Introduction and summary. When dealing with denotational semantics of programming languages, the natural question one may ask is whether the interpretation of a program is an effective object (function) in the given semantical domain. An answer to this question first requires a sound notion of computability in abstract structures, for computability of number theoretic functions may not suffice, i.e. not any program may be interpreted just as a (partial) recursive function. As a matter of fact, mathematical domains for the semantics of type-free and typed languages need to be much richer than $\omega$ (the natural numbers): $\omega$, say, does not yield a model for type-free $\lambda$-calculus (see LM [1984] for a discussion) and, by definition, the semantics of typed languages, such as typed $\lambda$-calculus, whose programs may act also as programs, immediately involve function spaces at any finite higher type. As a further motivation, just notice that Computer Science often deals with different and various sorts of data, besides $\omega$.

In the early 50's, Myhill and Shepherdson, MS [1955], gave an elegant characterization of type two functionals, the recursive operators, by a simple notion of application over $P\omega$. Shortly later, papers by Gödel, Kleene and Kreisel (Göd [1958], Kle [1959], Kre [1959], for example) introduced higher type recursion theory, i.e. recursion on the type structure generated by $\omega$. Gödel and Kreisel work was also motivated by consistency results for intuitionistic arithmetic and constructive mathematics.

Some 10-15 years ago, Scott's work on posets (lattices, in particular) provided the category theoretic framework for computability in abstract structures, by a suitable topological notion of approximation. Later on, Ershov (Er [1972,1976] and a lot more in Algebra and Logic
and ZML) and Hyland (Hy [1979]) studied the effective type structures over \( \omega \) of partial and total objects (respectively) as subcategories of topological or limit categories, relating by this higher type recursion theory to computability in abstract structures.

Scott's domains, Scott [1981], and Ershov's complete \( f_\omega \)-spaces are readily seen to be equivalent (see GL [1982] for a discussion and LM [1983] for recent recursion theoretic applications). Similarly to Hyland's approach, an element (function) of a domain is computable when it is the limit of a countable sequence with an r.e. set of indices. In particular the ideal of compact elements below (see later) must be indexed by an r.e. set.

For increasing types, though, the intuition of the "ideal below", say, gets more and more vague. The purpose of this paper is to take back to Myhill-Shepherdson <P\( \omega \),> as much as possible of the abstract (higher types) approach.

More precisely, for \( A, B \subseteq P\omega \), set \( A \rightarrow B = \{ d \in P\omega / \forall a \in A \; da \in B \} \). We first show that any effectively given domain can be embedded into \( P\omega \) by a continuous and computable retraction (notation: \( X \subseteq_c A_X \) for some \( A_X \subseteq P\omega \), which is also an effectively given domain). Then, if \( X \subseteq_c A_X \) and \( Y \subseteq_c A_Y \), one has

\[
(1) \quad \text{Cont}(X,Y) \subseteq_c A_X \rightarrow A_Y \quad \text{and} \quad X \times Y \subseteq_c A_X \times A_Y
\]

(for some simple product in \( P\omega \)). Also \( A_X \rightarrow A_Y \) and \( A_X \times A_Y \) are effectively given domains.

Thus an (effective) functional in a given type, over an arbitrary domain, is represented by the application "." and an (r.e.) set in the corresponding type as a subset of \( P\omega \).

In particular, let \( P \subseteq P\omega \) be the single valued sets, i.e. \( P \) is isomorphic to the effectively given domain of the partial functions on \( \omega \). Then, for \( P(1) = P, P(n+1) = P(n) \rightarrow P(n) \) extend the classical recursive operators at higher types (this is done for any finite type \( \sigma \in T \)).

By (1), Ershov's model of the Kleene-Xreisel countable functionals can be effectively embedded, by some \( g_\sigma \)'s, into the type structure \( \{ P^\sigma \}_{\sigma \in T} \) in \( P\omega \). Thus the recursive functionals correspond to the r.e. sets in the due types, e.g. \( f \) has type \( \sigma \rightarrow \tau \) iff \( g_{\sigma \rightarrow \tau}(f) \) is an r.e. set in \( P^\sigma \rightarrow P^\tau \).

§.1. Domains and \( P\omega \)

For the notion of domain we refer to Scott [1931]. Shortly, in a poset \( (X, \leq) \), set \( \bar{x} = \{ y \in X/ x \leq y \} \); then a domain is an algebraic