Automatisierte Logik und Programmierung

Lektion 19

Korrektheitserhaltende Optimierungen

1. Logische Vereinfachungen
2. Partielle Auswertung
3. Endliche Differenzierung
4. Fallanalyse
5. Datentyp-Verfeinerung

Optimierung schematisch erzeugter Algorithmen

Eliminiere überflüssige Berechnungen

- **Simplifikation: Logische Vereinfachung**
  – Äquivalenzumwandlung von Teilausdrücken, ggf. im Kontext
- **Partielle Auswertung**
  – Symbolische Auswertung von Ausdrücken mit konstanten Komponenten
- **Endliche Differenzierung**
  – Inkrementelle Berechnung von Teilausdrücken in Schleifen
- **Fallanalyse**
  – Analyse und Vereinfachung von Teilausdrücken
- **Datentyp-Verfeinerung**
  – Bestimmung konkreter Implementierungen für abstrakte Datentypen
- **Sprachabhängige Optimierung & Compilierung**
  – Ausnutzen der Besonderheiten einer konkreten Zielsprache

Simplifikation

Logische Vereinfachung von Teilausdrücken

- **Transformation in äquivalente Ausdrücke**
  - Term-Rewriting mit gerichteten Gleichungen
  - Gleichungen formuliert als Lemmata der Wissensbank
  - Identifikation geeigneter Lemmata über Operatoren im Ausdruck

    $\texttt{a} \in \texttt{(x.[])} \circ \texttt{(b.L)} \longrightarrow \texttt{a=x} \lor \texttt{a=b} \lor \texttt{a} \in \texttt{L}$

- **Kontextunabhängige CI-Simplifikation**
  - Anwendung einfacher Gleichungen ohne Rahmenbedingungen
  - Nur der aktuelle Teilausdruck muß betrachtet werden
  - Automatische Ausführung solange anwendbare Lemmata vorhanden
- **Kontextabhängige CD-Simplifikation**
  - Anwendung von Gleichungen mit Rahmenbedingungen
  - Rahmenbedingungen müssen durch Kontext erfüllt werden
- **Benutzerinteraktion**
  - Auswahl von Teilausdruck und Art der Vereinfachung (CI/CD)
  - Optional bei CD: Begrenzung der Vor- und Rückwärtsinferenzen

Ausgangspunkt: schematischer Globalsuchalgorithmus

FUNCTION Costas $(n{:}\mathbb{Z})$ WHERE $n \geq 1$

RETURNS $\{p{:}Seq(\mathbb{Z}) \mid perm(p,\{1..n\}) \wedge \forall j{<}|p|.nodups(dtrow(p,j))\}$

$\equiv$ if $nodups([]) \wedge \forall j{<}|[]|.nodups(dtrow([],j))$

then $Costas_{gs}(n,[])$ else $\emptyset$

FUNCTION $Costas_{aux}$ $(n,s{:}\mathbb{Z}\times Seq(\mathbb{Z}))$

WHERE $n \geq 1 \wedge range(s) \subseteq \{1..n\} \wedge nodups(s) \wedge \forall j{<}|s|.nodups(dtrow(s,j))$

RETURNS $\{p{:}Seq(\mathbb{Z}) \mid perm(p,\{1..n\}) \wedge s \sqsubseteq p \wedge \forall j{<}|p|.nodups(dtrow(p,j))\}$

$$\equiv \{p \mid p \in \{s\} \wedge perm(p,\{1..n\}) \wedge \forall j{<}|p|.nodups(dtrow(p,j))\}$$

$$\cup \bigcup\{Costas_{gs}(n,t) \mid t \in \{s{\cdot}i \mid i \in \{1..n\}\} \wedge nodups(t) \wedge \forall j{<}|t|.nodups(dtrow(t,j))\}$$

