

Automatisierte Logik und Programmierung

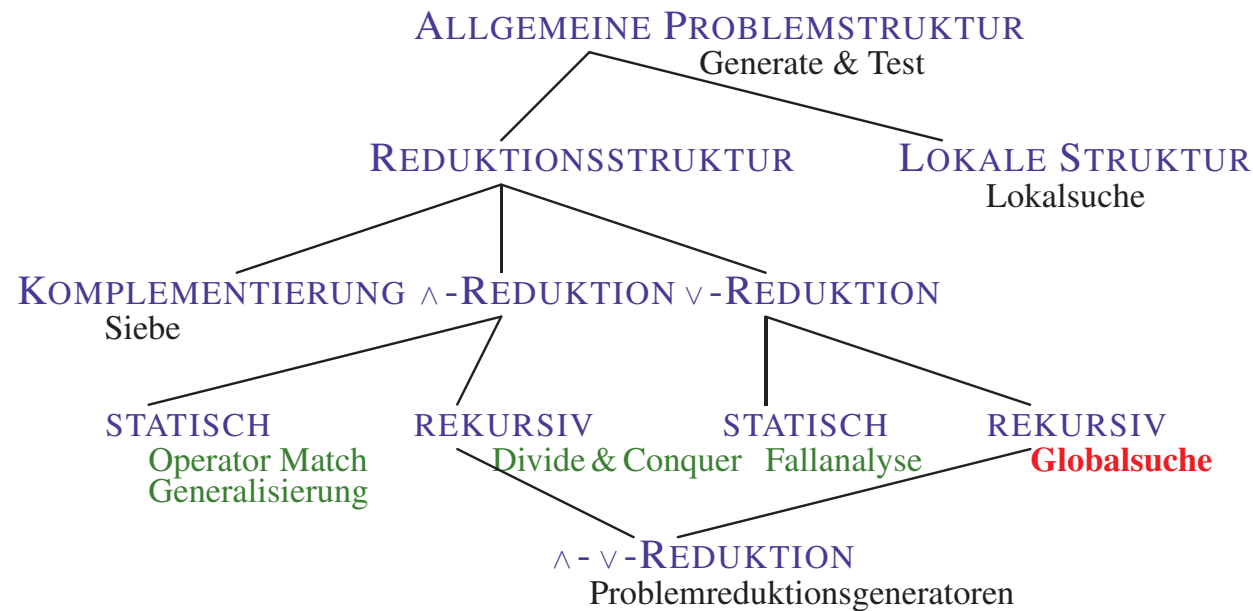
Einheit 20

Globalsuch-Algorithmen



1. Algorithmenschema
2. Korrektheit
3. Wissensbasierte Unterstützung
4. Synthesestrategie
5. Integration in Beweisumgebung

GLOBALSUCHALGORITHMEN



- **Bestimmung aller Lösungen eines Problems**
 - Aufzählen von Kandidaten
 - Eliminieren von Kandidaten, die keine Lösungen darstellen
 - Verallgemeinert Backtracking, Binärsuche, Branch & Bound ...
- **Rekursive \vee -Reduktion des Problems**
 - Gesamtlösung ist Vereinigung unabhängiger Teillösungen
 - Gut geeignet für Parallelverarbeitung
 - Auch für sequentielle Suche nach erstmöglicher Lösung geeignet

EIN TYPISCHER GLOBALSUCHALGORITHMUS

- **Suche alle Indizes eines Wertes k in einer geordneten Liste L**

FUNCTION `osearch`($L, k: \text{Seq}(\mathbb{Z}) \times \mathbb{Z}$) WHERE $L \neq [] \wedge \text{ordered}(L)$
RETURNS $\{ i: \mathbb{N} \mid i \in \{1..|L|\} \wedge L_i = k \}$

- **Binäre Suche und Aufsammeln von Lösungen**

- Spalte Liste L in zwei Teile $[L_i \mid 1 \leq i \leq m]$ und $[L_i \mid m < i \leq |L|]$
- Durchsuche beide Hälften und vereinige jeweilige Lösungsmengen

- **Verwalte Indexgrenzen anstelle der tatsächlichen Teillisten**

- Signifikante Reduktion der zu verwaltenden Daten
- Grenzen l und r der Indexmengen sind **Repräsentanten** des Suchraums
- Repräsentanten sind nur dann **sinnvoll** wenn $1 \leq l \leq r \leq |L|$

- **Verwende Hilfsfunktion `aux` für rekursive Breitensuche**

FUNCTION `aux`($L, k, l, r: \text{Seq}(\mathbb{Z}) \times \mathbb{Z} \times \mathbb{N} \times \mathbb{N}$)
WHERE $L \neq [] \wedge \text{ordered}(L) \wedge 1 \leq l \leq r \leq |L|$
RETURNS $\{ i: \mathbb{N} \mid i \in \{1..r\} \wedge L_i = k \}$
 \equiv if $l=r$ then if $L_l=k$ then $\{l\}$ else \emptyset
else let $m=(l+r)/2$ in `aux`(L, k, l, m) \cup `aux`($L, k, m+1, r$)

- **Initialisiere Hilfsfunktion mit gesamter Liste**

- `osearch`(L, k) \equiv `aux`($L, k, 1, |L|$)

OPTIMIERUNG DES GRUNDALGORITHMUS

- **Eliminiere Hilfsfunktionsaufrufe, wenn keine Lösung möglich**
 - Suche berücksichtigt nicht, daß Liste geordnet ist (linearer Algorithmus)
 - Suchraum $\{i..j\}$ ohne Lösung, falls $L_i > k$ oder $L_j < k$
 - Ergänze **Filter** $L_i \leq k \leq L_j$ zum rekursivem Aufruf

```
FUNCTION aux(L,k,l,r:Seq(Z)×Z×N×N)
  WHERE L≠[] ∧ ordered(L) ∧ 1≤l≤r≤|L|
  RETURNS {i:N | i∈{1..r} ∧ Li=k}
≡ if l=r then if Ll=k then {l} else ∅
   else let m=(l+r)/2
        in if Lm<k then aux(L,k,m+1,r)
           elseif Lm+1>k then aux(L,k,l,m)
           else aux(L,k,l,m) ∪ aux(L,k,m+1,r)
```

- Algorithmus hat nur noch logarithmische Laufzeit

- **Integriere Filter in Initialaufruf der Hilfsfunktion**

```
FUNCTION osearch(L,k:Seq(Z)×Z) WHERE L≠[] ∧ ordered(L)
  RETURNS {i:N | i∈{1..|L|} ∧ Li=k}
≡ if L1≤k≤L|L| then aux(L,k,1,|L|) else ∅
```

