As a formal theory, the \( \lambda \)-Calculus has equational rules for the explicit definition of functions in a type-free setting. The rules can be sketched as follows:

- **\( \alpha \)-conversion**
  \[ \lambda x . [ \ldots X \ldots ] = \lambda y . [ \ldots Y \ldots ] \]

- **\( \beta \)-conversion**
  \[ (\lambda x . [ \ldots X \ldots ]) (\tau) = [ \ldots T \ldots ] \]

- **\( \eta \)-conversion**
  \[ \lambda x . P(x) = P \]

Background and historical references can be found in the recent survey Cardone-Hindley [2009] cited below. The Graph Model for this theory satisfies the first two equations and modifies the third. The idea incorporates the notion of enumeration operators from Recursive Function Theory. (See the additional references cited.) The construction of the model proceeds as follows, where the elements of the model are just sets in the powerset of the integers, \( \mathcal{P} \mathbb{N} \).

Define a pairing function on the integers by: \( \langle n, m \rangle = 2^n \cdot (2 \cdot m + 1) \). This makes every positive integer uniquely the “Gödel number” of a pair of integers. Define numbering of finite sequences of integers by: \( \langle \rangle = 0 \) and \( \langle n_0, n_1, \ldots, n_{k-1}, n_k \rangle = \langle \langle n_0, n_1, \ldots, n_{k-1} \rangle, n_k \rangle \). That definition makes every integer uniquely the number of a finite sequence of integers. Further, define a numbering of finite sets of integers by:

\[ \text{set} \left( \{ n_0, n_1, \ldots, n_k \} \right) = \{ n_0, n_1, \ldots, n_k \}. \]  
And for \( X \in \mathcal{P} \mathbb{N} \), define \( X^* = \{ n \in \mathbb{N} \mid \text{set} \{ n \} \subseteq X \} \).

The structure of the model is given by defining:

**Application**
\[ P(X) = \{ m \mid \exists n \in X^*, \langle n, m \rangle \in P \} \]

**Abstraction**
\[ \lambda X . [ \ldots X \ldots ] = \{ \langle n, m \rangle \mid m \in [ \ldots \text{set} \{ n \} \ldots ] \} \]

\( \lambda \)-Abstraction applies to any context defining a function \( \Phi : X \rightarrow \ldots \) which is continuous as a function \( \Phi : \mathcal{P} \mathbb{N} \rightarrow \mathcal{P} \mathbb{N} \) in the topology on \( \mathcal{P} \mathbb{N} \) with the sets \( \omega_X = \{ X \subseteq \mathbb{N} \mid n \in X^* \} \) as a basis for the topology. The model satisfies the first two conversion rules and shows that any continuous function can be given by application with a suitable subset \( P \subseteq \mathcal{P} \mathbb{N} \). The third conversion rule has to be weakened to \( P \subseteq \lambda X . P(X) \). The continuous functions are the general enumeration operators, and the computable ones are those given by \( \mathcal{P} \mathbb{N} \) sets \( P \). There are many nice properties relating this \( \lambda \)-calculus structure to the lattice structure of \( \mathcal{P} \mathbb{N} \). A rich type theory can be added to \( \mathcal{P} \mathbb{N} \) by using partial equivalence relations (PERs) on \( \mathcal{P} \mathbb{N} \) as types. There is no room to give the definitions in this abstract, however.

**Probability** can now be added to the model simply by using random variables. For example, consider Borel functions \( X : [0, 1] \rightarrow \mathcal{P} \mathbb{N} \). These form a \( \lambda \)-calculus structure in the same way that real-valued random variables form a linear vector space. By using the standard measure on \( [0, 1] \), probabilities can be assigned to logical formulas involving \( \lambda \)-calculus equations. Inasmuch as \( \lambda \)-calculus is a programming language, random variables can be introduced as parameters in defining randomized algorithms. This gives us a denotational semantics for a stochastic \( \lambda \)-calculus.

There are many approaches to modeling \( \lambda \)-calculus, and expositions and historical references can be found in Cardone-Hindley [2009]. In 1972 Plotkin wrote an AI report at the University of Edinburgh entitled “A set-theoretical definition of application” which remained unpublished until it was incorporated into the more extensive paper Plotkin [1993]. Scott developed his model based on the powerset of the integers subsequently, but he only later realized it was basically the same as Plotkin’s model. See Scott [1976] for further details where he called the idea the Graph Model.


Much earlier, enumeration reducibility was introduced by Rogers in lecture notes and mentioned by Friedberg-Rogers [1959] as a way of defining a positive reducibility between sets. Enumeration degrees are discussed at length in Rogers [1967], and there is now a vast literature on the subject. Enumeration operators are also studied in Rogers [1967] as well. Earlier, Myhill-Shepherdson [1955] defined functionals on partial functions with similar properties. Neither of those teams saw that their operators possessed an algebra that would model \( \lambda \)-calculus, however.


Some historical remarks on the notion of PER as an interpretation of types are given by Bruce et al. [1990], where we learn that they were introduced by Myhill and Shepherdson [1955] for types of first-order functions, and then extended to simple types by Kreisel [1959]. Scott took the use of partial equivalence relations from the work of Kreisel and collaborators.


Two papers about introducing random features in \(\lambda\)-calculus are Deliguoro-Piperno [1995] and Dal Lagoa-Zorzia [2012]. Both of those articles have many background references.
