1 Tutorial problems

Recursive definitions

1. *Give a recursive definition of the set $P = \{1, 2, 4, 8, 16, \ldots\}$ of powers of two within $\mathbb{N}$. (You may assume the operation $+$ on $\mathbb{N}$ has already been defined.)*

   Base $1 \in P$

   Recursion If $x \in P$ then $x + x \in P$.

2. *Give a recursive definition of the subset, Eq, of $\mathbb{N} \times \mathbb{N}$ representing the relation* is equal to *using the successor function $s(n) = n + 1$.*

   Base $(0, 0) \in Eq$

   Recursion If $(x, y) \in Eq$ then $(s(x), s(y)) \in Eq$.

   Minor note here, the "recursion" should not assume that the only pairs in Eq have equal first and second coordinates (i.e., shouldn't start "if $(x, x) \in Eq$") since that's what we're really trying to define.

3. *Give a recursive definition of "linked list of natural numbers" (hint: the base case should be equivalent to `nil`, and the construction should use a pair).*

   Let's use LL to stand for "linked list of natural numbers", and as suggested we'll use an atomic case to stand for the empty list - I'll denote that by λ.

   Base $\lambda \in LL$

   Recursion If $h \in \mathbb{N}$, and $t \in LL$ then $(h, t) \in LL$.

   With this definition we can now recursively define all the usual functions we'd want for linked lists. For instance: "the length of λ is 0, and the length of (h, t) is the length of t plus one", "the concatenation of λ with any list a is just a, and the concatenation of (h, t) with a is (h, b) where b is the concatenation of t with a."

4. *Give a recursive definition of the set of* finite *subsets of* $\mathbb{N}$*, using the successor function* s *and union as the operators.*

   Let $F\mathbb{N}$ denote the finite subsets of $\mathbb{N}$. There are many possible choices - the one here is designed to ensure that each set has a unique "parent" – this is often a desirable characteristic of recursive definitions as it can help to simplify some proofs.

   Base $\emptyset \in F\mathbb{N}$

   Recursion If $a \in F\mathbb{N}$ then $\{0\} \cup A \in F\mathbb{N}$ and $\{s(x) : x \in A\} \in F\mathbb{N}$.

Inductive proofs

1. *Prove that* $2 + 5 + 8 + \cdots + (3n - 1) = n(3n + 1)/2$ *for all* $n > 0$.

   The base case is $n = 1$ for which the left hand side is 2 and the right hand side is $1(4)/2 = 2$. So the result holds in this case.

   Suppose that the result holds for $n = k$, that is:

   $$2 + 5 + 8 + \cdots + (3k - 1) = k(3k + 1)/2.$$

   Now consider $n = k + 1$. The left hand side is:

   $$\begin{aligned} 2 + 5 + 8 + \cdots + (3k - 1) + (3(k + 1) - 1) &= k(3k + 1)/2 + (3k + 2) \\ &= (3k^2 + 7k + 4)/2. \end{aligned}$$

   On the other hand, the right hand side is:

   $$(k + 1)(3k + 4)/2 = (3k^2 + 7k = 4)/2.$$

   Since the left hand side and right hand side are the same, the result holds for $n = k + 1$, and hence by induction for all $n > 0$.

   Of course, as noted in class the "correct" proof is "right the sum down again backwards, add corresponding pairs, then divide by 2".

2. *Prove that* $1 + 2 + 2^2 + \cdots + 2^n = 2^{n+1} - 1$ *for all* $n \geq 0$.

   The base case is $n = 0$ and the left hand side is 1 while the right hand side is $2^1 - 1 = 1$ so the result holds in this case. Suppose that the result holds for $n = k$, that is:

   $$1 + 2 + 2^2 + \cdots + 2^k = 2^{k+1} - 1$$

   Now consider $n = k + 1$. The left hand side is:

   $$1 + 2 + 2^2 + \cdots + 2^k + 2^{k+1} = 2^{k+1} - 1 + 2^{k+1} = 2^{k+2} - 1$$

   which matches the right hand side. So, in this case, the result holds for $n = k + 1$ and hence, by induction, for all n.

   Again a "better" proof is "double the sum and subtract the original, cancelling common terms".

3. [tSuppose that two algorithms do the same thing, but the first requires $f(n) = 4n+1$ steps, while the second requires $g(n) = n^2$ steps. For small values of n, the second algorithm is better, but it seems clear that when n is "big enough" we should prefer the first. Make this precise by proving that $f(n) < g(n)$ for all $n \geq 5$.