CI-Simplifikationen im Hauptalgorithmus Costas

```
FUNCTION Costas (n:ℤ) WHERE n≥1
  RETURNS {p:Seq(ℤ) | perm(p,{1..n}) ∧ ∀j<|p|.nodups(dtrow(p,j))}
≡ if nodups([]) ∧ ∀j<|[]|.nodups(dtrow([],j))
     then Costas_gs(n,[]) else ∅
```

```
FUNCTION Costas (n:ℤ) WHERE n≥1
  RETURNS {p:Seq(ℤ) | perm(p,{1..n}) ∧ ∀j<|p|.nodups(dtrow(p,j))}
≡ Costas_gs(n,[])
```

CI-Simplifikationen in der Hilfsfunktion Costas_{aux}

$$
\begin{array}{l}
\texttt{FUNCTION Costas}_{aux}\ \texttt{(n,s:}\mathbb{Z}\times\texttt{Seq(}\mathbb{Z}\texttt{))} \\
\quad \texttt{WHERE n}\geq\texttt{1} \wedge \texttt{range(s)}\subseteq\{\texttt{1..n}\} \wedge \texttt{nodups(s)} \wedge \forall\texttt{j<|s|.nodups(dtrow(s,j))} \\
\quad \texttt{RETURNS } \{\texttt{p:Seq(}\mathbb{Z}\texttt{)}\mid\texttt{perm(p,}\{\texttt{1..n}\}\texttt{)} \wedge \texttt{s}\sqsubseteq\texttt{p} \wedge \forall\texttt{j<|p|.nodups(dtrow(p,j))}\} \\
\equiv \qquad \{\texttt{p}\mid\texttt{p}\in\{\texttt{s}\} \wedge \texttt{perm(p,}\{\texttt{1..n}\}\texttt{)} \wedge \forall\texttt{j<|p|.nodups(dtrow(p,j))}\} \\
\quad \cup \bigcup\{\texttt{Costas}_{gs}\texttt{(n,t)}\mid\texttt{t}\in\{\texttt{s}\cdot\texttt{i}\mid\texttt{i}\in\{\texttt{1..n}\}\} \wedge \texttt{nodups(t)} \\
\qquad\qquad\qquad\qquad\qquad\qquad \wedge \forall\texttt{j<|t|.nodups(dtrow(t,j))}\}
\end{array}
$$

$$
\begin{array}{l}
\texttt{FUNCTION Costas}_{aux}\ \texttt{(n,s:}\mathbb{Z}\times\texttt{Seq(}\mathbb{Z}\texttt{))} \\
\quad \texttt{WHERE n}\geq\texttt{1} \wedge \texttt{range(s)}\subseteq\{\texttt{1..n}\} \wedge \texttt{nodups(s)} \wedge \forall\texttt{j<|s|.nodups(dtrow(s,j))} \\
\quad \texttt{RETURNS } \{\texttt{p:Seq(}\mathbb{Z}\texttt{)}\mid\texttt{perm(p,}\{\texttt{1..n}\}\texttt{)} \wedge \texttt{s}\sqsubseteq\texttt{p} \wedge \forall\texttt{j<|p|.nodups(dtrow(p,j))}\} \\
\equiv \qquad \texttt{if perm(s,}\{\texttt{1..n}\}\texttt{)} \wedge \forall\texttt{j<|s|.nodups(dtrow(s,j)) then } \{\texttt{s}\} \texttt{ else } \emptyset \\
\quad \cup \bigcup\{\texttt{Costas}_{gs}\texttt{(n,s}\cdot\texttt{i)}\mid\texttt{i}\in\{\texttt{1..n}\} \wedge \texttt{nodups(s}\cdot\texttt{i)} \\
\qquad\qquad\qquad\qquad\qquad \wedge \forall\texttt{j<|s}\cdot\texttt{i|.nodups(dtrow(s}\cdot\texttt{i,j))}\}
\end{array}
$$

Kontextabhängige Simplifikation

Vereinfachung unter Berücksichtigung von Kontext

- **Anwendung bedingter Gleichungen**
  - Gleichung $A \equiv B$ mit Vorbedingung C
  - Formuliert als Lemma der Form $C \Rightarrow A \equiv B$

