THEORY OF COMPUTATION
Primitive Recursive Predicates and Minimalization - 6

Prof. Dan A. Simovici

UMB
Outline

1. Bounded Minimalization
2. Unbounded Minimalization
3. Conclusion
Theorem

Let \( P(t, x_1, \ldots, x_n) \) be a predicate that belongs to some PRC class \( \mathcal{C} \). Define the function \( f(y, x_1, \ldots, x_n) \) as having the least value \( t \) such that \( t \leq y \) for which \( P(t, x_1, \ldots, x_n) \) is TRUE, if such a value exists. Otherwise, this value is 0. The function \( f \) belongs to the same PRC class \( \mathcal{C} \).

The function \( f \) is denoted as

\[
f(y, x_1, \ldots, x_n) = \min_{t \leq y} P(t, x_1, \ldots, x_n)
\]

and the construction of \( f \) is *bounded minimalization*. 
Proof.

Let \( g(y, x_1, \ldots, x_n) \) be the function defined by:

\[
g(y, x_1, \ldots, x_n) = \sum_{u=0}^{y} \prod_{t=0}^{u} \alpha(P(t, x_1, \ldots, x_n)).
\]

This function belongs to \( \mathcal{C} \) by a previous theorem.

We claim that \( g(y, x_1, \ldots, x_n) \) is the least value of \( t \) for which \( P(t, x_1, \ldots, x_n) = 1 \) (that is, \( P(t, x_1, \ldots, x_n) = 1 \) is TRUE).

Indeed, suppose that for some value of \( t_0 \leq y \) we have:

- \( P(t, x_1, \ldots, x_n) = 0 \) for \( t < t_0 \), and
- \( P(t_0, x_1, \ldots, x_n) = 1 \).
Proof cont’d

Proof.

In other words, \( t_0 \) is the \textbf{least value} of \( t \leq y \) for which \( P(t, x_1, \ldots, x_n) \) is TRUE.

Note that

\[
\prod_{t=0}^{u} \alpha(P(t, x_1, \ldots, x_n)) = \begin{cases} 
1 & \text{if } u < t_0, \\
0 & \text{if } u \geq t_0.
\end{cases}
\]

Therefore,

\[
g(y, x_1, \ldots, x_n) = \sum_{u < t_0} 1 = t_0,
\]

hence \( g(y, x_1, \ldots, x_n) \) is the least value of \( t \) for which \( P(t, x_1, \ldots, x_n) \) is TRUE.
Proof cont’d

Proof.

Now we define

$$\min_{t \leq y} P(t, x_1, \ldots, x_n) = \begin{cases} g(y, x_1, \ldots, x_n) & \text{if } (\exists t) \leq y P(t, x_1, \ldots, x_n) \\ 0 & \text{otherwise.} \end{cases}$$

This shows that $\min_{t \leq y} P(t, x_1, \ldots, x_n)$ belongs to $C$. \hfill \square
The bounded minimalization allows the definition of further primitive recursive functions.

**Example**

\[ \lfloor \frac{x}{y} \rfloor \] is the integer part of the quotient \( \frac{x}{y} \). For example, \( \lfloor \frac{7}{2} \rfloor = 3 \) and \( \lfloor \frac{3}{3} \rfloor = 0 \). We also define the “special case” \( \lfloor \frac{x}{0} \rfloor = 0 \).

This function is primitive recursive because

\[ \lfloor \frac{x}{y} \rfloor = \min_{t \leq x} [(t + 1) \cdot y > x]. \]
Example

The remainder of the division of $x$ by $y$, $R(x, y)$: Note that $R(x, 0) = x$.

Since

$$\frac{x}{y} = \lfloor x/y \rfloor + \frac{R(x, y)}{y},$$

we can write $R(x, y) = x \div (y \cdot \lfloor x/y \rfloor)$, so $R$ is primitive recursive.
The $n^{\text{th}}$ prime number is denoted by $p_n$. For example,

$$p_0 = 0 \text{ (special case)}, \ p_1 = 2, \ p_2 = 3, \ p_3 = 5, \ldots$$

The function $p_n$ is primitive recursive.

Begin by verifying the inequality

$$p_{n+1} \leq (p_n)! + 1.$$

Note that for $0 < i \leq n$ we have

$$\frac{p_n! + 1}{p_i} = K + \frac{1}{p_i},$$

where $K$ is an integer. Therefore, $p_n! + 1$ is not divisible by any of the primes $p_1, \ldots, p_n$. So, either $p_n! + 1$ is a prime itself, or it is divisible by a prime greater than $p_n$. In either case, there is a prime $q$ such that $p_n < q \leq p_n! + 1$, which implies $p_{n+1} \leq (p_n)! + 1$. 


Example

The function $p_n$ is primitive recursive. Consider the primitive recursive function

$$h(y, z) = \min_{t \leq z} [\text{Prime}(t) \& t > y].$$

Then, we define $k(x) = h(x, x! + 1)$, which is again primitive recursive. This allows us to define $p_n$ as

$$p_0 = 0,$$

$$p_{n+1} = k(p_n),$$

so $p_n$ is primitive recursive.
Definition

Let $P(x_1, \ldots, x_n, y)$ be a predicate. The least value of $y$ for which the predicate $P(x_1, \ldots, x_n, y)$ is TRUE is denoted by $\min_y P(x_1, \ldots, x_n, y)$ if such a value exists. If there is no value for which $P(x_1, \ldots, x_n, y)$ is TRUE, then $\min_y P(x_1, \ldots, x_n, y)$ is undefined.

The unbounded minimalization defines a partial function $y = f(x_1, \ldots, x_n) = \min_y P(x_1, \ldots, x_n, y)$. 
Example

Note that

\[ x - y = \min_z [y + z = x] \]

This is a partial function that is undefined if \( x < y \).
Theorem

If $P(x_1, \ldots, x_n, y)$ is a computable predicate and if

$$f(x_1, \ldots, x_n) = \min_y P(x_1, \ldots, x_n, y),$$

then $f$ is a partially computable function.
Proof.

The following program obviously computes $f$:

\[
[A] \quad \text{IF } P(X_1, \ldots, X_n, Y) \text{ GOTO } E \\
Y \leftarrow Y + 1 \\
\text{GOTO } A
\]
Bounded minimalization begins with a primitive recursive predicate $P(t, x_1, \ldots, x_n)$ with $1 + n$ arguments and produces a primitive recursive function $f : \mathbb{N}^{1+n} \rightarrow \mathbb{N}$.

$$f(y, x_1, \ldots, x_n) = \min_{t \leq y} P(t, x_1, \ldots, x_n)$$

of $1 + n$ arguments.
In contrast, unbounded minimalization begins with a **computable predicate** $P(x_1, \ldots, x_n, y)$ with $n + 1$ arguments and produces a **computable function** $f : \mathbb{N}^n \rightarrow \mathbb{N}$

$$f(x_1, \ldots, x_n) = \min_y P(x_1, \ldots, x_n, y),$$

of $n$ arguments.