COMPUTER SCIENCE TRIPOS Part IA 2019
Paper 2 Question 5

Thomas Forster

February 19, 2020

A binary relation $\prec$ on a set A is **well-founded** iff there are no infinite descending chains: $\ldots \prec a_i \prec \ldots \prec a_1 \prec a_0$.

(a) Show that a binary relation $\prec$ on a set A is well-founded iff any nonempty subset Q of A has a minimal element[1], i.e., an element m such that

$$m \in Q \wedge (\forall b)(b \prec m \rightarrow b \notin Q)$$

[5 marks]

(b) Show that defining

$$\langle n_1, n_2\rangle \prec \langle {n_1}', {n_2}'\rangle \longleftrightarrow \langle n_1, n_2\rangle \neq \langle {n_1}', {n_2}'\rangle \text{ and } n_1 \leq n_1' \text{ and } n_2 \leq n_2'.$$

determines a well-founded relation between pairs of positive natural numbers.
[7 marks]

(c) Let $\rightarrow$ be a binary relation between pairs of positive natural numbers for which $\langle m, n\rangle \rightarrow \langle m, n-m\rangle$ if $m < n$, and $\langle m, n\rangle \rightarrow \langle m-n, n\rangle$ if $n < m$.

Using (a) and (b), or otherwise, show that for all pairs of positive natural numbers $\langle m, n\rangle$, there is a natural number h such that $\langle m, n\rangle \rightarrow \langle h, h\rangle$.
[8 marks]

Discussion

Well-founded relations are important and you need to know about them. That is beco's they support a kind of induction (a generalisation of "strong induction" on the naturals) called *wellfounded induction*. This question is as good a way in as you are likely to find, which is why i am writing out this discussion answer.

[1] not neccessarily unique

(a)

This invites you to prove the equivalence of two definitions of "wellfounded". It is the second definition that is the primary one, in that it captures the property a relation has to have if it is to support (wellfounded) induction. The first definition—the "descending chain condition"—is easier to understand, and is equivalent to the second if we have the axiom of choice.

The first condition obviously implies the second because an infinite descending chain would be a '("bad") Q without a minimal element. For the other direction suppose there is a "bad" Q. Using the axiom of choice we build an infinite descending chain: no element is minimal so we can always find an element below the last member of the chain we are building.

For a concise introduction to wellfounded induction look at (e.g.) pp 11–12 of `https://www.dpmms.cam.ac.uk/~tf/cam_only/partiiicomputability.pdf` or read the section in *Logic, Induction and Sets*.

(b)

You will probably recognise this ordering as the *lexicographic ordering* on $\mathbb{N}\times\mathbb{N}$. Put another way, it is the *lexicographic product* of two copies of the strict ordering $\langle\mathbb{N}, <_{\mathbb{N}}\rangle$. It's wellfounded beco's it's a lexicographic product of two wellfounded strict total orders (aka *wellorderings*). This last fact (that a lexicographic product of two wellfounded strict total orders is likewise wellfounded) is worth committing to memory, so let's prove it—or rather this particular instance. Suppose we had an infinite descending chain of pairs of natural numbers. Keep you eye on the *first* components of the pairs you see as you descend. They cannot get bigger, and they cannot decrease indefinitely ($\langle\mathbb{N}, <_{\mathbb{N}}\rangle$ is wellfounded, after all) so are eventually constant. So, after a while, all the pairs have the same first component. Ignore all pairs above that point. Now look at the second components of the surviving pairs. The same line of argument applies so they are eventually constant, and we have found our minimal element. And of course it's unique.

(c)

We are being invited to build a descending chain by the following method. If we have a pair $\langle m, n\rangle$ in our hand, we put below it either $\langle m, n-m\rangle$ (if $m < n$) or $\langle m-n, n\rangle$ (if $m > n$); if $m = n$ we do nothing. Observe that both $\langle m, n-m\rangle$ and $\langle m-n, n\rangle$ come below $\langle m, n\rangle$ in the lexicographic order. So we cannot do this infinitely often. So we have to stop! But the only way we can stop is if we reach a stage where the two components are the same.

You will presumably have spotted the connection to Euclid's division algorithm. This is the standard proof that that algorithm always halts. That was always sort-of obvious, so what we have done here is not so much prove that Euclid's algorithm is good but rather show that it is good *for specific reasons which can be deployed elsewhere*. (For example to proving that the Ackermann function is defined everywhere).