

Wissensrepräsentation

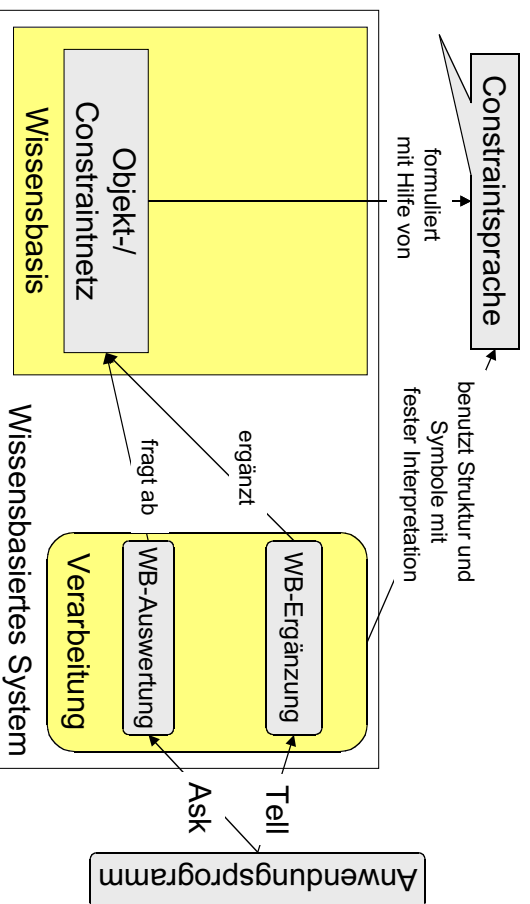
—

Christopher Habel, Hedda Schmidke
Sommersemester 2004

Sitzung 11: Constraintverarbeitung

- Backtracking
- Constraint-Propagation
 - Kantenkonsistenz
- Relationensysteme (Qualitatives Schließen)
 - Kompositionstabellen

Wissensbasiertes System mit Constraint



Constraint satisfaction problems (CSPs)

Standard search problem:

state is a "black box"—any old data structure
that supports goal test, eval, successor

CSP:

state is defined by *variables* X_i with *values* from *domain* D_i

goal test is a set of *constraints* specifying

allowable combinations of values for subsets of variables

Simple example of a *formal representation language*

Allows useful *general-purpose* algorithms with more power
than standard search algorithms

Example: Map-Coloring



Variables WA, NT, Q, NSW, V, SA, T

Domains $D_i = \{red, green, blue\}$

Constraints: adjacent regions must have different colors

e.g., $WA \neq NT$ (if the language allows this), or

$(WA, NT) \in \{(red, green), (red, blue), (green, red), (green, blue), \dots\}$

Standard search formulation (incremental)

Let's start with the straightforward, dumb approach, then fix it

States are defined by the values assigned so far

◇ **Initial state:** the empty assignment, $\{\}$

◇ **Successor function:** assign a value to an unassigned variable
that does not conflict with current assignment.
⇒ fail if no legal assignments (not fixable!)

◇ **Goal test:** the current assignment is complete

1) This is the same for all CSPs!

2) Every solution appears at depth n with n variables
⇒ use depth-first search

3) Path is irrelevant, so can also use complete-state formulation

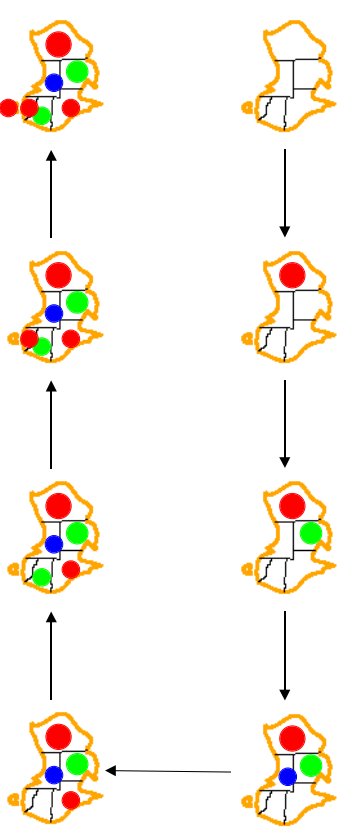
4) $b = (n - \ell)d$ at depth ℓ , hence $n!d^n$ leaves!!!!