GLOBALSUCHE: GENERELLE IDEE

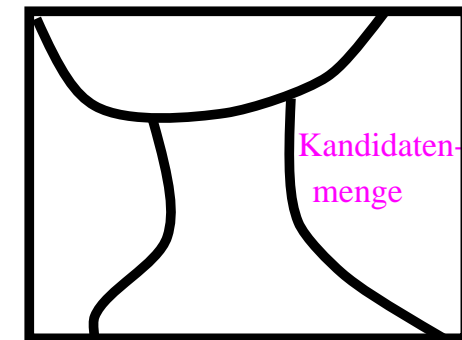
Durchsuchen eines gesamten Bildbereichs

● Suche von außen

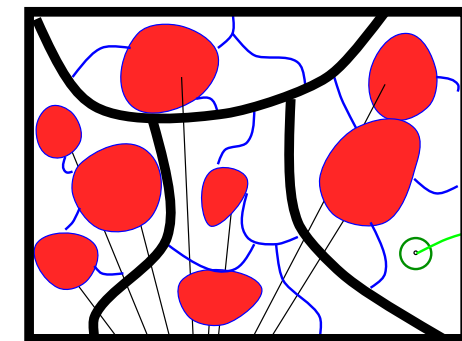
- **Global:** Untersuchung ganzer Mengen von Lösungskandidaten
- Wiederholtes **Aufteilen** von Kandidatenmengen
- **Extraktion** von tatsächlichen Lösungen
- **Elimination** von Kandidatenmengen ohne Lösung

● Repräsentanten erforderlich

- Verarbeitung der Mengen selbst zu aufwendig
- Codiere Kandidatenmengen durch **Deskriptoren**
- Simuliere **Aufteilen** und **Filtern** auf Deskriptoren
- Notwendige Informationen bei der Spezifikation:
 - Wann ist ein Deskriptor eine **sinnvolle Beschreibung** einer Menge?
 - Wie beschreibt man **Zugehörigkeit zur Menge** mittels Deskriptoren?



Zerteilen
von Kandidaten



Eliminierte Kandidaten

EINHEITLICHE NOTATION FÜR GLOBALSUCHALGORITHMEN

```
FUNCTION aux(L,k,l,r:Seq( $\mathbb{Z}$ ) $\times\mathbb{Z}\times\mathbb{N}\times\mathbb{N}$ ) ...  
≡ if l=r then if  $L_l=k$  then {l} else  $\emptyset$   
   else let m=(l+r)/2  
       in if  $L_m < k$  then aux(L,k,m+1,r)  
          elseif  $L_{m+1} > k$  then aux(L,k,l,m)  
          else aux(L,k,l,m)  $\cup$  aux(L,k,m+1,r)
```

- **Darstellung nur für binäre Aufspaltung des Suchraums geeignet**
 - Allgemeiner Fall ist Vereinigung vieler Lösungsmengen
 - Allgemeine Mengenschreibweise ist geeigneter als Darstellung
 - Indexgrenzen (i, j) werden aus Aufspaltungsmenge ausgewählt
 - Indexgrenzen werden ausgefiltert, wenn $L_i > k$ oder $L_j < k$
 - Direkte Lösung wird durch Extraktion aus $\{l..r\}$ erzeugt
 - Gesamtlösung ist Vereinigung aller Einzellösungsmengen
- **Endform: wohlstrukturierter mathematischer Algorithmus**

```
FUNCTION aux(L,k,l,r:Seq( $\mathbb{Z}$ ) $\times\mathbb{Z}\times\mathbb{N}\times\mathbb{N}$ ) ...  
≡ {i |  $i \in \{l..r\} \wedge L_i = k$ }  
    $\cup \bigcup \{ \textit{aux}(L,k,n,m) \mid$   
            $(n,m) \in \{(l, (l+r)/2), ((l+r)/2+1, r) \mid l < r\} \wedge L_n \leq k \leq L_m \}$ 
```


KORREKTHEIT DES GLOBALSUCH-SCHEMAS

FUNCTION $f(x:D)$ WHERE $I[x]$ RETURNS $\{y:R \mid O[x,y]\}$
 \equiv let rec $aux(x,s) = \{z \mid z \in ext[s] \wedge O[x,z]\} \cup \bigcup \{aux(x,t) \mid t \in split[x,s] \wedge \Phi[x,t]\}$
in if $\Phi[x,s_0(x)]$ then $aux(x,s_0(x))$ else \emptyset

ist korrekt, wenn 6 Axiome erfüllt sind

1. Initialdeskriptor ist sinnvoll für zulässige Eingaben

$$I[x] \Rightarrow J[x, s_0(x)]$$

2. Splitting erhält sinnvoller Deskriptoren

$$I[x] \wedge J[x, s] \Rightarrow \forall t \in split[x, s]. J[x, t]$$

3. Initialdeskriptor enthält alle Lösungen

$$I[x] \wedge O[x, z] \Rightarrow sat[z, s_0(x)]$$

4. Filter ist notwendig (keine Lösung wird eliminiert)

$$I[x] \wedge J[x, s] \Rightarrow (\Phi[x, s] \Leftarrow \exists z:R. sat[z, s] \wedge O[x, z])$$

5. Alle Lösungen in endlich vielen Schritten extrahierbar

$$I[x] \wedge O[x, z] \wedge J[x, s] \Rightarrow (sat[z, s] \Leftrightarrow \exists k:\mathbb{N}. \exists t \in split_{\Phi}^k[x, s]. z \in ext[t])$$

6. Splitting (mit Filterung) ist wohlfundiert

$$I[x] \wedge J[x, s] \Rightarrow \exists k:\mathbb{N}. split_{\Phi}^k[x, s] = \emptyset$$

GLOBALSUCH-SCHEMA: KORREKTHEITSBEWEIF

- **Abspalten und explizite Spezifikation der Hilfsfunktion *aux***

FUNCTION $f(x:D)$ WHERE $I[x]$ RETURNS $\{y:R \mid O[x,y]\}$

\equiv if $\Phi[x, s_0(x)]$ then $aux(x, s_0(x))$ else \emptyset

FUNCTION $aux(x,s:D \times S)$ WHERE $I[x] \wedge J[x,s] \wedge \Phi[x,s]$

RETURNS $\{y:R \mid O[x,y] \wedge sat[y,s]\}$

$\equiv \{z \mid z \in ext[s] \wedge O[x,z]\} \cup \bigcup \{aux(x,t) \mid t \in split[x,s] \wedge \Phi[x,t]\}$

- **Korrektheit von f folgt aus der von aux mit Axiom 1, 3 & 4**

– Für den Startwert $s_0(x)$ gilt $J[x, s_0(x)]$ (Axiom 1)

– Aus $I[x]$ folgt $\{y:R \mid O[x,y] \wedge sat[y, s_0(x)]\} = \{y:R \mid O[x,y]\}$ (Axiom 3)

– Aus $\{y:R \mid O[x,y] \wedge sat[y, s_0(x)]\} \neq \emptyset$ folgt $\Phi[x, s_0(x)]$ (Axiom 4)

- **Partielle Korrektheit von aux folgt aus Axiom 5 & 2 und folgendem**

Lemma: Hält $aux[x,s]$ nach i Schritten an ($split_{\Phi}^i[x,s] = \emptyset$), so ist das Resultat

$$\bigcup \{ \{z \mid z \in ext[t] \wedge O[x,z]\} \mid t \in \bigcup \{split_{\Phi}^j[x,s] \mid 0 \leq j < i\} \}$$

(Menge der aus Deskriptoren extrahierbaren Lösungen, die zu einem $split_{\Phi}^j[x,s]$ gehören)

$$split_{\Phi}^k[x,s] \equiv \text{if } k=0 \text{ then } \{s\} \text{ else } \bigcup \{split_{\Phi}^{k-1}[x,t] \mid t \in split[x,s] \wedge \Phi[x,t]\}$$

