Recursive relations (Part 2/2)

Bounded $\exists$

Definition. Let $R(x, i)$ be a relation on N. The relation $\exists R$ obtained from R by **bounded existential quantification** is defined as follows:

$$(\exists R)(x, y) \quad \text{iff} \quad \exists i \leq y : R(x, i).$$

Remark: this is the same as defining

$$(\exists R)(x, y) \quad \text{iff} \quad R(x, 0) \text{ or } R(x, 1) \text{ or } \ldots \text{ or } R(x, y).$$

Example

The usefulness of bounded quantification is illustrated by the following definition of the relation $divides(x, y)$ that holds if and only if x divides y:

$$divides(x, y) \quad \text{iff} \quad \exists i \leq y : x \cdot i = y.$$

Bounded $\forall$

Definition. Let $R(x, i)$ be a relation on N. The relation $\forall R$ obtained from R by **bounded universal quantification** is defined as follows:

$$(\forall R)(x, y) \quad \text{iff} \quad \forall i \leq y : R(x, i).$$

Remark: this is the same as defining

$$(\forall R)(x, y) \quad \text{iff} \quad R(x, 0) \text{ and } R(x, 1) \text{ and } \ldots \text{ and } R(x, y).$$

Summation and product

To show the desired properties of bounded quantification, we introduce **summation** $\sum$ and **product** $\prod$:

Definition. Given a total function $f : N^{k+1} \times N \to N$,

- the **summation** $\sum_{i=0}^{y} f(x,i)$ over f is defined as the total function $g : N^{k+1} \to N$ given by

$$g(x,y) = f(x,0) + f(x,1) + \cdots + f(x,y);$$

- the **product** $\prod_{i=0}^{y} f(x,i)$ over f is defined as the total function $g : N^{k+1} \to N$ given by

$$g(x,y) = f(x,0) \cdot f(x,1) \cdot \cdots \cdot f(x,y).$$

Exercise

Show the following proposition:

Proposition. Let $f : N^{k+1} \longrightarrow N$ be a total function.

- If f is primitive recursive, then so are the summation over f and the product over f.
- If f is recursive, then so are the summation over f and the product over f.

Back to bounded $\exists$ and $\forall$

Proposition. If $R(x, i)$ is a (primitive) recursive relation, then so are $\forall R$ and $\exists R$.

Proof. Letting x stand for $x_1, \ldots, x_k$, the characteristic functions of

$$\forall i \leq y. R(x, i) \quad \text{and} \quad \exists i \leq y. R(x, i)$$

are

$$\prod_{i=0}^{y} \xi_R(x, i) \quad \text{and} \quad sg(\sum_{i=0}^{y} \xi_R(x, i)).$$

Bounded maximization

Definition. Given a relation $R(x, i)$, the total function $Max[R]$ obtained from R by **bounded maximization** is defined as follows:

$$Max[R](x,y) = \begin{cases} \text{the largest } i \leq y \text{ for which } R(x,i) & \text{if such a } i \text{ exists} \\ 0 & \text{otherwise.} \end{cases}$$

Bounded maximization

Proposition. If $R(x_1, \ldots, x_k, i)$ is a (primitive) recursive relation, then so is $Max[R]$.

Proof. (Sketch.) Define the relation $S(x, y, i)$ by

$$S(x, y, i) \text{ iff } \exists j \leq y . j > i \text{ and } R(x, j).$$

Then

$$Max[R](x, y) = \sum_{i=0}^{y} \xi_S(x, y, i).$$

Quotient and remainder

So the quotient function

$$quo(x, y) = \begin{cases} \text{The largest } z \leq x \\ \text{such that } y \cdot z \leq x & \text{if } y \neq 0 \\ \\ 0 & \text{if } y = 0. \end{cases}$$

Using our new knowledge about primitive recursive relations, we can tell that the function quo is primitive recursive. The remainder function is also primitive recursive, because

$$rem(x, y) = x \dot{-} y \cdot quo(x, y)$$

Logarithms

- The logarithm $\log(x, b)$ of a number x w.r.t. a base b is a number l such that $b^l = x$ (if l exists).
- What is $\log(2, 4)$?
- Logarithms, when applied to natural numbers, can yield non-natural numbers.
- However, there are very useful modified versions that do not suffer from this problem…

Logarithms

$$lo(x,b) = \begin{cases} \text{the greatest } z \text{ such that } divides(b^z, x) & \text{if } b > 1 \text{ and } x > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Using our new knowledge about primitive recursive relations, we can tell that the function lo is primitive recursive.

Encoding/decoding pairs, triples, etc.

- As we have seen in the first week, pairs of natural numbers can be encoded as follows, where p and q are different prime numbers:

$$c(x, y) = p^x \cdot q^y.$$

- E.g., we could use the encoding

$$c(x, y) = 2^x \cdot 3^y.$$

Encoding/decoding pairs, triples, etc.

- Similarly, triples can be encoded by

$$triple(x, y, z) = 2^x \cdot 3^y \cdot 5^z.$$

- Because exponentiation and multiplication are primitive recursive, so is $triple$.
- Logarithms provide the decoding:

$$\begin{aligned} first(n) &= lo(n, 2) \\ second(n) &= lo(n, 3) \\ third(n) &= lo(n, 5). \end{aligned}$$

Exercise

Show that the function $\gcd(x, y)$ that returns the greatest common divisor of x and y is primitive recursive. (You can use the primitive recursive function $divides$ introduced earlier.)

Exercise

The n-th **Fibonacci number** $\mathit{fib}(n)$ is determined by the following conditions:

$$\mathit{fib}(0) = \mathit{fib}(1) = 1$$
$$\mathit{fib}(n+2) = \mathit{fib}(n+1) + \mathit{fib}(n),$$

so we get the sequence of pairs $1, 1, 2, 3, 5, 8, 13, 21, \ldots$. Show that fib is primitive recursive. (Hint: consider the sequence $(1,1), (1,2), (2,3), (3,5), (5,8), (8,13), \ldots$, and recall that we have primitive recursive functions for encoding and decoding pairs.)