Map coloring: incremental search (2)



- Inkrementelle Suche erfordert
- **backtracking**
- **und / oder**
- **Heuristiken / Strategien**

Map coloring: incremental search (1)



Backtracking search

Variable assignments are commutative, i.e.,

$[WA = red \text{ then } NT = green]$ same as $[NT = green \text{ then } WA = red]$

Only need to consider assignments to a single variable at each node

⇒ $b = d$ and there are d^n leaves

Depth-first search for CSPs with single-variable assignments is called **backtracking search**

Backtracking search is the basic uninformed algorithm for CSPs

Can solve n -queens for $n \approx 25$

Backtracking search

```

function BACKTRACKING-SEARCH(csp) returns solution/failure
  return RECURSIVE-BACKTRACKING([], csp)

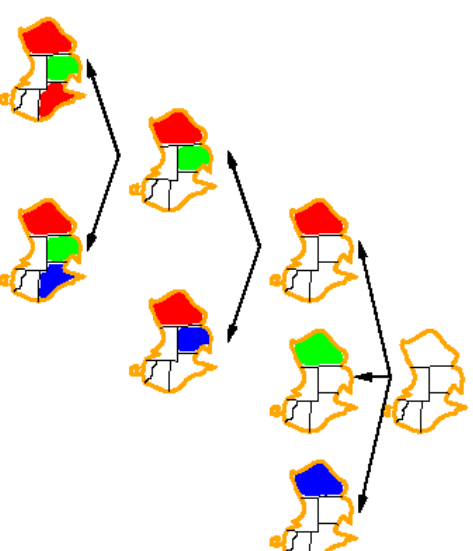
function RECURSIVE-BACKTRACKING(assigned, csp) returns solution/failure
  if assigned is complete then return assigned
  var  $\leftarrow$  SELECT-UNASSIGNED-VARIABLE(VARIABLES[csp], assigned, csp)
  for each value in ORDER-DOMAIN-VALUES(var, assigned, csp) do
    if value is consistent with assigned according to CONSTRAINTS[csp] then
      result  $\leftarrow$  RECURSIVE-BACKTRACKING([var = value | assigned], csp)
      if result  $\neq$  failure then return result
  end
  return failure
  
```

Improving backtracking efficiency

General-purpose methods can give huge gains in speed:

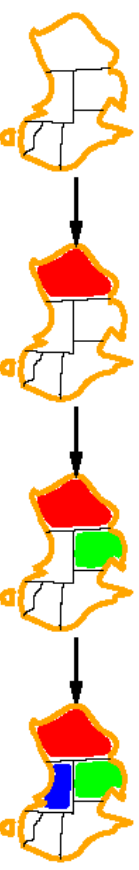
1. Which variable should be assigned next?
2. In what order should its values be tried?
3. Can we detect inevitable failure early?
4. Can we take advantage of problem structure?

Backtracking example



Most constrained variable

Most constrained variable:
choose the variable with the fewest legal values



jede Variable
kann gewählt werden:
freie Wahl

jetzt ist der gemeinsame
Nachbar der bisher
gewählten Variablen
maximal eingeschränkt

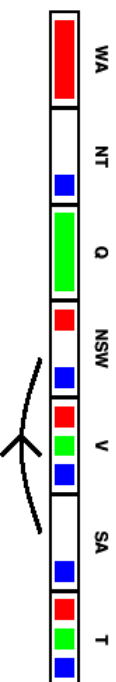
die Nachbarn der zuerst
gewählten Variablen
sind in ihrer Belegung eingeschränkt,
also zwischen diesen muss gewählt werden.

Arc consistency

Simplest form of propagation makes each arc **consistent**

$X \rightarrow Y$ is consistent iff

for **every** value x of X there is **some** allowed y



Falls Wert von SA blau,
dann existiert für NSW ein zulässiger Wert, nämlich rot.