- **Anwendbarbarkeit abhängig vom Kontext**
  - Ersetze Teilausdruck A durch B, wenn Kontext Bedingung C erfüllt
  - Kontext ergibt sich aus Syntaxbaum des Gesamtausdrucks
    - Bei Programmen: benachbarter Programmcode und Vorbedingung
  - Hauptanwendung: Elimination redundanter Teilausdrücke

- **Komplizierter als CI-Simplifikation**
  - Vorbedingung kann auch Teil einer Gleichung $C \wedge A \equiv B$ sein
  - Gleichung kann oft auch von rückwärts angewandt werden
  - Automatische Anwendung muß tiefenbeschränkt werden

CD-Simplifikationen in der Hilfsfunktion Costas_{aux}

$$
\begin{array}{l}
\texttt{FUNCTION Costas}_{aux}\ \texttt{(n,s:}\mathbb{Z}\times\texttt{Seq(}\mathbb{Z}\texttt{))}\\
\quad \texttt{WHERE n}\geq\texttt{1}\ \wedge\ \texttt{range(s)}\subseteq\{\texttt{1..n}\}\ \wedge\ \texttt{nodups(s)}\ \wedge\ \forall\texttt{j<|s|.nodups(dtrow(s,j))}\\
\quad \texttt{RETURNS}\ \{\texttt{p:Seq(}\mathbb{Z}\texttt{)}\mid\texttt{perm(p,}\{\texttt{1..n}\}\texttt{)}\wedge\ \texttt{s}\sqsubseteq\texttt{p}\wedge\ \forall\texttt{j<|p|.nodups(dtrow(p,j))}\}\\
\equiv \qquad \texttt{if perm(s,}\{\texttt{1..n}\}\texttt{)}\ \wedge\ \forall\texttt{j<|s|.nodups(dtrow(s,j)) then}\ \{\texttt{s}\}\ \texttt{else}\ \emptyset\\
\quad \cup\ \bigcup\{\texttt{Costas}_{gs}\texttt{(n,s}\cdot\texttt{i)}\mid\texttt{i}\in\{\texttt{1..n}\}\ \wedge\ \texttt{nodups(s}\cdot\texttt{i)}\\
\qquad\qquad\qquad\qquad\qquad\qquad \wedge\ \forall\texttt{j<|s}\cdot\texttt{i|.nodups(dtrow(s}\cdot\texttt{i,j))}\}
\end{array}
$$

$$
\begin{array}{l}
\texttt{FUNCTION Costas}_{aux}\ \texttt{(n,s:}\mathbb{Z}\times\texttt{Seq(}\mathbb{Z}\texttt{))}\\
\quad \texttt{WHERE n}\geq\texttt{1}\ \wedge\ \texttt{range(s)}\subseteq\{\texttt{1..n}\}\ \wedge\ \texttt{nodups(s)}\ \wedge\ \forall\texttt{j<|s|.nodups(dtrow(s,j))}\\
\quad \texttt{RETURNS}\ \{\texttt{p:Seq(}\mathbb{Z}\texttt{)}\mid\texttt{perm(p,}\{\texttt{1..n}\}\texttt{)}\wedge\ \texttt{s}\sqsubseteq\texttt{p}\wedge\ \forall\texttt{j<|p|.nodups(dtrow(p,j))}\}\\
\equiv \qquad \texttt{if}\ \{\texttt{1..n}\}\backslash\texttt{range(s)=}\emptyset\ \texttt{then}\ \{\texttt{s}\}\ \texttt{else}\ \emptyset\\
\quad \cup\ \bigcup\{\texttt{Costas}_{gs}\texttt{(n,s}\cdot\texttt{i)}\mid\texttt{i}\in\{\texttt{1..n}\}\backslash\texttt{range(s)}\ \wedge\ \forall\texttt{j<|s|.}\ \texttt{s}_{|s\cdot i|-j}\texttt{-i}\notin\texttt{dtrow(s,j)}\}
\end{array}
$$

