Semantics for CPython Virtual Machine

July 5, 2020

CPython VM provides very high level operations.

Python has 4 kinds of variables.

The scope of Python is divided by function boundaries. No other Python language constructs create/destroy a scope.

Any name in a scope may denote 4 kinds of variables.

1. **local-only variable** A bound variable, but used only in the scope
2. **cell variable** A bound variable, the its definition scope has nested functions referencing this variable. This variable is in form of *cell* data structure, in case of future mutations.
3. **free variable** A cell variable comes to a nested function referencing it, and becomes a free variable. It's also a *cell*
4. **global variable** Defined in the top-level of a Python module

Particularly, **global variables** are stored in a special hash table, which could be accessed by **globals**(), and is separately maintained per Python module. Users can modify the global variables outside its defined module, by modifying that special hash table. Other variables are stored in form of variable *slots*, where the indices of them are statically decided by the bytecode compiler.

1. **LOAD** load a variable to the top of stack
2. **LOAD** load

$$PYC \vdash PY \tag{1}$$

Bibliography