Arc consistency algorithm

function AC3(csp) returns an equivalent, arc consistent CSP

queue \leftarrow arcs-of(*csp*)

loop while *queue* is not empty **do**

$(X_i, X_j) \leftarrow$ Remove-Front(*queue*)

if Remove-Inconsistent(X_i, X_j, csp) **then**

loop for each X_k in Neighbors(*csp*, X_j) **do** add (X_k, X_j) to *queue*

function Remove-Inconsistent(X_i, X_j, csp) returns true iff we remove a value

removed \leftarrow false

loop for each x in Domain(*csp*, X_i) **do**

if no y in Domain(*csp*, X_j) allows (x, y) to satisfy Constraint(*csp*, X_i, X_j),

then delete x from Domain(*csp*, X_i); *removed* \leftarrow true

return *removed*

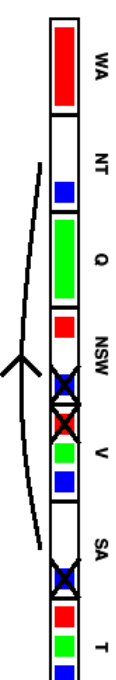
- $O(n^2d^3)$, kann auf $O(n^2d^2)$ reduziert werden
- kann nicht alle Inkonsistenzen aufdecken

Arc consistency

Simplest form of propagation makes each arc **consistent**

$X \rightarrow Y$ is consistent iff

for **every** value x of X there is **some** allowed y



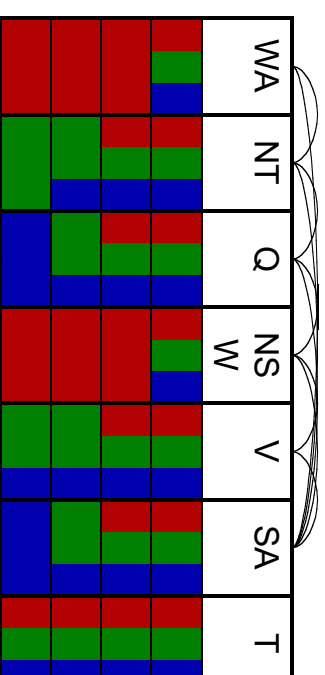
If X loses a value, neighbors of X need to be rechecked

Arc consistency detects failure earlier than forward checking

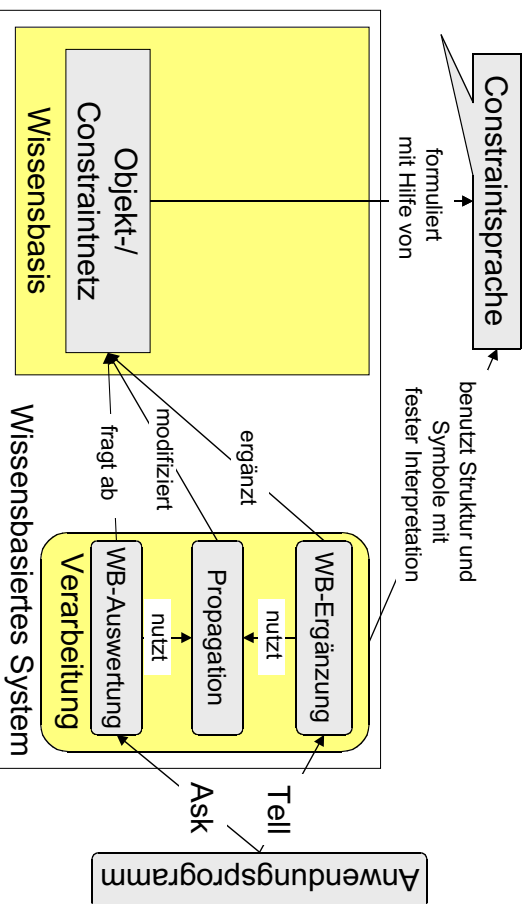
Can be run as a **preprocessor** or after each assignment

Beispiel: Inkonsistentes, Kanten-konsistentes Netz

- Die Zuordnung WA = rot, NSW = rot kann nicht zu einer konsistenten Markierung erweitert werden
- Dieses ist aber allein durch Constraint-Propagation nicht zu erkennen.



