Technische Universität Wien
Programmiersprachen und Übersetzerbau
Prof. Dr. J. Knoop

A-1040 Wien
Argentinierstr. 8
Tel.: 01-58801-18510

"Optimierende Compiler (185.A04, VU 2.0, ECTS 3.0)" WS 2020/21

Übungsblatt 5 / Prakt. Aufgabe 1 17.11.2020

---

## Intraprozedurale Dominator-Analyse mit PAG

Eine Dominator-Analyse ermittelt die Dominanzrelation zwischen Knoten in einem gerichteten Graphen. Diese Relation ist in einem gerichteten Graphen mit einem eindeutigen vorgängerlosen Startknoten $s_0$ wie folgt definiert: Ein Knoten $d$ dominiert einen Knoten $n$ genau dann, wenn jeder Pfad von $s_0$ nach $n$ durch $d$ verläuft. Trivialerweise dominiert sich jeder Knoten selbst.

In dieser Aufgabe soll eine Dominator-Analyse auf dem Kontrollflussgraphen (engl. control flow graph (CFG)) von Programmen definiert werden. Die Analyse ermittelt also, welche Anweisungen bestimmt vor welchen anderen ausgeführt werden müssen. Im folgenden ein Beispiel für intraprozedurale Dominator-Analyse für die Sprache While:

```
[a := 1]1;
[b := a]2;
while [a < 10]3 do (
    if [a < b]4 then
      [a := a + 1]5;
else
      [b := b + 1]6;
    [c := a + b]7;
)
```

| $\ell$ | $DA_\circ(\ell)$ | $DA_\bullet(\ell)$ |
|---|---|---|
| 1 | {} | {1} |
| 2 | {1} | {1, 2} |
| 3 | {1, 2} | {1, 2, 3} |
| 4 | {1, 2, 3} | {1, 2, 3, 4} |
| 5 | {1, 2, 3, 4} | {1, 2, 3, 4, 5} |
| 6 | {1, 2, 3, 4} | {1, 2, 3, 4, 6} |
| 7 | {1, 2, 3, 4} | {1, 2, 3, 4, 7} |

**Aufgabe 1** : (4+4 Punkte)

Spezifizieren Sie die Dominator-Analyse für While:

(a) Definieren Sie $kill_{DA}(\ell)$ und $gen_{DA}(\ell)$.

(b) Definieren Sie die Gleichungen für $DA_\circ(\ell)$, $DA_\bullet(\ell) : Lab_\star \to \mathcal{P}(Lab_\star)$.

**Aufgabe 2** : (20 Punkte)

### Implementierung mit PAG

Spezifizieren Sie mit PAG eine Dominator-Analyse für die Sprache $SL1$. Diese Sprache ist jene Teilmenge von C++, die der Sprache While ohne Funktionsaufrufe entspricht. Gültige Anweisungen mit Entsprechung in While sind somit Zuweisungen, `if`-Verzweigungen und `while`-Schleifen. Es sind lediglich einfache Ausdrücke aus den üblichen Rechen- und Vergleichsoperatoren erlaubt.

Als Unterschied gegenüber While muss jede Variable vor ihrer Verwendung deklariert werden, dabei sind die Typen `int` und `bool` zulässig. Es existiert eine einzige Funktion mit der Signatur `int main()`, deren letzte Anweisung eine `return`-Anweisung mit einem Wert vom Typ `int` ist. Das folgende Programm in $SL1$ (und somit C++) entspricht dem obigen While -Beispiel:

```
int main() {
    int a, b, c;
    a = 1;
    b = a;
    while (a < 10) {
        if (a < b)
            a = a + 1;
        else
           b = b + 1;
        c = a + b;
    }
    return 0;
}
```

Ihre PAG-Analyse soll den Namen `da` tragen. Zur Ermittlung einer eindeutigen Kennung für Anweisungen stellt SATIrE das Attribut `label` bereit. Sie können dieses innerhalb von Transferfunktionen einfach wie eine vordefnierte Variable vom Typ `snum` verwenden.

Ihre Analyse soll die gesamte Sprache $SL1$ abdecken. Das Verhalten der Analyse auf Programmen, die nicht in $SL1$ liegen, ist Ihnen freigestellt. Sie müssen also keine Fehlerbehandlung für sonstige Sprachkonstrukte implementieren.

---

**Abgabe**: **Dienstag, den 24.11.2020**, per e-mail an: `hans@complang.tuwien.ac.at` (Dr. Hans Moritsch). Geben Sie bitte als Betreffzeile 'OU: Prakt. Aufgabe 1, Nachname(n)' an und hängen Sie die Antworten auf die beiden Textfragen aus Aufgabe 1 als PDF-Datei, die Analysespezifikation für Aufgabe 2 als .optla-Datei an.