Arbeitsplan

In der Folge werden wir definieren...

- Die Menge der *lebenszeitoptimalen* PRE-Transformationen
- Die LCM-Transformation als eindeutig bestimmte einzige lebenszeitoptimale PRE-Transformation

Zur Formalisierung

...ist der Begriff des *Lebenszeitbereichs* zentral.

Sei $CM \in \mathcal{CM}$.

- *Lebenszeitbereich*
  LtRg $(CM) =_{df}$
  $\{p \mid Insert_{CM}(p_1) \wedge Repl_{CM}(p_{\lambda_p}) \wedge \neg Insert^{\exists}_{CM}(p]1, \lambda_p])\}$
- *Erstbenutzungslebenszeitbereich*
  FU-LtRg $(CM) =_{df}$
  $\{p \in$ *LtRg* $(CM) \mid \forall\, q \in$ *LtRg* $(CM).\ (q \sqsubseteq p) \Rightarrow (q = p)\}$

Erste Aussagen

Erstbenutzungslebenszeitbereichslemma
Sei $CM \in \mathcal{CM}$, $p \in \mathbf{P}[s,e]$ und seien i_1, i_2, j_1, j_2 Indizes so dass $p[i_1, j_1] \in$ *FU-LtRg* (CM) und $p[i_2, j_2] \in$ *FU-LtRg* (CM). Dann gilt:

- entweder stimmen $p[i_1, j_1]$ und $p[i_2, j_2]$ überein, d.h. $i_1 = i_2$ und $j_1 = j_2$, oder
- $p[i_1, j_1]$ und $p[i_2, j_2]$ sind disjunkt, d.h., $j_1 < i_2$ oder $j_2 < i_1$.

Lebenszeitbesser, lebenszeitoptimal

Eine CM-Transformation $CM \in \mathcal{CM}$ heißt *lebenszeitbesser* als eine CM-Transformation $CM' \in \mathcal{CM}$ gdw

$$\forall\, p \in \textit{LtRg}\,(CM)\ \exists\, q \in \textit{LtRg}\,(CM').\ p \sqsubseteq q$$

Bemerkung: Die Relation "lebenszeitbesser" ist eine partielle Ordnung, d.h. eine reflexive, transitive und antisymmetrische Relation.

Lebenszeitoptimalität

Definition [Lebenszeitoptimale CM-Transformation]
Eine berechnungsoptimale CM-Transformation $CM \in \mathcal{CM}_{CmpOpt}$ heißt *lebenszeitoptimal* gdw CM ist lebenszeitbesser als jede andere berechnungsoptimale CM-Transformation.

Wir bezeichnen die Menge der lebenszeitoptimalen CM-Transformationen mit $\mathcal{CM}_{LtOpt}$.

Wdhg: Mengen und Relationen 1(2)

Sei M eine Menge und R eine Relation auf M, d.h. $R \subseteq M \times M$.

Dann heißt R...

- *reflexiv* gdw. $\forall\, m \in M.\ m\,R\,m$
- *transitiv* gdw. $\forall\, m, n, p \in M.\ m\,R\,n \ \wedge\ n\,R\,p \ \Rightarrow\ m\,R\,p$
- *antisymmetrisch* gdw. $\forall\, m, n \in M.\ m\,R\,n \wedge n\,R\,m \ \Rightarrow\ m = n$

Wo wir dabei sind...

- *symmetrisch* gdw. $\forall\, m, n \in M.\ m\,R\,n \iff n\,R\,m$
- *total* gdw. $\forall\, m, n \in M.\ m\,R\,n \ \vee\ n\,R\,m$

Wdhg: Mengen und Relationen 2(2)

Eine Relation R auf M heißt

- *Quasiordnung* gdw. R ist reflexiv und transitiv
- *partielle Ordnung* gdw. R ist reflexiv, transitiv und antisymmetrisch

Zur Vollständigkeit sei noch ergänzt...