Beweis: Induktion über i , Auffalten der Rekursion, Standardlemmata

- **Terminierung von aux folgt aus Axiom 6**

Es sind 7 neue Komponenten zu bestimmen

- **Zusätzliche Datentypen**

- Raum der Deskriptoren für den Suchraum $\text{Set}(R)$ (S)

- **Zusätzliche Eingabebedingungen für Hilfsfunktion**

- Charakterisierung sinnvoller Deskriptoren (J)

- Zusammenhang zwischen Deskriptoren und Suchraum (sat)

- **Komponenten des Algorithmus**

- $s_0: D \rightarrow S$

Axiom 1,3

- $split: D \times S \rightarrow \text{Set}(S)$

Axiom 2,5,6

- $\Phi: D \times S \rightarrow \mathbb{B}$

Axiom 4,5,6

- $ext: S \rightarrow \text{Set}(R)$

Axiom 5

Lösungen müssen Beweis der Axiome 1–6 liefern

Kompakte Bezeichnung: wohlfundierte Globalsuchtheorie

- Struktur “ $\mathcal{G} = (D, R, I, O, S, J, s_0, sat, split, \Phi, ext)$ ”, die alle 6 Axiome erfüllt

SYNTHESE EINES GLOBALSUCH-ALGORITHMUS

INDIZES EINES WERTES IN EINER GEORDNETEN LISTE

Spezifikation: FUNCTION `osearch(L, k: Seq(\mathbb{Z}) \times \mathbb{Z})`
WHERE `L \neq [] \wedge ordered(L)`
RETURNS `{ i: \mathbb{N} | i \in {1..|L|} \wedge Li=k }`

Ziel: Bestimme Komponenten des Globalsuchschemas für `osearch`

1. Zahlenmengen sind Vereinigungen von Intervallen

– Intervalle $\{l..r\}$ sind durch Zahlenpaare (l, r) beschreibbar

Wähle $S \equiv \mathbb{N} \times \mathbb{N}$

– sat beschreibt Elemente von $\{1..r\}$, also $sat[i, (l, r)] \equiv i \in \{l..r\}$

– Sinnvolle Indexmengen sind nicht leer: $J[(L, k), (l, r)] \equiv 1 \leq l \leq r \leq |L|$

– Intervalle müssen k enthalten: $\Phi(L, k), (l, r) \equiv L_l \leq k \leq L_r$ Axiom 4

2. Das Anfangsintervall muß alle möglichen Indizes umfassen

– $s_0[(L, k)] \equiv (1, |L|)$ liefert Axiom 1,3

3. Intervalle lassen sich binär aufspalten liefert Axiom 2,6

– $split[(L, k), (l, r)] \equiv \text{if } l < r \text{ then } \{(l, (l+r)/2), ((l+r)/2+1, r)\} \text{ else } \emptyset$

4. Aus einelementigen Intervallen kann man extrahieren liefert Axiom 5

– $ext[(l, r)] \equiv \text{if } l=r \text{ then } \{l\} \text{ else } \emptyset$

ANALYSE LIEFERT GLOBALSUCHTHEORIE FÜR osearch

• Theorie $gs_osearch$ faßt alle Komponenten zusammen

D	\mapsto	$Seq(\mathbb{N}) \times \mathbb{N}$
R	\mapsto	\mathbb{N}
I	\mapsto	$\lambda L, k. L \neq [] \wedge ordered(L)$
O	\mapsto	$\lambda L, k, i. i \in \{1.. L \} \wedge L_i = k$
S	\mapsto	$\mathbb{N} \times \mathbb{N}$
J	\mapsto	$\lambda L, k, l, r. 1 \leq l \leq r \leq L $
s_0	\mapsto	$\lambda L, k. (1, L)$
sat	\mapsto	$\lambda i, l, r. i \in \{1..r\}$
$split$	\mapsto	$\lambda L, k, l, r. \text{if } l < r \text{ then } \{(1, (l+r)/2), ((l+r)/2+1, r)\} \text{ else } \emptyset$
Φ	\mapsto	$\lambda L, k, l, r. L_l \leq k \leq L_r$
ext	\mapsto	$\lambda l, r. \text{if } l = r \text{ then } \{1\} \text{ else } \emptyset$

• Alle 6 Axiome sind erfüllt

1. $L \neq [] \wedge ordered(L) \Rightarrow 1 \leq l \leq |L| \leq |L|$
2. $\dots \wedge 1 \leq l \leq r \leq |L| \Rightarrow \forall (x, y) \in split[L, k, l, r]. 1 \leq x \leq y \leq |L|$
3. $\dots \wedge i \in \{1..|L|\} \wedge L_i = k \Rightarrow i \in \{1..|L|\}$
4. $\dots \wedge 1 \leq l \leq r \leq |L| \Rightarrow L_l \leq k \leq L_r \Leftrightarrow \exists z: \mathbb{N}. z \in \{1..r\} \wedge z \in \{1..|L|\} \wedge L_z = k$
5. $\dots \wedge 1 \leq l \leq r \leq |L| \wedge i \in \{1..|L|\} \wedge L_i = k \Rightarrow$
 $i \in \{1..r\} \Leftrightarrow \exists k: \mathbb{N}. \exists (x, y) \in split_{\Phi}^k[L, k, l, r]. i \in (\text{if } x = y \text{ then } \{x\} \text{ else } \emptyset)$
6. $\dots \wedge 1 \leq l \leq r \leq |L| \Rightarrow \exists k: \mathbb{N}. split_{\Phi}^k[L, k, l, r] = \emptyset$

SCHEMATISCHER GLOBALSUCHALGORITHMUS FÜR osearch

```
FUNCTION osearch(L,k:Seq( $\mathbb{Z}$ ) $\times\mathbb{Z}$ ) WHERE L $\neq$ []  $\wedge$  ordered(L)
  RETURNS { i: $\mathbb{N}$  | i $\in$ {1..|L|}  $\wedge$  Li=k }
 $\equiv$  let rec aux(L,k,l,r)
  = { z | z $\in$ (if l=r then {1} else  $\emptyset$ )  $\wedge$  z $\in$ {1..|L|}  $\wedge$  Lz=k }
     $\cup \cup$  { aux(L,k,n,m) |
      (n,m) $\in$ (if l<r then {(1,(l+r)/2), ((l+r)/2+1,r)}
        else  $\emptyset$ )  $\wedge$  Ln $\leq$ k $\leq$ Lm }
  in if L1 $\leq$ k $\leq$ L|L| then aux(L,k,1,|L|) else  $\emptyset$ 
```

Nach (kontextabhängigen) Simplifikationen

```
FUNCTION osearch(L,k:Seq( $\mathbb{Z}$ ) $\times\mathbb{Z}$ ) WHERE L $\neq$ []  $\wedge$  ordered(L)
  RETURNS { i: $\mathbb{N}$  | i $\in$ {1..|L|}  $\wedge$  Li=k }
 $\equiv$  let rec aux(L,k,l,r)
  = if l=r then if Ll=k then {1} else  $\emptyset$ 
    else let m = (l+r)/2 in
      (if k $\leq$ Lm then aux(L,k,l,m) else  $\emptyset$ )
       $\cup$  (if Lm+1 $\leq$ k then aux(L,k,m+1,r) else  $\emptyset$ )
  in if L1 $\leq$ k $\leq$ L|L| then aux(L,k,1,|L|) else  $\emptyset$ 
```

WIE ERZEUGT MAN GLOBALSUCH-THEORIEN?

