

Alex would most like to work with D, followed by E and then F.
Brandon would most like to work with F, followed by E and then D.
Carmen would most like to work with D, followed by E and then F.
D would most like to work with Brandon, followed by Alex and then Carmen.
E would most like to work with Brandon, followed by Carmen and then Alex.
F would most like to work with Carmen, followed by Alex and then Brandon.

Assuming that you want the matching to be stable, but that subject to that, the preferences of your clients outweigh those of your employees, which employee would you assign to each of the clients? Explain your answer, using a property of the Gale-Shapley algorithm (aka the Algorithm of Happiness).

My Solution 6
Since the preferences of the clients outweigh those of the employees I shall implement the *Gale-Shapely Algorithm* which the clients proposing to the employees.
The initial preference tables:

A	D	E	F
B	F	E	D
C	D	E	F

D	B	A	C
E	B	C	A
F	C	A	B

D & E propose to B, since B prefers E to D they are paired together.
F proposes to C, C has no other offers so accepts.

A	D	E	F
B	F	E	D
C	D	E	F

D	B	A	C
E	B	C	A
F	C	A	B

D is the only client without an employee so proposes to A.
A has no other offers so accepts.

A	D	E	F
B	F	E	D
C	D	E	F

D	B	A	C
E	B	C	A
F	C	A	B

Since everyone now has a partner, the algorithm terminates.
Alex is assigned to Datadatadata.
Brandon is assigned to ElephantineLogs.
Carmen is assigned to FilesRUs.