- *Äquivalenzrelation* gdw. R ist reflexiv, transitiv und symmetrisch

...eine partielle Ordnung ist also eine antisymmetrische Quasiordnung, eine Äquivalenzrelation eine symmetrische Quasiordnung.

Eindeutigkeit lebenszeitoptimaler PRE

Offensichtlich gilt:

$$\mathcal{CM}_{LtOpt} \subseteq \mathcal{CM}_{CmpOpt} \subseteq \mathcal{CM}_{Adm} \subset \mathcal{CM}$$

Es gilt sogar weitergehend:

Theorem [Eindeutigkeit lebenszeitoptimaler CM-Transformationen]

$$|\,\mathcal{CM}_{LtOpt}\,| \leq 1$$

Zur Entwicklung der LCM-Transformation

Zunächst folgende Beobachtung:

Lemma
$\forall\, CM \in \mathcal{CM}_{CmpOpt}\ \forall\, p \in \mathit{LtRg}\,(CM)\ \exists\, q \in \mathit{LtRg}\,(BCM).\ p \sqsubseteq q.$

Intuitiv:

- Keine berechnungsoptimale CM-Transformation platziert die Berechnungen früher als die BCM-Transformation
- Die BCM-Transformation ist diejenige berechnungsoptimale CM-Transformation mit maximalem Registerdruck

Verzögerbarkeit

Definition [Verzögerbarkeit]
$\forall\, n \in N.\ \mathit{Delayed}\,(n)\ \Longleftrightarrow_{df}$
$\quad \forall\, p \in \mathbf{P}[s,n]\ \exists\, i \le \lambda_p.\ \mathtt{Earliest}(p_i) \wedge \neg \mathit{Comp}^{\exists}(\,p[i,\lambda_p[\,)$

Das Verzögerbarkeitslemma

Verzögerbarkeitslemma

1. $\forall\, n \in N.\ \mathit{Delayed}\,(n)\ \Rightarrow\ \mathit{D\text{-}Safe}(n)$
2. $\forall\, p \in \mathbf{P}[s,e]\ \forall i \le \lambda_p.\ \mathit{Delayed}\,(p_i)\ \Rightarrow\ \exists\ j \le i \le l.\ p[j,l] \in \mathit{FU\text{-}LtRg}\,(BCM)$
3. $\forall\, CM \in \mathcal{CM}_{CmpOpt}\ \forall\, n \in N.\ \mathit{Comp}_{\,CM}(n)\ \Rightarrow\ \mathit{Delayed}\,(n)$

Spätestheit

Definition [Spätestheit]
$\forall\, n \in N.\ \mathit{Latest}\,(n) =_{df}\ \mathit{Delayed}\,(n) \wedge (\,\mathit{Comp}\,(n) \vee \bigvee_{m \in succ(n)} \neg \mathit{Delayed}\,(m))$

Das Spätestheitlemma

Spätestheitlemma

1. $\forall\, p \in \mathit{LtRg}\,(BCM)\ \exists\, i \le \lambda_p.\ \mathit{Latest}\,(p_i)$
2. $\forall\, p \in \mathit{LtRg}\,(BCM)\ \forall\, i \le \lambda_p.\ \mathit{Latest}\,(p_i) \Rightarrow \neg \mathit{Delayed}^{\exists}(\,p]i,\lambda_p]\,)$

Die ALCM-Transformation

Die "Almost Lazy Code Motion" Transformation...

- $\mathit{Insert}_{ALCM}(n) =_{df} \mathit{Latest}\,(n)$
- $\mathit{Repl}_{ALCM}(n) =_{df} \mathit{Comp}\,(n)$

Fast lebenszeitoptimal

Definition [Fast lebenszeitoptimale CM-Transformation]
Eine berechnungsoptimale CM-Transformation $CM \in \mathcal{CM}_{CmpOpt}$ heißt *fast lebenszeitoptimal* gdw

$\forall\, p \in \mathit{LtRg}\,(CM).\ \lambda_p \ge 2 \Rightarrow$

$\quad \forall\, CM' \in \mathcal{CM}_{CmpOpt}\ \exists\, q \in \mathit{LtRg}\,(CM').\ p \sqsubseteq q$

Wir bezeichnen die Menge der fast lebenszeitoptimalen CM-Transformationen mit $\mathcal{CM}_{ALtOpt}$.