- **Viele Komponenten basieren auf Standard-Suchstrukturen**
 - Deskriptorenraum S , Aufspaltung $split$, Extraktion ext sowie die Prädikate J und sat hängen im wesentlichen nur vom Suchraum R ab
 - Synthese benötigt formales Wissen über solche Suchstrukturen
- **Filter und Initialdeskriptoren sind aus Axiomen herleitbar**
 - Aber auch hier geht Wissen über mögliche Initialsuchräume für s_0 ein, das auf die spezielle Situation angepaßt wird
- **Wohlfundiertheit ist nicht automatisch zu beweisen**
 - Beweis für Terminierung von $split_{\Phi}$ muß im Voraus geführt werden
 - Verlangt Wissen über “Mindestfilter”, die eine Terminierung garantieren

Spezialisiere generische Globalsuch-Theorien auf Problem

SPEZIALISIERUNG EINER SUCHSTRUKTUR FÜR osearch

• (Terminierende) Standard-Binärsuche auf Integer-Intervallen

– Durchsucht Elemente eines Intervalls $\{m..n\}$ durch binäres Aufspalten

$$D \mapsto \mathbb{Z} \times \mathbb{Z}$$

$$R \mapsto \mathbb{Z}$$

$$I \mapsto \lambda m, n. m \leq n$$

$$O \mapsto \lambda m, n, k. k \in \{m..n\}$$

$$S \mapsto \mathbb{Z} \times \mathbb{Z}$$

$$J \mapsto \lambda m, n, i, j. \{i..j\} \subseteq \{m..n\}$$

$$s_0 \mapsto \lambda m, n. (m, n)$$

$$sat \mapsto \lambda k, i, j. k \in \{i..j\}$$

$$split \mapsto \lambda m, n, i, j. \text{if } i < j \text{ then } \{(i, (i+j)/2), ((i+j)/2, j)\} \text{ else } \emptyset$$

$$\Phi \mapsto \lambda m, n, i, j. \{i..j\} \neq \emptyset$$

$$ext \mapsto \lambda i, j. \text{if } i = j \text{ then } \{i\} \text{ else } \emptyset$$

• GS-Theorie muß auf Spezifikation von osearch angepaßt werden

$$D \mapsto \text{Seq}(\mathbb{N}) \times \mathbb{N}$$

$$R \mapsto \mathbb{N}$$

$$I \mapsto \lambda L, k. L \neq [] \wedge \text{ordered}(L)$$

$$O \mapsto \lambda L, k, i. i \in \{1..|L|\} \wedge L_i = k$$

– Ausgabebedingung verwendet $\{1..|L|\}$ statt $\{m..n\}$, Suchraum ist \mathbb{N} anstelle von \mathbb{Z}

– Anpassung liefert $S \equiv \mathbb{N} \times \mathbb{N}$, $J[(L, k), (l, r)] \equiv 1 \leq l \leq r \leq |L|$, $s_0[(L, k)] \equiv (1, |L|)$

$sat, split, ext$ bleiben unverändert, $\Phi(L, k), (l, r) \equiv L_l \leq k \leq L_r$ folgt aus Axiom 4

STRATEGIE: SPEZIALISIERE STANDARD-SUCHSTRUKTUREN

● Wissensbank enthält relevantes Programmierwissen

Globalsuchtheorie: allgemeine Suchstruktur für R

- Vorgefertigte Zerlegungsstruktur, die Axiome 1–5 erfüllt
- Formalisiert als Objekt $\mathcal{G} = (D, R, I, O, S, J, s_\emptyset, sat, split, True, ext)$
- Speichere (mehrere) Globalsuchtheorien für alle Grunddatentypen

Wohlfundiertheitsfilter garantieren Terminierung von $split_\Phi$

- Erforderlich, wenn GS-Theorie \mathcal{G} Axiom 6 nicht erfüllt

Oft gibt es dann mehrere mögliche wf-Filter für \mathcal{G}

↪ Ax 6

● Unterstützende Techniken der Globalsuch-Strategie

Spezialisierungsmechanismus für GS-Theorien

↪ §18, Folie 20

- Wähle \mathcal{G} passend zum Bildbereich der Spezifikation $spec = (D, R, I, O)$
- Beweise $spec \ll spec_{\mathcal{G}}$ und extrahiere Substitution $\vartheta: D \rightarrow D_{\mathcal{G}}$
- Modifiziere \mathcal{G} mit ϑ zu wohlfundierter Globalsuchtheorie für $spec$

Vorwärtsinferenz (heuristisch)

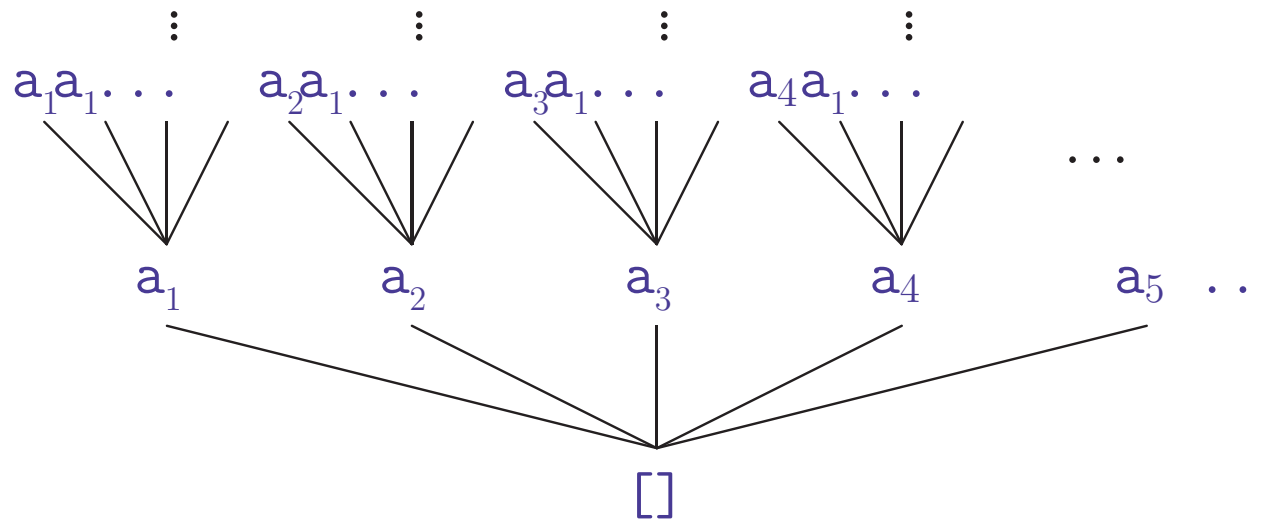
- Wähle Φ für \mathcal{G} , so daß Axiom 4 für Φ_ϑ beweisbar

