A Theory of Name Resolution

Descreve uma teoria independente de linguagem para name binding and resolution (Resumidamente é a teoria que servirá de pilar para a minha tese uma vez que a metodologia por eles usada vai servir como base para a minha implementação).

Eles dividem o problema em 2 fases:

- Construção de um grafo de scope usando as regras da linguagem de uma árvore de sintaxe
- Depois são feitas as verificações do grafo usando um processo de resolução independente de linguagem

Esta abordagem colmata problemas da maior parte das linguagens que usam processos repetitivos e por vezes ambíguos para as suas resoluções de nomes (manipulation of a symbol table in a compiler, a use-to-definition display in an IDE, or a substitution function in a mechanized soundness proof).

SCOPE GRAPH

Node :

- Name references
- Declarations
- Scopes

Edges :

- References to scopes
- Declarations to scopes
- Scopes to “parent” scopes

Capacidades da teoria :

- O cálculo de resolução especifica como construir um caminho através do gráfico a partir de uma referência a uma declaração, o que corresponde a uma possível resolução da referência. Referências ambíguas correspondem naturalmente a vários caminhos de resolução a partir do mesmo nó de referência; referências não resolvidas correspondem à ausência de caminhos de resolução.
- A teoria desenvolvida suporta também “imports”.

  “The calculus supports complex import patterns including transitive and cyclic import of scopes.“

- Para qualquer linguagem, a construção pode ser especificada por uma definição convencional dirigida por sintaxe sobre a gramática da linguagem.
- Um algoritmo prático para calcular ambientes estáticos convencionais que mapeiam identificadores de limite para os locais AST das

declarações correspondentes, que podem ser usados para implementar uma função de resolução determinística e de terminação que é consistente com o cálculo.

O paper divide-se em 6 capítulos que se complementam para desenvolver e provar a teoria em questão:

- **Scope Graph and Resolution Calculus**: We introduce a language-independent framework to capture the relations among references, declarations, scopes, and imports in a program. We give a declarative specification of the resolution of references to declarations by means of a calculus that defines resolution paths in a scope graph .
- **Variants**: We illustrate the modularity of our core framework design by decribing several variants that support more complex binding schemes.
- **Coverage**: We show that the framework covers interesting name binding patterns in existing languages, including various flavors of let bindings, qualified names, and inheritance in Java.
- **Scope graph construction**: We show how scope graphs can be constructed for arbitrary programs in a simple example language via straightforward syntax-directed traversal.
- **Resolution algorithm**: We define a deterministic and terminating resolution algorithm based on the construction of binding environments, and prove that it is sound and complete with respect to the calculus.
- **α-equivalence and renaming**: We define a language-independent characteriation of α-equivalence of programs, and use it to define a notion of valid renaming.