Das ALCM-Theorem

ALCM-Theorem

Die $ALCM$-Transformation ist fast lebenszeitoptimal, d.h., $ALCM \in \mathcal{CM}_{ALtOpt}$.

Isolierte Berechnungen

Definition [CM-Isolation]

$\forall\, CM \in \mathcal{CM}\ \forall\, n \in N.\ Isolated_{CM}(n) \iff_{df}$
$\forall\, p \in \mathbf{P}[n,e]\ \forall\, 1 < i \leq \lambda_p.\ Repl_{CM}(p_i) \Rightarrow Insert^{\exists}_{CM}(\,p]1,i]\,)$

Das Isolationslemma

Isolationslemma

1. $\forall\, CM \in \mathcal{CM}\ \forall\, n \in N.\ Isolated_{CM}(n) \iff$
   $\forall\, p \in LtRg\,(CM).\ \langle n \rangle \sqsubseteq p \Rightarrow \lambda_p = 1$
2. $\forall\, CM \in \mathcal{CM}_{CmpOpt}\ \forall\, n \in N.\ Latest\,(n) \Rightarrow$
   $(Isolated_{CM}(n) \iff Isolated_{BCM}(n))$

Die LCM-Transformation

- $Insert_{LCM}(n) =_{df} Latest\,(n) \wedge \neg Isolated_{BCM}(n)$
- $Repl_{LCM}(n) =_{df}$
  $Comp\,(n) \wedge \neg(Latest\,(n) \wedge Isolated_{BCM}(n))$

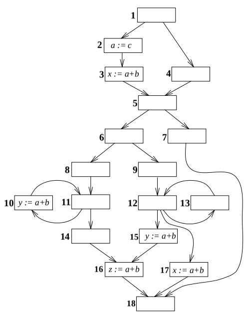
Das LCM-Theorem

LCM-Theorem

Die LCM-Transformation ist lebenszeitoptimal, d.h., $LCM \in \mathcal{CM}_{LtOpt}$.

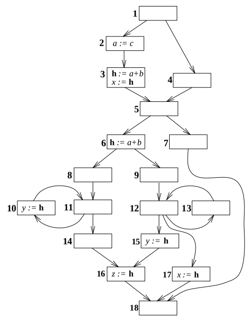
Ein größeres Beispiel zur Illustration (1)

Das Ausgangsprogramm...



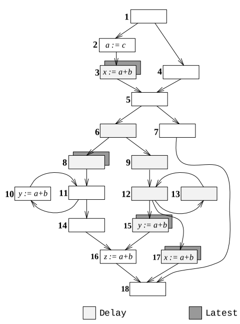
Ein größeres Beispiel zur Illustration (2)

Das Resultat der BCM-Transformation...



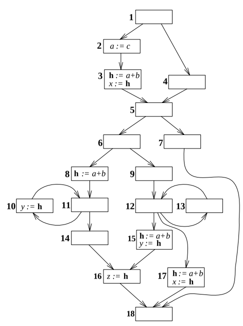
Ein größeres Beispiel zur Illustration (3)

Verzögerte und späteste Berechnungspunkte...



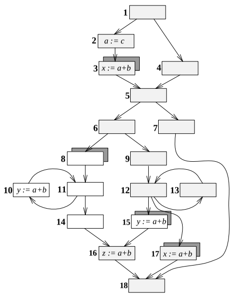
Ein größeres Beispiel zur Illustration (4)

Das Resultat der ALCM-Transformation...



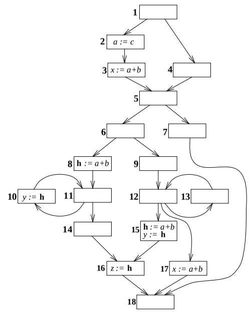
Ein größeres Beispiel zur Illustration (5)

Späteste und isolierte Berechnungspunkte...



