CS 617 Object Oriented Systems Lecture 12 Implementations of Dynamic Dispatch 3:30-5:00 pm, Thu Feb 14

Rushikesh K Joshi

Department of Computer Science and Engineering
Indian Institute of Technology Bombay

Outline

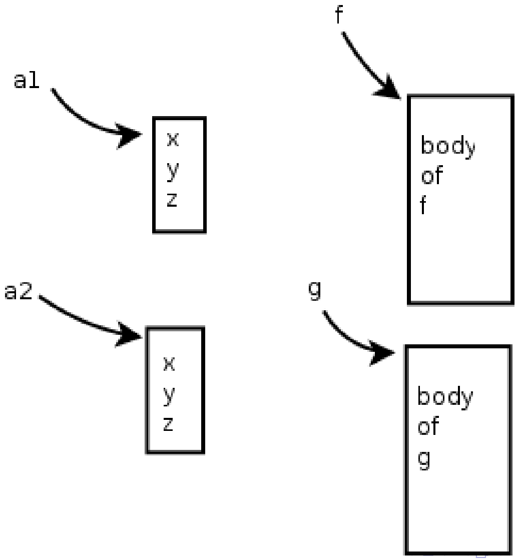
Outline

Method sharing

```
class A {
int x; int y;
some z; public:
        void f(int a) ...;
        void g(int b) ...;
};

main () {
A *a1 = new A();
A *a2 = new A();
        a1->f(10);
        a2->f(20);
}
```

A runtime view



Summary

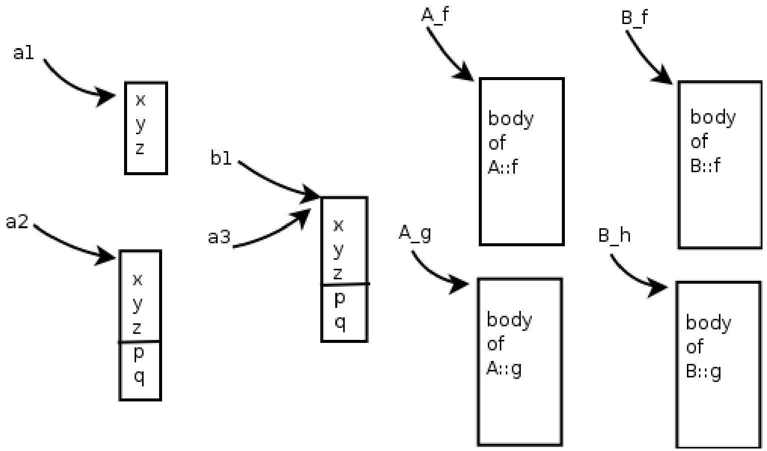
- instance variables: per object
- method bodies shared
- relative addressing
- use of ’this’ or ’self’

Outline

Single Inheritance, Single Subclass

```
class A {
int x, y; some z;
public:        void f(int a) ...;
               void g(int b) ...;
};
class B : public A {
int p,q;
public:        void f(int a) {...}
               void h(int c) {...}
}
main () {
A *a1 = new A();
A *a2 = new B();
B *b1 = new B();
A *a3 = b1;
... invoke f,g,h as permissible on the instances ..
}
```

A runtime view



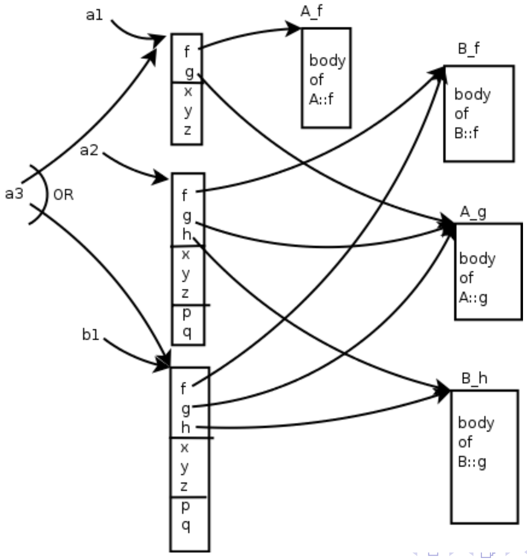
Summary

- Instance includes sub-objects corresponding to parents
- Method sharing as before
- Memory allocation scheme for sub-objects: methods should be able to find the addresses of instance variables accessible to them

Dynamic Binding

```
main () {
...
A *a3;
...
if C1, a3 = a1;
          else a3 = b1;
... invoke f,g on a ..
}
```

Accounting for Dynamic Binding

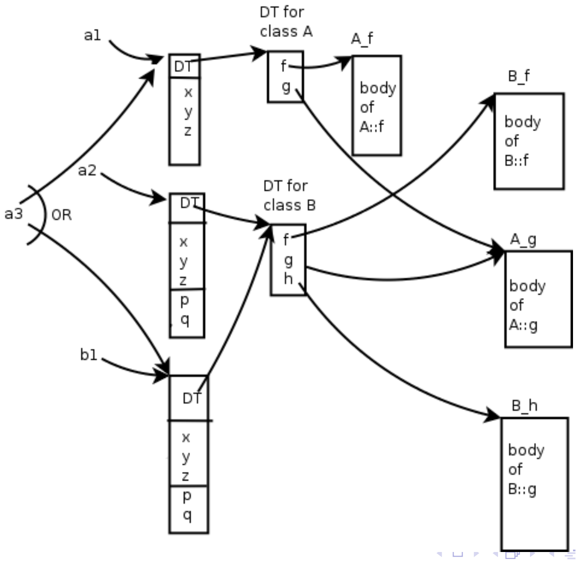


Locating Instance Variables?

Will shared function bodies be able to locate their respective instance variables?

What should pass on as ’this’?

Dispatch Tables, and Sharing Them



Translating Assignments and Invocations

```
A *a1 = new A();
B* b1 = new B();
A *a3; ...
if C1, a3 = a1; else a3 = b1;
        a->f(val1);
        a->g(val2);
```

The Scheme of Implementation:

```
A *a1 = allocate_A()
a1->DT=A's DT
B* b1 = allocate_B()
b1->DT=B's DT
A *a3; ...
if C1, a3 = a1; else a3 = b1;
        a3->(DT[0])(a3,val1);
        a3->(DT[1])(a3,val2);
```

Single Inheritance, Multiple Subclasses I

```
class A {
int x, y; some z;
public:        void f(int a) ...;
               void g(int b) ...;
};
class B : public A {
int p,q;
public:        void f(int a) {...}
               void h(int c) {...}
}
class C : public A {
int r,s;
public:        void f(int a) {...}
               void g(int c) {...}
}
```

Single Inheritance, Multiple Subclasses II

```
main () {
A *a;
...
if C1, a = new A();
                else if C2 a = new B();
                else a = new C();
... invoke f,g on a ..
}
```

Multiple Inheritance

Will MI pose new problems?