# Recursive Functionals

- A recursive functional $\Phi$ maps functions to functions, i.e. $\Phi : F(\mathbb{N}^k) \to F(\mathbb{N}^h)$ where $F(\mathbb{N}^k)$ is the set of total functions from $\mathbb{N}^k$ to $\mathbb{N}$.
- $\Phi$ is recursive if there exists a computable function $\phi : \mathbb{N}^{h+1} \to \mathbb{N}$ such that for any $f \in F(\mathbb{N}^k)$ and any $y \in \mathbb{N}^h$, $\Phi(f)(y) = y$ iff there is a finite subfunction $\theta \subseteq f$ such that $\phi(\tilde{\theta}, y) = y$.
- Intuitively, $\Phi(f)$ is computable in a finite number of steps using a finite amount of information about $f$. The finite subfunction $\theta$ provides the relevant information about $f$ needed to compute $\Phi(f)(y)$.
- If $\Phi$ is a recursive functional, then for any computable function $f$, $\Phi(f)$ is also computable. Recursive functionals preserve computability.

# Myhill-Shepherdson Theorem

- This theorem characterizes recursive functionals in terms of extensional functions on indices.
- Part 1: If $\Phi : F(\mathbb{N}^k) \to F(\mathbb{N}^h)$ is a recursive functional, then there exists a total computable extensional function $h : \mathbb{N} \to \mathbb{N}$ such that for all $e \in \mathbb{N}$, $\Phi(\phi_e) = \phi_{h(e)}$.
    - In other words, the behavior of $\Phi$ on computable functions is captured by an extensional function $h$ on indices. If $\phi_e = \phi_{e'}$, then $\Phi(\phi_e) = \Phi(\phi_{e'}) = \phi_{h(e)} = \phi_{h(e')}$.
- Part 2: If $h : \mathbb{N} \to \mathbb{N}$ is a total computable extensional function, then there is a unique recursive functional $\Phi_h$ such that for all $e \in \mathbb{N}$, $\Phi_h(\phi_e) = \phi_{h(e)}$.
    - An extensional function $h$ that transforms indices induces a recursive functional $\Phi_h$ on the actual computable functions. $\Phi_h$ is uniquely determined by $h$.

# First Recursion Theorem

- This theorem states that any recursive functional $\Phi : F(\mathbb{N}^k) \to F(\mathbb{N}^k)$ has a least computable fixed point $f_\Phi$.
- A function $f$ is a fixed point of $\Phi$ if $\Phi(f) = f$. The least fixed point $f_\Phi$ satisfies:
    1. $\Phi(f_\Phi) = f_\Phi$
    2. For any $g \in F(\mathbb{N}^k)$, if $\Phi(g) = g$ then $f_\Phi \subseteq g$
    3. $f_\Phi$ is computable
- This allows defining computable functions by very general recursive schemes. As long as each recursive definition corresponds to a recursive functional $\Phi$, the theorem guarantees the existence of a computable function satisfying the recursive equations.

Regarding the fixed point part:

The First Recursion Theorem deals with recursive functionals of the form $\Phi : F(\mathbb{N}^k) \to F(\mathbb{N}^k)$ , i.e., functionals that map functions of $k$ arguments to functions of $k$ arguments.

A function $f$ is a fixed point of $\Phi$ if applying $\Phi$ to $f$ yields $f$ itself, i.e., $\Phi(f) = f$. In other words, $f$ is a solution to the recursive equation defined by $\Phi$.

The theorem states that every such recursive functional has a least fixed point $f_\Phi$ which is computable. "Least" here means that for any other fixed point $g$ of $\Phi$, $f_\Phi$ is a subfunction of $g$ (denoted $f_\Phi \subseteq g$). This implies that $f_\Phi$ is in some sense the "smallest" or "minimal" solution to the recursive equation defined by $\Phi$.

The significance of this theorem is that it allows defining computable functions recursively using very general schemes. As long as the recursive definition can be expressed as a recursive functional $\Phi$, the theorem guarantees that there is a computable function $f_\Phi$ that satisfies this definition. Moreover, this function is the least solution in the sense described above.

This provides a powerful tool for defining and reasoning about computable functions. It shows that recursive definitions, as long as they correspond to recursive functionals, always have a well-defined computable solution. This result is fundamental in the theory of computability and in the study of recursive function definitions.