   The base case is now $n = 5$ and there $f(n) = 21 < 25 = g(n)$ so the result holds.

   Now suppose that $k \geq 5$ and $f(k) < g(k)$ then:

   $$f(k + 1) = 4(k + 1) + 1 = 4k + 1 + 4 = f(k) + 4 < g(k) + 4 = k^2 + 4$$

   On the other hand

   $$g(k + 1) = (k + 1)^2 = k^2 + 2k + 1 = k^2 + 4 + (2k - 3) > k^2 + 4$$

   since $2k - 3 > 0$ if $k \geq 5$. So, in this case $f(k + 1) < g(k + 1)$ and hence inductively $f(n) < g(n)$ for all $n \geq 5$.

4. *Consider the following recursively defined function $f : \mathbb{N} \to \mathbb{N}$.*

$$\begin{aligned} f(0) &= 0 \\ f(n+1) &= f(n) + 2n + 1 \end{aligned}$$

Compute the first few values of f, form a conjecture for a more "natural" description of f, and prove that your conjecture is correct.

Inspection of the first few values suggests that $f(n) = n^2$. This is obviously true for $n = 0$. If true for $n = k$ then

$$f(k+1) = f(k) + 2k + 1 = k^2 + 2k + 1 = (k+1)^2$$

so also at $k + 1$, and hence by induction for all n.

5. *Let a and b be two symbols. Recursively define a set of strings (sequences), D as follows:*

Basis *The empty string is in D.*

Recursive step *If a string s is in D, then so is the string asb. If strings s and t are in D then so is the string st.*

(a) *List all the strings in D consisting of 6 or fewer symbols.*
Using λ for the empty string.

$$\lambda, ab, aabb, abab, aaabbb, aababa, aabbab, ababab, abaabb.$$

(b) *Prove by induction that every string in D has even length.*
This is true for strings specified in the basis step. In the recursive step, the strong produced is either two symbols longer than some other string in D or has length the sum of the lengths of two other strings in D. In either case, assuming inductively that the inputs to that process have even length, so will the outputs, and thus all strings in D must have even length.

(c) *Prove by induction that every string in D has the same number of a's as b's.*
As above, this is true in the base case, and clearly preserved in the recursive construction.

(d) *Prove by induction that in any prefix of a string in D there are at least as many a's as b's.*

True in the base case. In recursive constructions of the first type, all prefixes have one more a than a prefix of s (and no extra b's) so still have at least as many a's as b's. In the second type, prefixes are either prefixes of s (so have at least as many a's as b's) or are s (having equal numbers) concatenated with a prefix of t (having at least as many a's as b's). In either case they still have at least as many a's as b's and so inductively all strings in D have that property.

6. *Consider the following piece of pseudocode defining a function* $\mathsf{foo}(x, y)$*, where we assume* $x, y \in \mathbb{N}$ *(the division operator is the usual e.g., Java, integer division, and % is the remainder operation). To maintain some mathematical integrity we use* $=$ *in the mathematical sense (i.e. of an equality test) and use* $\leftarrow$ *to denote assignment.*

if $x = 0$ **then**
  `return` 0
end if
$s \leftarrow \mathsf{foo}(x/2, 2y)$
if $x\%2 \neq 0$ **then**
  $s \leftarrow s + y$
end if
`return` s

Compute a table of values for foo*, form a conjecture about its more natural definition, and prove that conjecture by induction.*

$\mathsf{foo}(x, y) = x \times y$.

The proof is by induction on x. For $x = 0$ it's obviously true. For even $x > 0$ say $x = 2a$,

$$\mathsf{foo}(x, y) = \mathsf{foo}(2a, y) = \mathsf{foo}(a, 2y) = a \times 2y = 2a \times y = x \times y$$

where we've used the inductive hypothesis in replacing $\mathsf{foo}(a, 2y)$ by $a \times 2y$. For odd $x > 0$, say $x = 2a + 1$ similarly:

$$\mathsf{foo}(x, y) = \mathsf{foo}(2a{+}1, y) = \mathsf{foo}(a, 2y){+}y = a{\times}2y{+}y = 2a{\times}y{+}y = (2a{+}1){\times}y = x{\times}y.$$

So, by induction the result is true for all x.