

Aufgabe: Dominator-Analyse

Optimierende Übersetzer WS 2017/2018

Abgabetermin: 28.11.2017, 15:00 Uhr

1 Intraprozedurale Dominator-Analyse

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 (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:

| | | | |
|---|--------|--------------------|----------------------|
| <code>[a := 1]¹;</code> | ℓ | $DA_{\circ}(\ell)$ | $DA_{\bullet}(\ell)$ |
| <code>[b := a]²;</code> | 1 | {} | {1} |
| <code>while [a < 10]³ do (</code> | 2 | {1} | {1, 2} |
| <code>if [a < b]⁴ then</code> | 3 | {1, 2} | {1, 2, 3} |
| <code>[a := a + 1]⁵;</code> | 4 | {1, 2, 3} | {1, 2, 3, 4} |
| <code>else</code> | 5 | {1, 2, 3, 4} | {1, 2, 3, 4, 5} |
| <code>[b := b + 1]⁶;</code> | 6 | {1, 2, 3, 4} | {1, 2, 3, 4, 6} |
| <code>[c := a + b]⁷;</code> | 7 | {1, 2, 3, 4} | {1, 2, 3, 4, 7} |
| <code>)</code> | | | |

Spezifizieren Sie die Dominator-Analyse für WHILE:

- Definieren Sie $kill_{DA}(\ell)$ und $gen_{DA}(\ell)$.
- Definieren Sie die Gleichungen für $DA_{\circ}(\ell), DA_{\bullet}(\ell): Lab_{\star} \rightarrow \mathcal{P}(Lab_{\star})$.

2 Implementierung mit PAG

Spezifizieren Sie mit PAG eine Dominator-Analyse für die Sprache SL_1 . 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 SL_1 (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 vordefinierte Variable vom Typ `snum` verwenden.

Ihre Analyse soll die gesamte Sprache SL_1 abdecken. Das Verhalten der Analyse auf Programmen, die nicht in SL_1 liegen, ist Ihnen freigestellt. Sie müssen also keine Fehlerbehandlung für sonstige Sprachkonstrukte implementieren.

3 Abgabe

Senden Sie Ihre Lösungen per E-Mail an `hans@complang.tuwien.ac.at`.

Geben Sie bitte als Betreffzeile ‘OU: Aufgabe 1, *Nachname(n)*’ an und hängen Sie die Antworten auf die Textfragen 1 (a) und 1 (b) als PDF-Datei, die Analysespezifikation für Teilaufgabe 2 als `.opt1a`-Datei an.