↪ Notwendigkeit

- Verfeinere Φ_ϑ um weitere Folgerungen von $sat \wedge O$

↪ Effizienzsteigerung

Suche $\hat{=}$ Aufzählung der Präfixe einer Liste L



- **Deskriptoren: gemeinsamer Präfix s**
- **Initialdeskriptor: leerer Präfix**
- **Splitting: Verlängern des Präfix**
- **Extraktion: Gesamter Präfix**
- **Sinnvoll: nur Elemente aus M**

$$\text{sat}[L, s] \equiv s \sqsubseteq L$$

$$s_0(M) \equiv [\]$$

$$\text{split}[M, s] \equiv \{s \cdot a \mid a \in M\}$$

$$\text{ext}[s] \equiv \{s\}$$

$$J[M, s] \equiv s' \subseteq' M$$

FORMALE GS-THEORIE FÜR LISTEN ÜBER $M \subseteq \alpha$

- Deskriptoren: gemeinsamer Präfix s $sat[L, s] \equiv s \sqsubseteq L$
- Initialdeskriptor: leerer Präfix $s_0(M) \equiv []$
- Splitting: Verlängern des Präfix $split[M, s] \equiv \{s \cdot a \mid a \in M\}$
- Extraktion: Gesamter Präfix $ext[s] \equiv \{s\}$
- Sinnvoll: nur Elemente aus M $J[M, s] \equiv s \text{ '}\subseteq\text{' } M$

Darstellung als formales Objekt der Wissensbank

$gs_seq_set(\alpha)$	\equiv	D	\mapsto	$Set(\alpha)$
		R	\mapsto	$Seq(\alpha)$
		I	\mapsto	$\lambda M. true$
		O	\mapsto	$\lambda M, L. range(L) \subseteq M$
		S	\mapsto	$Seq(\alpha)$
		J	\mapsto	$\lambda M, s. range(s) \subseteq M$
		s_0	\mapsto	$\lambda M. []$
		sat	\mapsto	$\lambda L, s. s \sqsubseteq L$
		$split$	\mapsto	$\lambda M, s. \{s \cdot a \mid a \in M\}$
		Φ	\mapsto	$\lambda M, s. true$
		ext	\mapsto	$\lambda s. \{s\}$

Axiome 1–5 sind formal beweisbar, Axiom 6 ist nicht erfüllt

WOHLFUNDIERTHEITSFILTER FÜR $gs_seq_set(\alpha)$

- $\Phi_1[M, s] \equiv |s| \leq k$
 - Filter testet absolute Längenbegrenzung der Deskriptorfolge
 - Einfacher, schnell auszuführender Test
 - Filter garantiert Terminierung nach k Schritten, Baumgröße $|M|^k$
- $\Phi_2[M, s] \equiv |s| \leq k * |M|$
 - Filter testet Längenbegrenzung relativ zur Größe von M
 - Einfacher, schnell auszuführender Test
 - Terminierung nach $k * |M|$ Schritten, Baumgröße $|M|^{k * |M|}$
- $\Phi_3[M, s] \equiv \text{nodups}(s)$
 - Filter testet Deskriptorfolge auf Duplikate
 - Test ist aufwendiger und sollte optimiert werden
 - Terminierung nach $|M|$ Schritten, Baumgröße $|M|!$

Axiom 6 ist für jeden dieser Filter formal beweisbar

ENTWICKLUNG EINES GLOBALSUCH-ALGORITHMUS FÜR DAS COSTAS-ARRAYS PROBLEM

- **Costas Array** der Größe n :

- Permutation von $\{1..n\}$ ohne Duplikate in Zeilen der Differenztabelle

2	4	1	6	5	3
-2	3	-5	1	2	
1	-2	-4	3		
-4	-1	-2			
-3	1				
-1					

- **Ziel: Berechne alle Costas Arrays der Größe n**

- **Formalisierung vorkommender Begriffe:**

$dtrow(L, j) \equiv [L_i - L_{i+j} \mid i \in [1..|L|-j]]$

$perm(L, S) \equiv nodups(L) \wedge range(L) = S$

- **Spezifikation des Problems**

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

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

ERFORDERLICHE ZUSÄTZLICHE MATHEMATISCHE GESETZE FÜR LÖSUNG DES COSTAS ARRAYS PROBLEMS

● Gesetze von Differenzentafeln

$\forall L, L' : \text{Seq}(\mathbb{Z}) . \forall i : \mathbb{Z} . \forall j : \mathbb{N} .$

1. $\text{dtrow}([], j) = []$
 2. $j \leq |L| \Rightarrow \text{dtrow}(i.L, j) = (i - L[j]) . \text{dtrow}(L, j)$
 3. $j \neq 0 \Rightarrow \text{dtrow}([i], j) = []$
 4. $L \sqsubseteq L' \Rightarrow \text{dtrow}(L, j) \sqsubseteq \text{dtrow}(L', j)$
 5. $j \geq |L| \Rightarrow \text{dtrow}(L, j) = []$
 6. $j \leq |L| \Rightarrow \text{dtrow}(L \cdot i, j) = \text{dtrow}(L, j) \cdot (L[|L|+1-j] - i)$
- ⋮

● Gesetze duplikatenfreier Listen

$\forall L, L' : \text{Seq}(\mathbb{Z}) . \forall i : \mathbb{Z} .$

1. $\text{nodups}([])$
 2. $\text{nodups}(i.L) = \text{nodups}(L) \wedge i \notin L$
 3. $\text{nodups}(L \cdot i) = \text{nodups}(L) \wedge i \notin L$
 4. $L \sqsubseteq L' \Rightarrow \text{nodups}(L') \Rightarrow \text{nodups}(L)$
- ⋮

