

Domain Theory Notes*

Gregory M. Malecha

January 28, 2008

We continued our discussion of Domain Theory focusing on section 2 of the notes *Operations on Data*. Our focus was computable operations, two incarnations of this are approximable mappings and continuous functions. The theme for the class, once again, is the importance to the relationship between incremental computation and domains and the constraints of consistency and monotonicity.

Computable operations are maps between domains and satisfy two properties:

- the image, $\{y : \exists x.xFy\}$ is an ideal, and
- F is monotonic.

As was previously mentioned, the monotonicity constraint is important because we don't want computations that "take back" previously generated answers.

An *approximable mapping* is a computable operation which maps \perp to \perp and is closed so that aFb and aFb' implies $aF(b \sqcup b')$. In addition to these constraints, that aFb and $b \sqsubseteq b'$ implies aFb' and if aFb and $a \sqsubseteq a'$ then $a'Fb$. The subtle difference is important to note as it is similar to the rules for functional sub-typing.

We can construct our approximable mapping by using one-step functions which are functions that are bottom everywhere below some point and a constant everywhere above that point. Figure 1 shows abstract illustrations of these functions and their behavior when they interact. In Figure 1c, \mathbf{c} denotes $\sqcup a, a'$ and \mathbf{d} denotes $\sqcup b, b'$ thus adhering to the property that $a \wedge a' \mapsto b \wedge b'$. Because of their use in building finitary bases, one-step functions are referred to a pre-basis.

While approximate mappings are technically good enough, they don't provide a very elegant solution. Mainly, we want a stronger condition than consistency, namely we want sequentially. The example of parallel or was given.

$$\begin{array}{l} \perp \vee \text{true} \mapsto \text{true} \quad \text{false} \vee \text{false} \mapsto \text{false} \\ \text{true} \vee \perp \mapsto \text{true} \quad \text{otherwise} \mapsto \perp \end{array}$$

In addition to parallel or being very difficult to implement, an existential search operation is also required to make this formulation go through. Cartwright and Felleisen addressed these same issues by using exceptions and `callcc` which are possibly more familiar to computer scientists.

*These notes are based on the lecture which is based on Robert Cartwright and Rebecca Parson's notes on domain theory, *Domain Theory: An Introduction*

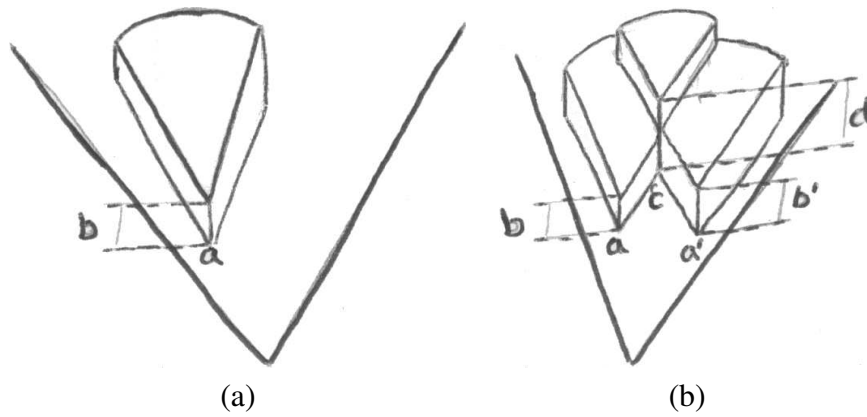


Figure 1: One-step functions and their interactions. (a) The wedge represents the domain. The one-step function takes all things which a approximates and assigns value b . (b) When two step functions overlap, the values above $c = \sqcup a, a'$ is the least upper-bound (denoted by $d = \sqcup b, b'$)

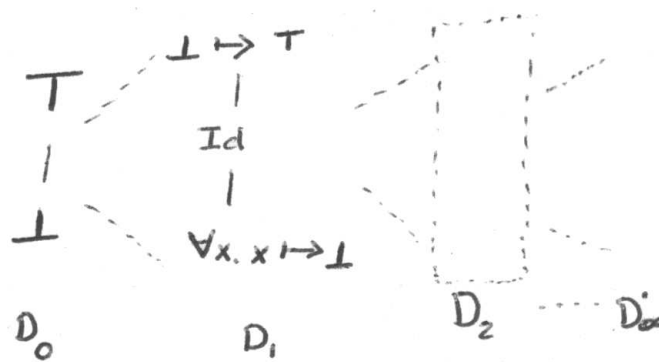


Figure 2: The beginning of the D^∞ construction of the domain of the λ calculus.

We concluded with a definition of universal domains. A *universal domains* is a domain in which any Scott domain can be embedded. This lead to an abbreviated discussion of Scott's D^∞ construction (Figure 2) which is less elegant than the $P\omega$ model since it doesn't work in the frame of constructive mathematics by providing a witness.