Wissensbasiertes System mit Constraint



Beispiel: "Allen-Relationen"

Dreizehn einander wechselseitig ausschließende Relationen zwischen Intervallen auf der Linie

before(t_1, t_2):	t_1 ist vor t_2 und es ist noch was dazwischen.
equal(t_1, t_2):	t_1 und t_2 sind dieselbe Periode
meets(t_1, t_2):	t_1 ist vor t_2 und es gibt keine Periode zwischen t_1 und t_2 , d.h. t_1 endet, wenn t_2 beginnt
overlaps(t_1, t_2):	t_1 beginnt vor t_2 und endet nach dem Anfang von t_2
during(t_1, t_2):	t_1 ist vollständig in t_2 enthalten
starts(t_1, t_2):	t_1 hat denselben Anfang wie t_2 , endet aber vor dem Ende von t_2
finishes(t_1, t_2):	t_1 hat dasselbe Ende wie t_2 , beginnt aber nach dem Anfang von t_2

Relationensysteme und Verarbeitung

Binäre Relationen

Intervallrelationen (auf der Linie)

X before Y	<	Y after X	>	XXX YYY
X equal Y	=	Y equal X	=	XXX YYY
X meets Y	m	Y met-by X	mi	XXXXYYY
X overlaps Y	o	Y overlaped-by X	oi	XXXX YYYY
X during Y	d	Y contains X	di	XXX YYYYYYY
X starts Y	s	Y started-by X	si	XXX YYYYYYY
X finishes Y	f	Y finished-by X	fi	XXX YYYYYYY

Beispiel: "RC-C-8"

DC(X, Y) DisConnected		EC(X, Y) Externally Connected	
PO(X, Y) Partially Overlapping		EQ(X, Y) Equal	
TPPX, Y) Tangential Proper Part		TPPI(X, Y) Tangential Proper Part Inverse	
NTPPX, Y) Non-Tangential Proper Part		NTPPI(X, Y) Non-Tangential Proper Part Inverse	

Wissensrepräsentation, SoSe 2004

Ch. Habel / C. Eschenbach / H. Schmidtko

11 – 25

Endliche Mengen von Basisrelationen

Die Relationen sind

- paarweise exklusiv
 - kein Paar von Intervallen kann in zwei der Relationen stehen
 - Exhaustiv
 - jedes Paar steht in mindestens einer Relation
- Ergibt sich aus**
- Intuition und Anschauung (zur Motivation)
 - Formalisierung (Axiome, Definitionen)
 - Beweis der (intuitiv gewünschten) Eigenschaften

Wissensrepräsentation, SoSe 2004

Ch. Habel / C. Eschenbach / H. Schmidtko

11 – 27

Restriktionen durch Komposition

Informationskombinatorik aus Relationskomposition

- entsprechen Formeln der Form:
 $\forall t_1, t_2, t_3 [t_1 R_1 t_2 \wedge t_2 R_2 t_3 \Rightarrow t_1 R_3 t_3]$
- Beispiel:
 $\forall t_1, t_2, t_3 [\text{meets}(t_1, t_3) \wedge \text{meets}(t_3, t_2)] \Rightarrow \text{before}(t_1, t_2)]$
 $\forall t_1, t_2, t_3 [\text{before}(t_1, t_3) \wedge \text{before}(t_3, t_2)] \Rightarrow \text{before}(t_1, t_2)]$
- Restriktionen sollten den 'intendierten' Bedeutungen entsprechen
- Theoreme ergeben sich aus formaler Spezifikation

Wissensrepräsentation, SoSe 2004

Ch. Habel / C. Eschenbach / H. Schmidtko

11 – 26

Definition von binären Relationen mit den Basisrelationen (2)

Relationensysteme

- Sei \mathcal{B} die Menge der Basisrelationen, dann kann jede binäre Relation R über dem Grundbereich dargestellt werden,
- als Disjunktion von Basisrelationen $R = B_i \cup \dots \cup B_j$
- alternativ, als $R = \{B_i, \dots, B_j\}$
- Jede Disjunktion von Basisrelationen stellt eine andere Relation dar.
- Intervallrelationen: Insgesamt $2^{13} - 1 = 8191$ konsistente Relationen
- Disjunktionen von Basisrelationen als **Normalformdarstellung** von Relationen
- **Boolesche Algebra** der Relationen: Konjunktion, Disjunktion, Komplement

Wissensrepräsentation, SoSe 2004

Ch. Habel / C. Eschenbach / H. Schmidtko

11 – 28

Kompositionstabellen

- Die Kompositionstabelle kann als ein System von Abhängigkeiten von Beschränkungen (Constraints) aufgefasst werden.