Auswertung von Ausdrücken mit Konstanten

- **Symbolische Auswertung zur Entwurfszeit**
  - Reduktion von Ausdrücken, deren "Wert" bestimmt werden kann
  - Anwendbar, solange ein konstanter Teilausdruck vorhanden
  - Formale Technik: Auffalten von Definitionen + Simplifikation

    Je nach Inhalt der Wissensbank auch reine Simplifikation

Beispiel:

	|[x;5] $\circ$ L|	unfold append
$\mapsto$	|x.([5] $\circ$ L)|	unfold append
$\mapsto$	|x.(5.([] $\circ$ L))|	unfold append
$\mapsto$	|x.(5.L)|	unfold length
$\mapsto$	1 + |5.L|	unfold length
$\mapsto$	1 + 1 + |L|	simplify
$\mapsto$	2 + |L|	

- **Benutzerinteraktion:**
  - Auswahl von auszuwertendem Ausdruck und Optimierungstechnik

Endliche Differenzierung

- **Inkrementelle Berechnung wiederkehrender Teilausdrücke**
  - Ersetze Teilausdruck $E[x]$ in $f(x)$ durch neue Variable c
  - Initialisiere mit c Ausdruck für Basiswerte
  - Bestimme differentielle Veränderungen bei rekursivem Aufruf
  - Effizienter als ständige Neuberechnung

- **Endliche Differenzierung am Beispiel**

```
FUNCTION Costas_aux(n,s:ℤ×Seq(ℤ))  WHERE ...  RETURNS ...
≡      if {1..n}\range(s)=∅ then {s} else ∅
   ∪ ⋃{ Costas_gs(n,s·i) | i ∈ {1..n}\range(s) ∧ ∀j<|s|. s_{|s·i|−j} −i ∉ dtrow(s,j)}
```

```
FUNCTION Costas_aux(n,s,pool ...)  WHERE ...  pool = {1..n}\range(s)
≡      if pool=∅ then {s} else ∅
   ∪ ⋃{ Costas_gs(n,s·i,pool\{i}) | i ∈ pool
                                  ∧ ∀j<|s|. s_{|s·i|−j} −i ∉ dtrow(s,j)}
```

  - Benutzer identifiziert `{1..n}\range(s)` als wiederkehrende Berechnung
  - System ändert $\texttt{Costas}_{aux}$`(n,s)` in $\texttt{Costas}_{aux}$`(n,s,pool)` und ändert Aufrufe von $\texttt{Costas}_{aux}$ entsprechend
  - System ergänzt `pool = {1..n}\range(s)` zur Eingabebedingung
  - System vereinfacht Vorkommen von `{1..n}\range(s)` zu `pool`

Endliche Differenzierung – Formalia

- **Abstraktion über Ausdruck $E[x]$ in Funktion $f(x)$**

  `FUNCTION` $main...$ `WHERE ... RETURNS ... SUCH THAT...` $\equiv$ $....f(x_0)....$

  `FUNCTION` f`(`x`:`D`):`R `WHERE` $I[x]$ `RETURNS` y `SUCH THAT` $O[x,y]$

  $\equiv$ $....E[x]....f(\,t[x]\,)..$

  – Erweitere f zu neuem f', so daß $f(x) = f'(x, E[x])$

  – Ersetze alle Aufrufe der Art $f(\,t[x]\,)$ durch $f'(\,t[x],\, E[t[x]]\,)$

  – Ergänze Gleichung $c{=}E[x]$ zur Eingabebedingung von f'

  `FUNCTION` $main...$ `WHERE ... RETURNS ... SUCH THAT...` $\equiv$ $....f'(\,x_0,\, E[x_0]\,)....$

  `FUNCTION` f`(`x,c`:`$D{\times}D$`'):`R `WHERE` $I[x] \wedge c{=}E[x]$ `RETURNS` y `SUCH THAT` $O[x,y]$