Ein größeres Beispiel zur Illustration (6)

Das Resultat der LCM-Transformation...



Zur Implementierung der BCM-Transformation auf EA-Graphen

...auf Einzelanweisungsniveau, hier für knotenbenannte EA-Graphen.

Beachte: ...wir nehmen für das folgende an, dass nur kritische Kanten gespalten sind (deshalb N- und X-Einsetzungen).

Busy Code Motion (EA-1)

1. Die Analysen für Aufwärts- und Abwärtssicherheit

Lokale Prädikate:

- $\textbf{COMP}_\iota(t)$: ι berechnet t.
- $\textbf{TRANSP}_\iota(t)$: ι modifiziert keinen Operanden von t.

Busy Code Motion (EA-2)

Das Gleichungssystem für Aufwärtssicherheit:

$$\text{N-USAFE}_\iota = \begin{cases} \mathit{false} & \text{falls } \iota = s \\ \prod_{\hat\iota \in pred(\iota)} \text{X-USAFE}_{\hat\iota} & \text{sonst} \end{cases}$$

$$\text{X-USAFE}_\iota = (\text{N-USAFE}_\iota + \textbf{COMP}_\iota) \cdot \textbf{TRANSP}_\iota$$

Busy Code Motion (EA-3)

Das Gleichungssystem für Abwärtssicherheit:

$$\text{N-DSAFE}_\iota = \textbf{COMP}_\iota + \text{X-DSAFE}_\iota \cdot \textbf{TRANSP}_\iota$$

$$\text{X-DSAFE}_\iota = \begin{cases} \mathit{false} & \text{falls } \iota = e \\ \prod_{\hat\iota \in succ(\iota)} \text{N-DSAFE}_{\hat\iota} & \text{sonst} \end{cases}$$

Busy Code Motion (EA-4)

2. Die Transformation: Einsetzungs- und Ersetzungspunkte

Lokale Prädikate:

- N-USAFE*, X-USAFE*, N-DSAFE*, X-DSAFE*: größte Lösungen der Gleichungssysteme für Aufwärts- und Abwärtssicherheit aus Schritt 1.

Busy Code Motion (EA-5)

$$\text{N-INSERT}_\iota^{\text{BCM}} =_{df} \text{N-DSAFE}_\iota^\star \cdot \prod_{\hat\iota \in pred(\iota)} (\overline{\text{X-USAFE}_{\hat\iota}^\star + \text{X-DSAFE}_{\hat\iota}^\star})$$

$$\text{X-INSERT}_\iota^{\text{BCM}} =_{df} \text{X-DSAFE}_\iota^\star \cdot \overline{\textbf{TRANSP}_\iota}$$

$$\text{REPLACE}_\iota^{\text{BCM}} =_{df} \textbf{COMP}_\iota$$

Zur Implementierung der BCM-Transformation auf BB-Graphen (1)

...auf Basisblockniveau, hier für knotenbenannte BB-Graphen.

Beachte: ...wir nehmen für das folgende an, dass (1) nur kritische Kanten gespalten sind (deshalb N- und X-Einsetzungen), und (2) dass alle Redundanzen innerhalb eines Basisblocks schon durch einen Präprozess beseitigt sind.

Zur Implementierung der BCM-Transformation auf BB-Graphen (2)

t-verfeinerte Flussgraphen...

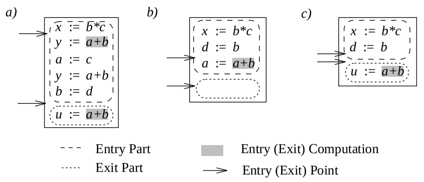
Bezüglich einer Berechnung t lässt sich ein Basisblock $\mathbf{n}$ in zwei disjunkte Teile unterteilen:

- ein *Eingangsteil* (*entry part*), der aus allen Anweisungen bis zu und einschließlich der letzten Modifikation von t besteht
- ein *Ausgangsteil* (*exit part*), der aus den verbleibenden Anweisungen von $\mathbf{n}$ besteht.

