Universität Karlsruhe (TH)

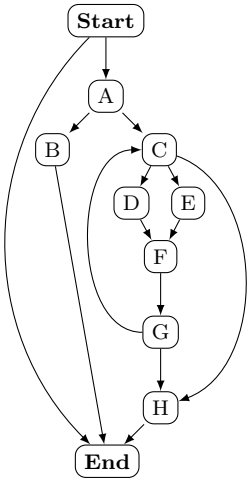
Lehrstuhl für Programmierparadigmen

Sprachtechnologie und Compiler II SS 2009 http://pp.info.uni-karlsruhe.de/
Dozent: Prof. Dr.-Ing. G. Snelting snelting@ipd.info.uni-karlsruhe.de
Übungsleiter: Matthias Braun braun@ipd.info.uni-karlsruhe.de

Lösung zu Übungsblatt 3 | Ausgabe: 20.5.2009 | Besprechung: 26.5.2009

Aufgabe 1: *Dominanz - (Lengauer-Tarjan-Algorithmus)*

Gegeben sei der aus der Vorlesung bekannte Ablaufgraph.



a) Berechnen Sie den Dominatorbaum mittels Fixpunktiteration (Folien Dominanz S. 16)

Lösung:

Berechne Dominatormengen mit rekursiver Formel:

$$\mathrm{dom}(X) = \{X\} \cup \bigcap_{Y \in \mathrm{pred}(X)} \mathrm{dom}(Y)$$

$dom(\mathbf{Start})$	=	$\{\mathbf{Start}, A, B, C, D, E, F, G, H, \mathbf{End}\}$
$dom(A)$	=	$\{A, B, C, D, E, F, G, H\}$
$dom(B)$	=	$\{B\}$
$dom(C)$	=	$\{C, D, E, F, G, H\}$
$dom(D)$	=	$\{D\}$
$dom(E)$	=	$\{E\}$
$dom(F)$	=	$\{F, G\}$
$dom(G)$	=	$\{G\}$
$dom(H)$	=	$\{H\}$
$dom(\mathbf{End})$	=	$\{\mathbf{End}\}$

Rechnet man die transitiven Elemente aus den Mengen heraus, ergeben sich die direkten Dominatoren, die dem Dominanzbaum am Ende entsprechen.

b) Berechnen Sie den Dominatorbaum mit Hilfe des Lengauer-Tarjan-Algorithmusses (Folien Dominanz S. 17ff). Geben Sie die berechneten Semidominatoren und vorläufigen Dominatoren an.

Lösung:

Berechne Tiefensuchbaum. In dieser Lösung wurden die Knoten in folgender Reihenfolge besucht:

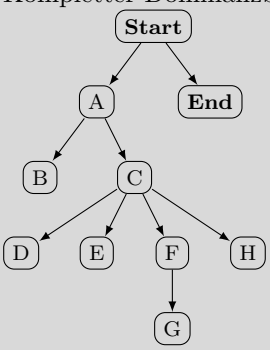
Knoten	Tiefensuchnummer
Start	0
End	1
A	2
B	3
C	4
D	5
F	6
G	7
H	8
E	9

Im nächsten Schritt werden Semidominatoren und vorläufige Dominatoren berechnet.

$$
\begin{aligned}
sdom(E) &= 4 \\
sdom(H) &= 4 \\
sdom(G) &= 6 \\
sdom(F) &= 4 \\
sdom(\textbf{End}) &= 0 \\
sdom(A) &= 0 \\
sdom(B) &= 2 \\
sdom(C) &= 2 \\
sdom(D) &= 4 \\
sdom(\textbf{Start}) &= 0
\end{aligned}
$$

In diesem Fall wurden keine vorläufigen Dominatoren berechnet. Nach der letzten Phase ergeben sich die direkten Dominatoren und damit der komplette Dominanzbaum (s. unten).

Lösung:
Kompletter Dominanzbaum:



Aufgabe 2: *Datenflussanalyse*

In der Vorlesung wurden die folgenden Datenflussanalysen vorgestellt:

- Available Expression Analysis
- Reaching Definitions Analysis
- Very Busy Expressions Analysis
- Live Variables Analysis

2.1 *Wiederholung*

Wie sehen die allgemeinen Datenflussgleichungen für die Verfahren aus?

Um welche Art der Analyse handelt es sich (vorwärts oder rückwärts; sind wir an der größten oder an der kleinsten Lösung interessiert)?

