zbMATH Open
THE FIRST RESOURCE FOR MATHEMATICS

Li, Bixin; Wang, Yunfeng; Li, Xuandong; Zheng, Guoliang

An approach for constructing VFCG in statically-typed object-oriented language. (Chinese. English summary) Zbl 0964.68508

J. Nanjing Univ., Nat. Sci. 36, No. 2, 163-170 (2000).

Summary: A major advantage of object-oriented languages is abstraction. The most important language feature that supports abstraction is the dynamic dispatch of methods based on the run-time type of an object. In dynamically typed language like Smalltalk and SELF, all dispatches are considered dynamic. In statically typed language, such as C++, virtual function call is a both important and complicated problem, and programmers must explicitly request dynamic dispatch by declaring a method to be virtual. Virtual functions make code easier for programmers to reuse. But virtual function call makes it harder to analyze or understand C++ programs. We think there are at least 3 reasons: first, C++ programs suffer less of an initial performance penalty, at the cost of reduced flexibility and increased programmer effort in the presence of virtual function call. Second, if every potential target of a virtual function call is included in the call graph, the user will face with a vastly larger space of object types and functions that must be comprehended to understand the meaning of the program as a whole. Third, because of the existence of inheritance relationship (especially multiple inheritance relationship), all classes in a program form a complicated class hierarchy structure.

We represent such structure with Class-Hierarchy-Graph (CHG). Methods defined in some class can be either visible or invisible in its corresponding class node in CHG. We use Visible-Method-Class-Hierarchy-Graph (VM-CHG) to represent such CHG. Some child classes inherit their parent class' method; others override their parent class' method. We use inherit-set to represent the set of classes that inherit their parent class' method, and override-set to represent the set of classes that override their parent class' method. If some class has many base (or parent) classes and these base classes have only one super base class, it is hard to determine the inherited method by the derived class belongs to which base class or which version of super base class. In this paper, we present an approach for constructing VFCG in statically-typed object-oriented language so as to understand virtual function call easily. First, VM-CHG is constructed to represent class hierarchy relationship between objects where method defined or inherited is visible. Then, inherit-set (which represents the union of the set of classes that inherit a particular class' method m and the particular class that defines the method m) and override-set (representing the set of classes that override a particular class' method) are computed, and override-frontier is also determined by having computed inherit-set and override-set. Finally, VFCG is constructed with VM-CHG and override-frontier. On the other hand, we present all algorithms for constructing VM-CHG, computing inherit-set and override-set, determining override-frontier, and creating VFCG. Meantime, we discuss the importance of VFCG in resolving static object-oriented virtual function call.

MSC:

68N15 Theory of programming languages

Keywords:

class-hierarchy-graph; virtual function call