Beachte: ein nichtleerer Basisblock hat stets einen nichtleeren Eingangsteil; im Unterschied dazu kann der Ausgangsteil leer sein (zur Illustration siehe folgende Abbildung).

Zur Implementierung der BCM-Transformation auf BB-Graphen (3)

Zur Illustration von Eingangs- und Ausgangsteil eines Basisblocks...



Busy Code Motion (BB-1)

1. Die Analysen für Aufwärts- und Abwärtssicherheit

Lokale Prädikate:

- BB-NCOMP$_\beta(t)$: β enthält eine Anweisung ι, die t berechnet, und der keine Anweisung vorausgeht, die einen Operanden von t modifiziert.
- BB-XCOMP$_\beta(t)$: β enthält eine Anweisung ι, die t berechnet, und weder ι noch irgendeine andere Anweisung von β nach ι modifiziert einen Operanden von t.
- BB-TRANSP$_\beta(t)$: β enthält keine Anweisung, die einen Operanden von t modifiziert.

Busy Code Motion (BB-2)

Das Gleichungssystem für Aufwärtssicherheit:

$$\text{BB-N-USAFE}_\beta = \begin{cases} \mathit{false} & \text{falls } \beta = \mathbf{s} \\ \prod\limits_{\beta \in pred(\beta)} (\text{BB-XCOMP}_\beta + \text{BB-X-USAFE}_\beta) & \text{sonst} \end{cases}$$

$$\text{BB-X-USAFE}_\beta = (\text{BB-N-USAFE}_\beta + \text{BB-NCOMP}_\beta) \cdot \text{BB-TRANSP}_\beta$$

Busy Code Motion (BB-3)

Das Gleichungssystem für Abwärtssicherheit:

$$\text{BB-N-DSAFE}_\beta = \text{BB-NCOMP}_\beta + \text{BB-X-DSAFE}_\beta \cdot \text{BB-TRANSP}_\beta$$

$$\text{BB-X-DSAFE}_\beta = \text{BB-XCOMP}_\beta + \begin{cases} \mathit{false} & \text{falls } \beta = \mathbf{e} \\ \prod\limits_{\beta \in succ(\beta)} \text{BB-N-DSAFE}_\beta & \text{sonst} \end{cases}$$

Busy Code Motion (BB-4)

2. Die Transformation: Einsetzungs- und Ersetzungspunkte

Lokale Prädikate:

- BB-N-USAFE*, BB-X-USAFE*, BB-N-DSAFE*, BB-X-DSAFE*: größte Lösungen der Gleichungssysteme für Aufwärts- und Abwärtssicherheit aus Schritt 1.

Busy Code Motion (BB-5)

$$\text{N-INSERT}^{\text{BCM}}_\beta =_{df} \text{BB-N-DSAFE}^\star_\beta \cdot \prod_{\beta \in pred(\beta)} (\overline{\text{BB-X-USAFE}^\star_\beta + \text{BB-X-DSAFE}^\star_\beta})$$

$$\text{X-INSERT}^{\text{BCM}}_\beta =_{df} \text{BB-X-DSAFE}^\star_\beta \cdot \overline{\text{BB-TRANSP}_\beta}$$

$$\text{N-REPLACE}^{\text{BCM}}_\beta =_{df} \text{BB-NCOMP}_\beta$$

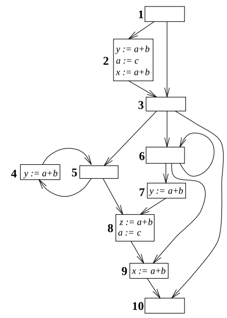
$$\text{X-REPLACE}^{\text{BCM}}_\beta =_{df} \text{BB-XCOMP}_\beta$$

Die Gleichungssysteme für LCM

Ähnlich!

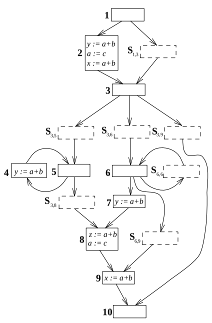
Ein größeres BB-Beispiel zur Illustration (1)

Das Ausgangsprogramm...