  $\equiv$ $....E[x]....f'(\,t[x],\, E[t[x]]\,)..$

- **Simplifikation mit Gleichung $c = E[x]$**

  – Transformiere Ausdrücke der Form $E[\,g[x]\,]$ in die Form $g'[\,E[x]\,]$

  – Ersetze alle Vorkommen von $E[x]$ durch c

- **Benutzerinteraktion:**

  – Auswahl des zu ersetzenden Teilausdrucks $E[x]$ in $f(x)$

Endliche Differenzierung von $\texttt{Costas}_{aux}$

```
FUNCTION Costas_aux (n,s:ℤ×Seq(ℤ))
  WHERE n≥1 ∧ range(s)⊆{1..n} ∧ nodups(s) ∧ ∀j<|s|.nodups(dtrow(s,j))
  RETURNS {p:Seq(ℤ)|perm(p,{1..n})∧ s⊑p∧ ∀j<|p|.nodups(dtrow(p,j))}
≡      if {1..n}\range(s)=∅ then {s} else ∅
   ∪ ⋃{Costas_gs(n,s·i)|i∈{1..n}\range(s) ∧ ∀j<|s|. s_{|s·i|-j} -i ∉ dtrow(s,j)}
```

```
FUNCTION Costas_aux(n,s,pool,ssize:ℤ×Seq(ℤ)×Seq(ℤ)×ℤ)
  WHERE n≥1 ∧ range(s)⊆{1..n} ∧ nodups(s) ∧ ∀j<|s|.nodups(dtrow(s,j))
            ∧ pool = {1..n}\range(s) ∧ ssize=|s|
  RETURNS {p:Seq(ℤ)|perm(p,{1..n})∧ s⊑p∧ ∀j<|p|.nodups(dtrow(p,j))}
≡     if pool=∅ then {s} else ∅
  ∪ ⋃{Costas_gs(n,s·i,pool\{i},ssize+1)|i∈pool
                                  ∧∀j<ssize. s_{ssize+1-j} -i ∉ dtrow(s,j)}
```

Modifizierter Aufruf aus Hauptfunktion

```
FUNCTION Costas (n:ℤ) WHERE n≥1
  RETURNS {p:Seq(ℤ)|perm(p,{1..n})∧ ∀j<|p|.nodups(dtrow(p,j))}
≡ Costas_gs(n,[],{1..n},0)
```

Fallanalyse

- **Separate Analyse von Einzelfällen**
  - Auswertung von Tests aus anderen Programmteilen
  - Zweck: globale Vereinfachung durch lokale Einzelanalyse
  - Formal: Erzeugung eines Kontexts + CD-Simplifikation

- **Kontexterzeugung mit Prädikat P**
  - Ersetze Ausdruck $E[x]$ durch `if` $P[x]$ `then` $E[x]$ `else` $E[x]$
  - Vereinfache $E[x]$ in den entsprechenden Kontexten $P[x]$ und $\neg P[x]$
  - Distributiere `if` $P[x]$ `then`...`else`...
    über Ausdrücke außerhalb von $E[x]$

- **Benutzerinteraktion:**
  - Auswahl des zu ersetzenden Teilausdrucks $E[x]$
  - Auswahl des Prädikats $P[x]$
  - Auslösung der nachfolgenden Simplifikationen

Fallanalyse in $\texttt{Costas}_{aux}$

```
FUNCTION Costas_aux(n,s,pool,ssize:ℤ×Seq(ℤ)×Seq(ℤ)×ℤ)
  WHERE n≥1 ∧ range(s)⊆{1..n} ∧ nodups(s) ∧ ∀j<|s|.nodups(dtrow(s,j))
              ∧ pool = {1..n}\range(s) ∧ ssize=|s|
  RETURNS {p:Seq(ℤ) | perm(p,{1..n}) ∧ s⊑p ∧ ∀j<|p|.nodups(dtrow(p,j))}
≡      if pool=∅ then {s} else ∅
  ∪ ⋃{ Costas_gs(n,s·i,pool\{i},ssize+1) | i ∈ pool
                                          ∧ ∀j<ssize. s_{ssize+1−j} −i ∉ dtrow(s,j)}
```

```
FUNCTION Costas_aux(n,s,pool,ssize:ℤ×Seq(ℤ)×Seq(ℤ)×ℤ)
  WHERE n≥1 ∧ range(s)⊆{1..n} ∧ nodups(s) ∧ ∀j<|s|.nodups(dtrow(s,j))
              ∧ pool = {1..n}\range(s) ∧ ssize=|s|
  RETURNS {p:Seq(ℤ) | perm(p,{1..n}) ∧ s⊑p ∧ ∀j<|p|.nodups(dtrow(p,j))}
≡   if pool=∅ then {s}
              else ⋃{ Costas_gs(n,s·i,pool\{i},ssize+1) | i ∈ pool
                                          ∧ ∀j<ssize. s_{ssize+1−j} −i ∉ dtrow(s,j)}
```

