

Programmanalysen und Compilerbau

Sommersemester 2005
Prof. E. Plödereder, E. Wiebe

5. Übungsblatt

Die Übung findet am 23.06 statt.

Aufgabe 5.1

Gegeben ist das folgende Programmfragment.

```
function Create return Stack is
    S : Stack := Alloc_Stack(100);
begin
    Init(S);
    return S;
end Create;

function P return Stack is
    S : Stack;
begin
    if Random() then
        return Create();
    else
        S := P();
        Push(S, ...);
        return S;
    end if;
end P;

procedure Swap(Stack S1, Stack S2) is
    I1, I2 : Item;
begin
    if not Is_Empty(S1) and then not Is_Empty(S2) then
        I1 := Pop(S1);
        I2 := Pop(S2);
        Push(S1, I2); Push(S2, I1);
    end if;
end Swap;

procedure Main is
    MS,S : Stack;
    FP   : Function_Pointer;
begin
    MS := Alloc_Stack(100);
    Init(MS);
    if Random() then
        FP := P'Access;
    else
        FP := Create'Access;
    end if;
    MS := FP();
    Swap(MS, S);
    MS := S;
    Swap(MS, S);
end Main;
```

Hinweis: Für den Zeiger FP hat eine Zeigeranalyse die möglichen Ziele P und Create festgestellt.

- a. Erstellen Sie den Aufrufgraphen für das Beispielprogramm.
- b. Wie würde der Aufrufgraph aussehen, wenn keine genauen Informationen über FP zur Verfügung stehen würden?

Aufgabe 5.2

- a. Bestimmen Sie zu dem Programm aus Aufgabe 4.2 alle Lebensbereiche der vorkommenden Variablen.
- b. Verwenden Sie die Lebensbereiche als Kandidaten für eine Registervergabe. Zur Verfügung stehen drei Register. Falls bei der Registervergabe Lebensbereiche von der Vergabe ausgeschlossen werden müssen, wählen Sie die Kandidaten, die in der innersten Schleife des Flußgraphen die wenigsten Load- und Store-Instruktionen brauchen.

Aufgabe 5.3

- a. Fügen Sie in den Flußgraphen aus Aufgabe 4.2 die ϕ -Knoten ein, die für eine SSA-Form benötigt werden.
- b. Bestimmen Sie den zugehörigen Präferenzgraphen.