SPEZIALISIERE $gs_seq_set(\mathbb{Z}) / \Phi_3$ AUF COSTAS-ARRAYS

```

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

D	\mapsto	$Set(\alpha)$
R	\mapsto	$Seq(\alpha)$
I	\mapsto	$\lambda M. true$
O	\mapsto	$\lambda M, L. range(L) \subseteq M$
S	\mapsto	$Seq(\alpha)$
J	\mapsto	$\lambda M, s. range(s) \subseteq M$
s_0	\mapsto	$\lambda M. []$
sat	\mapsto	$\lambda L, s. s \subseteq L$
$split$	\mapsto	$\lambda M, s. \{s \cdot a \mid a \in M\}$
ext	\mapsto	$\lambda s. \{s\}$

1. Bildbereich stimmt mit $R_G = Seq(\mathbb{Z})$ überein
2. Eingabebereiche \mathbb{Z} , $D_G = Set(\mathbb{Z})$ sind anzupassen
3. Keine Eingabebedingung zu prüfen: $I_G(M) = true$
4. Zu zeigen ist also:

$\forall n: \mathbb{Z}. n \geq 1 \Rightarrow \exists M: Set(\mathbb{Z}). \forall p: Seq(\mathbb{Z}). O(n, p) \Rightarrow range(p) \subseteq M$

- Heuristik: Suche Folgerungen von $O(n, p)$, in denen $range(p)$ vorkommt
- Auffalten von $perm$ liefert: $perm(p, \{1..n\}) \Rightarrow range(p) \subseteq \{1..n\}$
- Wähle $M \equiv \{1..n\}$ und extrahiere $\vartheta = \lambda n. \{1..n\}$ \mapsto §18, Folie 28

5. Modifiziere $gs_seq_set(\mathbb{Z})$ und Φ_3 mit ϑ und der Spezifikation

$G_\vartheta = (\mathbb{Z}, Seq(\mathbb{Z}), \lambda n. n \geq 1, O, Seq(\mathbb{Z}), \lambda n, s. range(s) \subseteq \{1..n\},$
 $\lambda n. [], \lambda L, s. s \subseteq L, \lambda n, s. \{s \cdot a \mid a \in \{1..n\}\}, \lambda s. \{s\})$

$\Phi_{3,\vartheta}(n, s) = \Phi_3(\{1..n\}, s) = nodups(s)$

$\Phi_{3,\vartheta}$ ist notwendig für G_ϑ

SYNTHESE EINER GLOBALSUCHE FÜR COSTAS ARRAYS

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

1. Wähle Globalsuchtheorie $\mathcal{G} = \text{gs_seq_set}(\mathbb{Z})$
2. Beweis für $(D, R, I, O) \ll \text{gs_seq_set}(\mathbb{Z})$ liefert $\vartheta[n] = \{1..n\}$
3. Wähle WF-Filter Φ so, daß Φ_ϑ notwendig für \mathcal{G}_ϑ beweisbar
 - $\text{perm}(p, \{1..n\}) \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)) \wedge s \sqsubseteq p \Rightarrow \Phi[\{1..n\}, s]$
 - Leicht beweisbar nur für $\Phi_3[M, s] = \text{nodups}(s)$
4. Leite zusätzlichen notwendigen Filter Ψ ab
 - Aus $\text{perm}(p, \{1..n\}) \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)) \wedge s \sqsubseteq p$
leite ab $\Psi[n, s] = \forall i < |s|. \text{nodups}(\text{dtrow}(s, j))$
5. Instantiiere den Standard-Globalsuchalgorithmus

```
FUNCTION Costas ( $n:\mathbb{Z}$ ) WHERE  $n \geq 1$  RETURNS  $\{p:\text{Seq}(\mathbb{Z}) \mid \dots \}$   
 $\equiv$  let rec Costasgs( $n, s$ )  
    =  $\{p \mid p \in \{s\} \wedge \text{perm}(p, \{1..n\}) \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)) \}$   
       $\cup \bigcup \{ \text{Costas}_{gs}(n, t) \mid t \in \{s \cdot i \mid i \in \{1..n\}\} \wedge \text{nodups}(t) \wedge \forall j < |t|. \text{nodups}(\text{dtrow}(t, j)) \}$   
    in if  $\text{nodups}([]) \wedge \forall j < |[]|. \text{nodups}(\text{dtrow}([], j))$   
        then Costasgs( $n, []$ ) else  $\emptyset$ 
```

SYNTHESESTRATEGIE FÜR GLOBALSUCHALGORITHMEN

Gegeben eine Problemspezifikation

FUNCTION $f(x:D)$ WHERE $I[x]$ RETURNS $\{y:R \mid O[x,y]\}$

1. Wähle Globalsuchtheorie \mathcal{G} mit Ausgabety R (Wissensbank)
2. Beweise $(D, R, I, O) \ll \mathcal{G}$ und extrahiere Substitution ϑ
Verfeinere \mathcal{G} zu Globalsuchtheorie \mathcal{G}_ϑ für (D, R, I, O)
3. Wähle Wohlfundiertheitsfilter Φ für \mathcal{G} (Wissensbank)
Beweise ‘ Φ_ϑ notwendig für \mathcal{G}_ϑ ’ (Axiom 4)
4. Bestimme zusätzlichen notwendigen Filter Ψ für \mathcal{G}_ϑ
– Leite Eigenschaften von x und s aus $sat[z,s] \wedge O[x,z]$ ab (Vorwärtsinferenz)
5. Instantiiere Globalsuch-Schema mit $\mathcal{G}_\vartheta, \Phi_\vartheta, \Psi$

FUNCTION $f(x:D)$ WHERE $I[x]$ RETURNS $\{y:R \mid O[x,y]\}$

\equiv if $\Phi[\vartheta(x), s_0(\vartheta(x))] \wedge \Psi[x, s_0(\vartheta(x))]$ then $aux(x, s_0(\vartheta(x)))$ else \emptyset

FUNCTION $aux(x, s:D \times S)$ WHERE $I[x] \wedge J[\vartheta(x), s] \wedge \Phi[\vartheta(x), s] \wedge \Psi[x, s]$

RETURNS $\{y:R \mid O[x,y] \wedge sat[y,s]\}$

$\equiv \{z \mid z \in ext[s] \wedge O[x,z]\} \cup \bigcup \{aux(x, t) \mid t \in split[\vartheta(x), s] \wedge \Phi[\vartheta(x), t] \wedge \Psi[x, t]\}$

PASST GLOBALSUCHE ZU SORTIERVERFAHREN?

Spezifikation muß mengenwertig formuliert werden

```
FUNCTION sort(L:Seq( $\mathbb{Z}$ )) WHERE true  
RETURNS {S:Seq( $\mathbb{Z}$ ) | rearranges(L,S) ∧ ordered(S)}
```

1. Wähle GS-theorie $\mathcal{G} = \text{gs_seq_set}(\mathbb{Z})$, beweise $(D, R, I, O) \ll \mathcal{G}$

- Zeige: $\forall L:\text{Seq}(\mathbb{Z}). \exists M:\text{Set}(\mathbb{Z}). \forall S:\text{Seq}(\mathbb{Z}). \text{SORT}(L,S) \Rightarrow \text{range}(S) \subseteq M$
- Aus `rearranges(L,S)` folgt `range(S)=range(L)`, also ist $\vartheta(L) = \text{range}(L)$

2. Wähle wf-Filter Φ so, daß Φ_{ϑ} notwendig für \mathcal{G}_{ϑ} beweisbar

- `rearranges(L,S) ∧ ordered(S) ∧ s ⊆ S` \Rightarrow $\Phi[\text{range}(L), s]$
- Aus `rearranges(L,S)` folgt `|S|=|L|`, also `|s| ≤ |L|`
- Instanz von $\Phi_{1,\vartheta}[L, s] = |s| \leq k$ für $k=|L|$

3. Leite zusätzlichen notwendigen Filter Ψ ab

- Aus `ordered(S) ∧ s ⊆ S` folgt `ordered(s) =: $\Psi[L, s]$`

Weitere ableitbare Eigenschaften sind algorithmisch nicht verwertbar

4. Instantiiere und optimiere den Standard-Globalsuchalgorithmus

SORTIEREN ALS GLOBALSUCHALGORITHMUS

- **Instantiiere Globalsuchschema mit \mathcal{G}_ϑ , Φ_ϑ und Ψ**