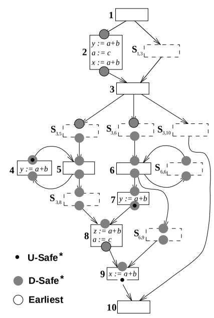
Ein größeres BB-Beispiel zur Illustration (2)

Das Ausgangsprogramm mit kritischen Kanten gespalten...



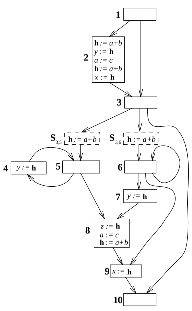
Ein größeres BB-Beispiel zur Illustration (3)

Die Berechnung der frühesten Berechnungspunkte...



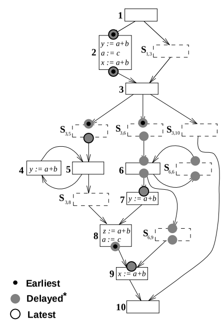
Ein größeres BB-Beispiel zur Illustration (4)

Das Ergebnis der BCM-Transformation...



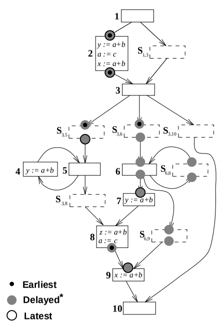
Ein größeres BB-Beispiel zur Illustration (5)

Die Berechnung der spätesten Berechnungspunkte...



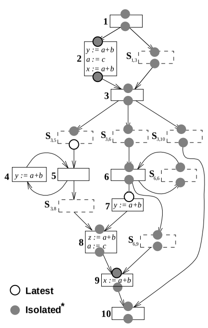
Ein größeres BB-Beispiel zur Illustration (6)

Das Ergebnis der ALCM-Transformation...



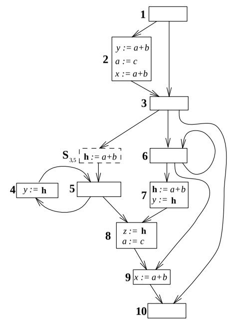
Ein größeres BB-Beispiel zur Illustration (7)

Die Berechnung isolierter Berechnungspunkte...



Ein größeres BB-Beispiel zur Illustration (8)

Das Ergebnis der LCM-Transformation...



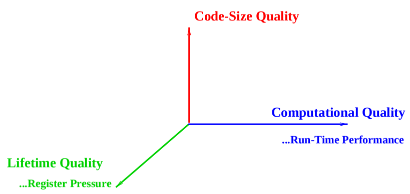
Heutzutage...

Lazy Code Motion ist...

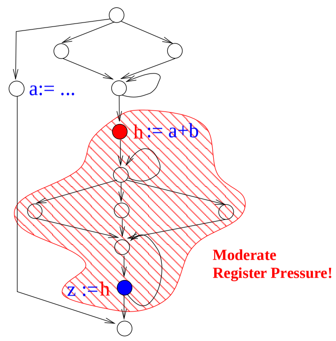
- ...der de-facto Standardalgorithmus für PRE, der in aktuellen state-of-the-art Übersetzern zum Einsatz kommt
  - Gnu compiler family
  - Sun Sparc compiler family
  - ...

In der Folge...

(Modulare) Erweiterung von LCM, um Anwenderprioritäten zu berücksichtigen!



...um auch diese Transformation zu ermöglichen:



There is more than speed!

Vorschau auf die weiteren Vorlesungstermine...

- Mo, 10.12.2007: Vorlesung von 16:15 Uhr bis 17:45 Uhr im Hörsaal 14, TU-Hauptgebäude
- *Mo, 17.12./24.12./31.12.2007: Keine Vorlesung(en)!* (*Ferialzeit*)
- Mo, 14.01.2008: Vorlesung von 16:15 Uhr bis 17:45 Uhr im Hörsaal 14, TU-Hauptgebäude