Lösung:

- Available Expression Analysis - vorwärts, größte Lösung
- Reaching Definitions Analysis - vorwärts, kleinste Lösung
- Very Busy Expressions Analysis - rückwärts, grösste Lösung
- Live Variables Analysis - rückwärts, kleinste Lösung

2.2 *Live Variables Analysis*

Gegeben folgendes Programm in der aus der Vorlesung bekannten While-Sprache:

$[a := 1]^1\ [z := a + y]^2\ \texttt{while}\ [y < z]^3\ \texttt{do}(\ [a := a + 1]^4\ \texttt{if}\ [z > 4]^5\ \texttt{then}\ [\texttt{skip}]^6\ \texttt{else}\ [z := z - 1]^7)\ [a := a]^8$

Auf welcher Menge operiert die Live Variables Analysis.

Lösung:
Es muss angegeben werden welche Variablen an einem Programmpunkt lebendig sind. Deshalb benutzt man die Potenzmenge aller Variablen. Bei Nielson mit $\mathcal{P}(\mathbf{Var}_\star)$ bezeichnet.

Führe eine Live Variable Analysis für das Programm durch. Erstelle dafür zunächst eine Tabelle mit den Gen- und Kill-Mengen der einzelnen Blöcke und führe danach eine Fixpunktiteration durch.

Lösung:

l	$kill_{\mathrm{LV}}(l)$	$gen_{\mathrm{LV}}(l)$
1	$\{a\}$	$\emptyset$
2	$\{z\}$	$\{a, y\}$
3	$\emptyset$	$\{y, z\}$
4	$\{a\}$	$\{a\}$
5	$\emptyset$	$\{z\}$
6	$\emptyset$	$\emptyset$
7	$\{z\}$	$\{z\}$
8	$\{a\}$	$\{a\}$

l	$\mathrm{LV}_{entry}(l)$	$\mathrm{LV}_{exit}(l)$
1	$\{a\}$	$\{a, y\}$
2	$\{a, y\}$	$\{a, y, z\}$
3	$\{a, y, z\}$	$\{a, y, z\}$
4	$\{a, y, z\}$	$\{a, y, z\}$
5	$\{a, y, z\}$	$\{a, y, z\}$
6	$\{a, y, z\}$	$\{a, y, z\}$
7	$\{a, y, z\}$	$\{a, y, z\}$
8	$\{a\}$	$\{a\}$

2.3 *Very Busy Expressions Analysis*

Betrachte folgendes Programm in der aus der Vorlesung bekannten While-Sprache:

$[a := 1]^1\,;\texttt{while}\ [y < z]^2\ \texttt{do}(\ [z := a * b]^3\,;\texttt{if}\ [a < b]^4\ \texttt{then}\ [y := (y + z) + 1]^5\ \texttt{else}\ [y := (y + z) - 1]^6)$

Auf welcher Menge operiert die Very Busy Expression Analyse?

Lösung:
Es geht darum welche Ausdrücke auf jeden Fall benötigt werden. Es wird also auf der Potenzmenge aller Ausdrücke gearbeitet. Allerdings interessieren uns nur nicht-triviale/zusammengesetzte Ausdrücke (also zum Beispiel $y < z$ aber nicht y). Bei Nielson wird diese Menge mit $\mathcal{P}(\mathbf{AExp}_\star)$ bezeichnet.

Führe eine Very Busy Expression Analysis für das Programm durch. Erstelle dafür zunächst eine Tabelle mit den Gen- und Kill-Mengen der einzelnen Blöcke und führe danach eine Fixpunktiteration durch.

Lösung:

l	$kill_{\mathrm{VB}}(l)$	$gen_{\mathrm{VB}}(l)$
1	$\emptyset$	$\{a * b, a < b\}$
2	$\{y < z\}$	$\emptyset$
3	$\{a * b\}$	$\{y < z, y + z, (y + z) + 1, (y + z) - 1\}$
4	$\{a < b\}$	$\emptyset$
5	$\{y + z, (y + z) + 1\}$	$\{y + z, y < z, (y + z) + 1, (y + z) - 1\}$
6	$\{y + z, (y + z) - 1\}$	$\{y + z, (y + z) + 1, (y + z) - 1\}$

l	VB_{entry}	VB_{exit}
1	$\{a * b, a < b, y < z\}$	$\{y < z\}$
2	$\{y < z\}$	$\emptyset$
3	$\{a * b, a < b\}$	$\{y + z, a < b\}$
4	$\{y + z, a < b\}$	$\{y + z\}$
5	$\{y + z, (y + z) + 1\}$	$\emptyset$
6	$\{y + z, (y + z) - 1\}$	$\emptyset$