```
FUNCTION sort(L:Seq( $\mathbb{Z}$ )) WHERE true
  RETURNS {S:Seq( $\mathbb{Z}$ ) | rearranges(L,S)  $\wedge$  ordered(S)}
 $\equiv$  if  $|[]| \leq |L| \wedge$  ordered([]) then sortgs(L, []) else  $\emptyset$ 
FUNCTION sortgs(L:Seq( $\mathbb{Z}$ ), s:Seq( $\mathbb{Z}$ )) WHERE  $|s| \leq |L| \wedge$  ordered(s)
  RETURNS {S:Seq( $\mathbb{Z}$ ) | rearranges(L,S)  $\wedge$  ordered(S)}
 $\equiv$  { S |  $S \in \{s\} \wedge$  rearranges(L,S)  $\wedge$  ordered(S) }
   $\cup \bigcup \{ \text{sort}_{gs}(L,t) \mid t \in \{s \cdot i \mid i \in \text{range}(L)\} \wedge |t| \leq |L| \wedge \text{ordered}(t) \}$ 
```

- **Nach (kontextabhängigen) Simplifikationen**

```
FUNCTION sort(L:Seq( $\mathbb{Z}$ )) WHERE true
  RETURNS {S:Seq( $\mathbb{Z}$ ) | rearranges(L,S)  $\wedge$  ordered(S)}
 $\equiv$  sortgs(L, [])
FUNCTION sortgs(L:Seq( $\mathbb{Z}$ ), s:Seq( $\mathbb{Z}$ )) WHERE  $|s| \leq |L| \wedge$  ordered(s)
  RETURNS {S:Seq( $\mathbb{Z}$ ) | rearranges(L,S)  $\wedge$  ordered(S)}
 $\equiv$  if rearranges(L,s) then {s}
  else  $\bigcup \{ \text{sort}_{gs}(L,s \cdot i) \mid i \in L \wedge \text{last}(s) \leq i \}$ 
```

- **Weitere Optimierungen sind möglich**

– Dennoch ist Globalsuche nicht das beste Prinzip für Sortierverfahren

- **Formalisiere notwendige Begriffe**

- GS-Theorie, Filter, Wohlfundiertheit, Notwendigkeit, Generalisierung,...

- **Formuliere und beweise Globalsuch-Synthesetheorem**

$\forall \text{spec} = (D, R, I, O) : \text{SPEC}.$

FUNCTION $F(x:D) : \text{Set}(R)$ WHERE $I(x)$ RETURNS $\{z \mid O(x,z)\}$ ist erfüllbar

$\Leftarrow \exists G : \text{GS}. G$ is a GS-Theory

$\wedge \exists \vartheta : D \not\rightarrow D_G. G$ generalizes spec with ϑ

$\wedge \exists \Phi : \text{Filters}(G). \Phi$ wf-filter for $G \wedge \Phi_{\vartheta}$ necessary for $G_{\vartheta}(\text{spec})$

$\wedge \exists \Psi : \text{Filters}(G_{\vartheta}(\text{spec})). \Psi$ necessary for $G_{\vartheta}(\text{spec})$

- **Wende Synthesetheorem auf konkrete Probleme an**

- Beweisaufgaben sind formal leichter zu lösen als bei D&C Synthesen

- Wähle G (und Korrektheitsbeweis) aus Wissensbank

- Bestimme ϑ durch Extraktion aus Spezialisierungsbeweis

- Wähle Φ aus Wissensbank und beweise Notwendigkeit

- Leite Folgerungen von $\text{sat} \wedge O$ ab und wähle Ψ entsprechend

- Extrahiere Globalsuch-Algorithmus aus dem Beweis

GS-THEORIEN: FORMALISIERUNG VON STRUKTUR UND AXIOMEN

$GS \equiv \text{spec} : \text{SPEC} \times \text{S} : \text{TYPES} \times D \times S \rightarrow \mathbb{B} \times \dots$

G is a GS-theory

$\equiv \text{let } ((D, R, I, 0), S, J, s_0, \text{sat}, \text{split}, \text{ext}) = G \text{ in}$
 $\quad \forall x : D. I(x) \Rightarrow s_0(x) \text{ hält} \wedge J(x, s_0(x))$
 $\quad \wedge \forall x : D. \forall s : S. I(x) \wedge J(x, s) \Rightarrow \forall t \in \text{split}(x, s). J(x, t)$
 $\quad \wedge \forall x : D. \forall z : R. I(x) \wedge 0(x, z) \Rightarrow \text{sat}(z, s_0(x))$
 $\quad \wedge \forall x : D. \forall s : S. I(x) \wedge J(x, s) \Rightarrow \forall z : R. 0(x, z) \Rightarrow$
 $\quad \text{sat}(z, s) \Leftrightarrow \exists k : \mathbb{N}. \exists t \in \text{split}^k(x, s). z \in \text{ext}(t)$

$\text{Filters}(G)$

$\equiv \text{let } \dots = G \text{ in } D \times S \rightarrow \mathbb{B}$

Φ necessary for G

$\equiv \text{let } \dots = G \text{ in } \forall x : D. \forall s : S. \exists z : R. \text{sat}(z, s) \wedge 0(x, z) \Rightarrow \Phi(x, s)$

Φ wf-filter for G

$\equiv \text{let } \dots = G \text{ in } \forall x : D. \forall s : S. I(x) \wedge J(x, s) \Rightarrow \exists k : \mathbb{N}. \text{split}_{\Phi}^k(x, s) = \emptyset$

G generalizes spec with ϑ

$\equiv \text{let } \dots = G. \text{ and } (D', R', I', 0') = \text{spec} \text{ in}$

$R' \subset R \wedge \forall x : D'. I'(x) \Rightarrow I(\vartheta(x)) \wedge \forall z : R'. 0'(x, z) \Rightarrow 0(\vartheta(x), z)$

$G_{\vartheta}(\text{spec}) \dots$

\vdots

SYNTHESE VON COSTAS ARRAYS MIT EINEM BEWEISSYSTEM

top

```
⊢ FUNCTION Costas(n:ℤ):Set(Seq(ℤ)) WHERE n ≥ 1
  RETURNS {p | perm(p, {1..n}) ∧ ∀j < |p|. nodups(dtrow(p, j))}
  ist erfüllbar
```

BY lemma 'GS-Synthese'

