Suppose a function $h$ is defined by primitive recursion

\[
\begin{align*}
    h(\vec{x}, 0) &= f(\vec{x}) \\
    h(\vec{x}, y + 1) &= g(\vec{x}, y, h(\vec{x}, y))
\end{align*}
\]

and suppose the functions $f$ and $g$ are computable. (We use $\vec{x}$ to abbreviate $x_0, \ldots, x_{k-1}$.) Then $h(\vec{x}, 0)$ can obviously be computed, since it is just $f(\vec{x})$ which we assume is computable. $h(\vec{x}, 1)$ can then also be computed, since $1 = 0 + 1$ and so $h(\vec{x}, 1)$ is just

\[
h(\vec{x}, 1) = g(\vec{x}, 0, h(\vec{x}, 0)) = g(\vec{x}, 0, f(\vec{x})).
\]

We can go on in this way and compute

\[
\begin{align*}
    h(\vec{x}, 2) &= g(\vec{x}, 1, h(\vec{x}, 1)) = g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x}))) \\
    h(\vec{x}, 3) &= g(\vec{x}, 2, h(\vec{x}, 2)) = g(\vec{x}, 2, g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x})))) \\
    h(\vec{x}, 4) &= g(\vec{x}, 3, h(\vec{x}, 3)) = g(\vec{x}, 3, g(\vec{x}, 2, g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x})))))) \\
    &\vdots
\end{align*}
\]

Thus, to compute $h(\vec{x}, y)$ in general, successively compute $h(\vec{x}, 0), h(\vec{x}, 1), \ldots$, until we reach $h(\vec{x}, y)$.

Thus, a primitive recursive definition yields a new computable function if the functions $f$ and $g$ are computable. Composition of functions also results in a computable function if the functions $f$ and $g_i$ are computable.

Since the basic functions zero, succ, and $P^n_i$ are computable, and composition and primitive recursion yield computable functions from computable functions, this means that every primitive recursive function is computable.

Photo Credits

Bibliography