Datentypverfeinerung

Ersetze abstrakte Datentypen durch effiziente konkrete Implementierung

- **Endliche Mengen**
  - Listen: Standardimplementierung
  - Bitvektor: Mengen über endlichem Domain
  - Charakteristische Funktion: effiziente Elementrelation/Einfügen/Löschen
- **Folgen**
  - Verkettete Liste: Standardimplementierung
  - Umgekehrt verkettete Liste: gut für append-Operation
- **Benutzerinteraktion:**
  - System stellt Auswahl von Implementierungen bereit
  - Benutzer wählt Nichtstandard-Implementierung einzeln für jede Variable
  - System ersetzt abstrakte Notation durch konkrete Implementierung und fügt ggf. Konversionen ein

DATENTYPVERFEINERUNG FÜR $\texttt{Costas}_{aux}$

$$
\begin{array}{l}
\texttt{FUNCTION Costas}_{aux}\texttt{(n,s,pool,ssize:}\mathbb{Z}\times\texttt{Seq(}\mathbb{Z}\texttt{)}\times\texttt{Seq(}\mathbb{Z}\texttt{)}\times\mathbb{Z}\texttt{)} \\
\quad \texttt{WHERE n}\geq\texttt{1} \;\wedge\; \texttt{range(s)}\subseteq\{\texttt{1..n}\} \;\wedge\; \texttt{nodups(s)} \;\wedge\; \forall\texttt{j<|s|.nodups(dtrow(s,j))} \\
\qquad\qquad \wedge\; \texttt{pool = }\{\texttt{1..n}\}\backslash\texttt{range(s)} \;\wedge\; \texttt{ssize=|s|} \\
\quad \texttt{RETURNS } \{\texttt{p:Seq(}\mathbb{Z}\texttt{) | perm(p,}\{\texttt{1..n}\}\texttt{)} \wedge \texttt{ s}\sqsubseteq\texttt{p} \wedge \forall\texttt{j<|p|.nodups(dtrow(p,j))}\} \\
\equiv \quad \texttt{if pool=}\emptyset\texttt{ then } \{\texttt{s}\} \\
\qquad\qquad \texttt{else } \bigcup\{\,\texttt{Costas}_{gs}\texttt{(n,s}\cdot\texttt{i,pool}\backslash\{\texttt{i}\}\texttt{,ssize+1) | i}\in\texttt{pool} \\
\qquad\qquad\qquad\qquad\qquad \wedge\; \forall\texttt{j<ssize. s}_{\texttt{ssize+1-j}}\texttt{-i} \notin \texttt{dtrow(s,j)}\}
\end{array}
$$

$\texttt{n}$: $\mathbb{Z}$ $\mapsto$ Standardimplementierung positiver ganzen Zahlen

$\texttt{s}$: $\texttt{Seq(}\mathbb{Z}\texttt{)}$, Elemente werden hinten angehängt $\mapsto$ umgekehrt verkettete Liste

$\texttt{pool}$: $\texttt{Set(}\mathbb{Z}\texttt{)}$: Elemente werden aus fester Menge entnommen $\mapsto$ Bitvektor

$\texttt{ssize}$: $\mathbb{Z}$ $\mapsto$ Standardimplementierung positiver ganzen Zahlen