```
1. ⊢ ∃G. G is a GS-Theory
  ∧ ∃ϑ. G generalizes spec with ϑ
  ∧ ∃Φ. Φ wf-filter for G ∧ Φϑ necessary for Gϑ(spec)
  ∧ ∃Ψ. Ψ necessary for Gϑ(spec)
  where spec = (ℤ, Seq(ℤ), λn. n ≥ 1,
                λn, p. perm(p, {1..n}) ∧ ∀j < |p|. nodups(dtrow(p, j)) )
```

Typinformation zugunsten der Lesbarkeit unterdrückt (Display-Mechanismus)

Restliche Aufgabe: Bestimmung von G , ϑ , Φ und Ψ .

Extraktion des Algorithmus automatisch nach Lösung dieses Ziels

COSTAS ARRAYS SYNTHESE (2): WAHL DER GS-THEORIE

top 1

$\vdash \exists G. G \text{ is a GS-Theory}$
 $\wedge \exists \vartheta. G \text{ generalizes spec with } \vartheta$
 $\wedge \exists \Phi. \Phi \text{ wf-filter for } G \wedge \Phi_{\vartheta} \text{ necessary for } G_{\vartheta}(\text{spec})$
 $\wedge \exists \Psi. \Psi \text{ necessary for } G_{\vartheta}(\text{spec})$
where $\text{spec} = (\mathbb{Z}, \text{Seq}(\mathbb{Z}), \lambda n. n \geq 1,$
 $\lambda n, p. \text{perm}(p, \{1..n\}) \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)))$

BY \exists -intro $\text{gs_seq}(\mathbb{Z})$ THEN unfold 'perm'

1. $\vdash \text{gs_seq}(\mathbb{Z})$ is a GS-Theory
2. $\vdash \exists \vartheta. \text{gs_seq}(\mathbb{Z})$ generalizes spec with ϑ
 $\wedge \exists \Phi. \Phi \text{ wf-filter for } \text{gs_seq}(\mathbb{Z})$
 $\wedge \Phi_{\vartheta} \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\vartheta}(\text{spec})$
 $\wedge \exists \Psi. \Psi \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\vartheta}(\text{spec})$
where spec
 $= (\mathbb{Z}, \text{Seq}(\mathbb{Z}), \lambda n. n \geq 1,$
 $\lambda n, p. \text{nodups}(p) \wedge \text{range}(p) = \{1..n\} \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)))$

Teilziel 1 ist Lemma der Wissensbank

COSTAS ARRAYS SYNTHESE (3): BESTIMMUNG DER SUBSTITUTION ϑ

top 1 2

$\vdash \exists \vartheta. \text{gs_seq}(\mathbb{Z}) \text{ generalizes spec with } \vartheta$
 $\wedge \exists \Phi. \Phi \text{ wf-filter for } \text{gs_seq}(\mathbb{Z})$
 $\wedge \Phi_{\vartheta} \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\vartheta}(\text{spec})$
 $\wedge \exists \Psi. \Psi \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\vartheta}(\text{spec})$
 where spec
 $= (\mathbb{Z}, \text{Seq}(\mathbb{Z}), \lambda n. n \geq 1,$
 $\lambda n, p. \text{nodups}(p) \wedge \text{range}(p) = \{1..n\} \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)))$

BY andR THEN exR $\lambda n. \{1..n\}$

1. $\vdash \text{gs_seq}(\mathbb{Z}) \text{ generalizes spec with } \lambda n. \{1..n\}$
 where spec =
2. $\vdash \exists \Phi. \Phi \text{ wf-filter for } \text{gs_seq}(\mathbb{Z})$
 $\wedge \Phi_{\vartheta} \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
 $\wedge \exists \Psi. \Psi \text{ necessary for } \text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
 where spec =

Teilziel 1 lösbar durch einfachen Beweiser

In Teilziel 1 kann ϑ auch aus einem Beweis von $\text{gs_seq}(\mathbb{Z}) \text{ generalizes spec}$ extrahiert werdend. Hierzu wird das konstruktive Auswahlaxiom (§9, Folie 39) angewandt und dann (nach Auffalten) folgendes Ziel bewiesen, aus dem sich dann $\vartheta = \lambda n. \{1..n\}$ ergibt:

$$\begin{aligned}
 \forall n: \mathbb{Z}. n \geq 1 &\Rightarrow \exists M: \text{Set}(\mathbb{Z}). \forall p: \text{Seq}(\mathbb{Z}). \text{nodups}(p) \wedge \text{range}(p) = \{1..n\} \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)) \\
 &\Rightarrow \text{range}(p) \subseteq M
 \end{aligned}$$

COSTAS ARRAYS SYNTHESE (4): WAHL DES FILTERS Φ

top 1 2 2

$\vdash \exists \Phi. \Phi$ wf-filter for $\text{gs_seq}(\mathbb{Z})$
 $\wedge \Phi_{\vartheta}$ necessary for $\text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
 $\wedge \exists \Psi. \Psi$ necessary for $\text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
where spec
= $(\mathbb{Z}, \text{Seq}(\mathbb{Z}), \lambda n. n \geq 1,$
 $\lambda n, p. \text{nodups}(p) \wedge \text{range}(p) = \{1..n\} \wedge \forall j < |p|. \text{nodups}(\text{dtrow}(p, j)))$

BY \exists -intro $\lambda S, V. \text{nodups}(V)$

1. $\vdash \lambda S, V. \text{nodups}(V)$ wf-filter for $\text{gs_seq}(\mathbb{Z})$
2. $\vdash \lambda n, V. \text{nodups}(V)$ necessary for $\text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
 where spec =
3. $\vdash \exists \Psi. \Psi$ necessary for $\text{gs_seq}(\mathbb{Z})_{\lambda n. \{1..n\}}(\text{spec})$
 where spec =

Teilziel 1 ist Lemma der Wissensbank

COSTAS ARRAYS SYNTHESE (5): Φ_{ϑ} NOTWENDIG/ WAHL VON Ψ

```
# top 1 2 2 2
```

```
⊢ λn, V. nodups(V) necessary for gs_seq( $\mathbb{Z}$ )λn. {1..n}(spec)
```

```
where spec
```

```
= ( $\mathbb{Z}$ , Seq( $\mathbb{Z}$ ), λn. n ≥ 1,
```

```
λn, p. nodups(p) ∧ range(p) = {1..n} ∧ ∀j < |p|. nodups(dtrow(p, j)))
```

```
BY undo_abstractions THEN unfold 'specialize' THEN unfold 'necessary'
```

```
1. ⊢ ∀n. ∀V.
```

```
(∃p. nodups(p) ∧ range(p) = {1..n} ∧ ∀j < |p|. nodups(dtrow(p, j)) ∧ V ⊆ p)
```

```
⇒ nodups(V)
```

```
* BY Auto
```

Auto zerlegt logisch und konsultiert dabei Fakten der Wissensbank

```
# top 1 2 2 3
```

```
⊢ ∃Ψ. Ψ necessary for gs_seq( $\mathbb{Z}$ )λn. {1..n}(spec)
```

```
where spec = .....
```

```
BY undo_abstractions THEN unfold 'specialize' THEN unfold 'necessary'
```

```
1. ⊢ ∃Ψ. ∀n. ∀V.
```

```
(∃p. nodups(p) ∧ range(p) = {1..n} ∧ ∀j < |p|. nodups(dtrow(p, j)) ∧ V ⊆ p)
```

```
⇒ Ψ(n, V)
```

```
* BY ∃-intro λn, V. ∀j ∈ dom(V). nodups(dtrow(V, j)) THEN Auto
```

GLOBALSUCH-SYNTHESE IM RÜCKBLICK

- **Wissensbasierte Erzeugung eines Suchalgorithmus**

- Sinnvoll, wenn Problem rekursiv zerlegbar in “additive” Teillösungen
- Schematischer Algorithmus verwendet 4 Basiskomponenten
- Komponenten lassen sich in **einem Schritt** aus generischen Suchstrukturen erzeugen und durch Filter verfeinern
- Standard-Globalsuchtheorien und passende Wohlgeformtheitsfilter müssen in Wissensbank vorgespeichert sein
- Techniken sind Spezialisierung und Vorwärtsinferenz

- **Erfolgreich in der Praxis**

- Schematische Algorithmen sind schnell erzeugt und formal optimiert
- Effiziente Lösung von Scheduling Problemen wurde 1993 demonstriert
Entscheidend war Wahl der richtigen Algorithmenstruktur
- Korrektheit der Algorithmen ist durch theoretische Vorarbeiten garantiert
- Korrektheit der **implementierten** Methode kann durch Integration in Beweissysteme gesichert werden