Mögliche Fälle

- eine passende Relation:
 $\text{meets}(t, t'') \wedge \text{meets}(t'', t') \rightarrow \text{before}(t, t')$
- mehrere mögliche Relationen:
 $\text{contains}(t, t'') \wedge \text{meets}(t'', t')$
 $\rightarrow (\text{contains}(t, t') \vee \text{finished_by}(t, t') \vee \text{overlaps}(t, t'))$
- keine 'neue' Information:
 $\text{before}(t, t'') \wedge \text{after}(t'', t')$

<	>	d	di	o	oi	m	mi	s	si	f	fi
<	\mathcal{B}	<om ds	<	<	<om ds	<	<om ds	<	<	<om ds	<
>	\mathcal{B}	>oi midf	>	>oi midf	>	>oi midf	>	<om ds	>	>	>
d	<	d	\mathcal{B}	<om ds	>oi midf	<	>oi midf	d	>oi midf	d	<om ds
di	<om di fi	>oi di mi si	oi di di =	oi di fi	oi di si	oi di fi	oi di si	oi di fi	oi di si	oi di si	oi di fi
o	<	>oi di mi si	ods	<om di fi	<om di =	<	oi di si	oi df	o di fi	ods	<om di fi
oi	<om di fi	>	oi df	>oi mi di =	oi df	o di fi	o di fi	oi df	oi mi si	oi df	oi di si
m	<	>oi di mi si	ods	<	ods	<	ffi =	m	m	ods	<
mi	<om di fi	>	oi df	>	oi df	>	oi df	>	mi	mi	<mi
s	<	>	d	<om di fi	<om df	<	mi	s	s si =	d	m o
si	<om di fi	>	oi df	di	oi df fi	oi	oi df fi	mi	s si =	si	oi di
f	<	>	d	>oi di mi si	>oi mi si	>	d	>oi mi si	f	f fi =	f fi =
fi	<	>oi di mi si	ods	di	o	oi di si	m	oi di si	oi di si	fi	fi

Kompositionstabelle – Kompositionsaxiome

Kompositionstabelle der Basisrelationen für Perioden

<	>	d	di	o	oi
<	\mathcal{B}	<om ds	<	<	<om ds
>	\mathcal{B}	>oi midf	>	>oi midf	>
d	<	d	\mathcal{B}	<om ds	>oi midf
di	<om di fi	>oi di mi si	oi di di =	di	oi di si

→ Allen (1983)

Zu je zwei Basisrelationen ist die Menge der kompatiblen Basisrelationen eingetragen.

Die Tabelle kann als alternatives Axiomensystem [AK] interpretiert werden. Z.B.:

$$[A<d] \vee t t' t'' [t < t'' \wedge t'' d t' \rightarrow (t < t' \vee t o t' \vee t m t' \vee t d t' \vee t s t')]]$$

Schließen mit Kompositionstabellen: Constraint-Verfahren

Berechnungssystem für Schließen über Relationen

- Objekte werden als Knoten in einem gerichteten Graphen repräsentiert.
- Die gerichteten Kanten sind mit Mengen von Relationensymbolen etikettiert:
 - Information über die Relation zwischen den verbundenen Knoten (gegebenenfalls disjunktive Information)
- Die Verarbeitung besteht in der Prüfung der Kompatibilität der Relationen an den **Kanten**, die jeweils **drei Knoten** verbinden.
- Die (3-stelligen) Constraints gelten einheitlich und sind in einer gesonderten Kompositionstabelle gespeichert.