# Optimierungen in Übersetzern: Verfahren

Marco Ammon (my04mivo)

11. August 2020

## Inhaltsverzeichnis

# 1 Kontrollflussanalyse

## 1.1 Kontrollflussgraph

- Gerichteter Graph
- Knoten: Grundblöcke (meist maximal)
- Kante zwischen zwei Blöcken $A$ und $B$ wenn $B$ direkt nach $A$ ausgeführt werden kann (etwa [un-]bedingter Sprung oder Fallthrough)
- Synthetische Ergänzung um Entry- und Exit-Knoten, die mit Kante verbunden sind
- Kontrollflussabhängigkeit: Bei Verzweigungsknoten $v$ mit direkten Nachfolgern $a$ und $b$: $y$ kontrollflussabhängig von $v \Leftrightarrow$ mindestens ein Pad von $a$ zum Exit-Knoten ohne $y$ und jeder Pfad von $b$ zum Exit-Knoten über $y$

## 1.2 Dominanz

- Knoten $x$ dominiert $y$ ($x \geq\geq y$), wenn jeder Pfad von Wurzel zu $y$ durch $x$ laufen muss
- Strikte Dominanz $x >> y$, falls zusätzlich $x \neq y$ gilt
- ImmDom$[y]$ ist strikter Dominator von $y$, der $y$ am Nächsten ist
- Dominatorbaum enthält jeden Knoten als Kind seines ImmDomms $\rightarrow$ Pfad zwischen $x$ und $z$ in Dominatorbaum $\Leftrightarrow x >> z$

### 1.2.1 Berechnung der Dominatoren $D(n)$ eines Knoten $n$

**Iterativer Fixpunkt-Algorithmus (Lengauer)**

- mit $\mathcal{O}(|E||N|^2)$
- Zunächst Überapproximation der Dominatorenmenge
- Initialisierung aller $D(n) \in N$ mit $N$ außer Startknoten $S$ mit $D(S) = S$
- Bis Fixpunkt erreicht ist: alle $D(n)$ zu $D'(n) = \{n\} \cup \bigcap_{(p,n)\in E} D(p)$
- $n$ am besten in Tiefensuchereihenfolge durchlaufen

**Verfahren mit Spannendem Tiefenbaum $T$**

- Besuch des KFG in Tiefensuchereihenfolge mit zugehöriger Nummerierung:
  - „Spannende“ Kanten gehen zu frisch nummerierten Knoten
  - Rückschreitende Kanten gehen zu Vorgänger (kleinere DFS-Nummer) in $T$
  - Fortschreitende Kanten gehen zu Nachfolger (größere DFS-Nummer) in $T$
  - Kreuzkanten führen in früher besuchten Ast in $T$
- Dominatoren $D(n)$ liegen auf jeden Fall „über“ $n$ in $T$
- Berechnung der Semidominatoren SemDom$[w]$ in Reihenfolge fallender DFS-Nummern:
  - Direkte Vorgänger auf $T$ sind Kandidaten
  - $\min_{u\in \mathrm{Pred}(w)}$ SemDom$[u]$ ist Kandidat
  - Minimum der Kandidaten ist SemDom$[w]$

- Berechnung von ImmDom[$w$] durch Durchlaufen in Tiefenordnung von SemDom[$w$] nach $w$:
  - Jeweils alle Vorgänger $u$ untersuchen und $u$ mit kleinstem SemDom[$u$] finden
  - $$\text{ImmDom}[w] = \begin{cases} \text{SemDom}[u] & \text{falls}\,\text{SemDom}[w] = \text{SemDom}[u] \\ \text{ImmDom}[u] & \text{sonst} \end{cases}$$

### 1.2.2 Dominanzgrenze

- Dominanzgrenze $DG[x]$ enthält Knoten $y$, die einen von $x$ dominierten Vorgänger besitzen, aber nicht von $x$ streng dominiert werden
- Berechnung der $DG[x]$:

$$DG[x] = DG_{\text{local}}[x] \cup \bigcup_{z \in N, \text{ImmDom}[z]=x} DG_{\text{up}}[x, z]$$
$$DG_{\text{local}}[x] = \{y \in \text{Succ}(x) \mid \text{ImmDom}[y] \neq x\}$$
$$DG_{\text{up}}[x, z] = \{y \in DG[z] \mid \text{ImmDom}[y] \neq x\}$$

- Invertierung der Dominanzgrenzen liefert Kontrollflussabhängigkeiten

## 1.3 Schleifenerkennung

- Region:
  - Untergraph mit einem „Header“ $d$, der (potentiell mehrere) Eingangskante von außerhalb besitzt
  - Wichtige Region: maximale Region mit $d$ dominiert alle Knoten der Region
  - Hierarchischer Flussbaum: Baum der Regionen
- Rückwärtskante: Kante $(n, d)$ mit $d \geq\geq n$
- Natürliche Schleife:
  - Rückwärtskante $(n, d)$ sowie alle Knoten $k$ mit $d \geq\geq k$ und es gibt einen Pfad von $k$ nach $n$ ohne $d$
  - Bestimmung mit Worklist-Algorithmus, der bei $n$ beginnt und rekursiv die Vorgänger bis $d$ durchläuft und in Menge aufnimmt
- Suche nach Rückwärtskanten und natürlichen Schleifen in wichtigen Regionen ausreichend
- „Unsaubere“ Regionen:
  - ein Knoten dominiert nachgeordneten Zyklus
  - Erkennung durch Prüfung der Reduzierbarkeit des Graphs:
    * Entfernung der Rückwärtskanten aus KFG $\rightarrow$ azyklischer Graph, in dem jeder Knoten von der Wurzel erreicht werden kann $\Leftrightarrow$ KFG frei von unnatürlichen Schleifen
    * Alternative mit Transformationen: Am Ende Graph aus einem einzigen Knoten $\Leftrightarrow$ KFG reduzierbar (ohne Zyklen)

      **T1-Transformation** Selbstschleifen aus Graph löschen

      **T2-Transformation** Knoten mit eindeutigem Vorgänger mit diesem zusammenfassen

# 2 Datenflussanalyse

## 2.1 Datenabhängigkeiten

- Schreiben vor Lesen:
- Schreiben vor Schreiben: Ausgabeabhängigkeit
- Lesen vor Schreiben: Anti-Abhängigkeit

# 3 Aliasanalyse

# 4 Induktionsvarianten und schleifeninvarianter Code

# 5 Schleifen und Arrays

# 6 Schleifentransformationen

# 7 Schleifenrestrukturierungen