

Grundlagen der Künstlichen Intelligenz

27. Constraint-Satisfaction-Probleme: Constraint-Graphen

Malte Helmert

Universität Basel

20. April 2015

Grundlagen der Künstlichen Intelligenz

20. April 2015 — 27. Constraint-Satisfaction-Probleme: Constraint-Graphen

27.1 Constraint-Graphen

27.2 Unzusammenhängende Graphen

27.3 Bäume

27.4 Zusammenfassung

Constraint-Satisfaction-Probleme: Überblick

Kapitelüberblick Constraint-Satisfaction-Probleme:

- ▶ 22.–23. Einführung
- ▶ 24.–26. Kernalgorithmen
- ▶ 27.–28. Problemstruktur
 - ▶ 27. **Constraint-Graphen**
 - ▶ 28. Zerlegungsmethoden

27.1 Constraint-Graphen

Motivation

- ▶ Zum Lösen eines Constraint-Netzes müssen bei n Variablen mit jeweils k Werten bis zu k^n Belegungen getestet werden.
- ▶ Inferenz kann diese Explosion der Möglichkeiten oft abmildern, aber nicht immer verhindern.
- ▶ Viele praktisch relevante Constraint-Netze sind aber dennoch effizient lösbar, wenn man ihre spezielle **Struktur** ausnutzt.

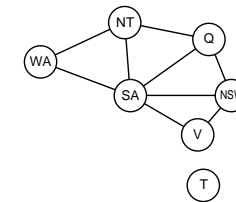
Constraint-Graph

Definition (Constraint-Graph)

Sei $\mathcal{C} = \langle V, \text{dom}, (R_{uv}) \rangle$ ein Constraint-Netz.

Der **Constraint-Graph** von \mathcal{C} ist der Graph, dessen Knoten V sind und der eine Kante zwischen u und v genau dann hat, wenn R_{uv} ein nichttrivialer Constraint ist.

Beispiel: Färbung von Australiens Bundesstaaten und Territorien



27.2 Unzusammenhängende Graphen

Unzusammenhängende Constraint-Graphen

Satz (Unzusammenhängende Constraint-Graphen)

Wenn der Constraint-Graph von \mathcal{C} in mehrere Zusammenhangskomponenten zerfällt, reicht es aus, das Teilproblem für jede Komponente separat zu lösen.

Die Vereinigung der Teillösungen ist dann eine Lösung für \mathcal{C} .

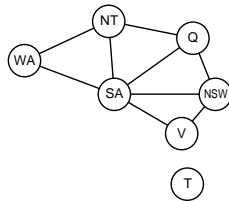
Beweis.

Eine aus Teillösungen zusammengesetzte Lösung erfüllt alle Constraints, die **innerhalb** eines Teilproblems liegen.

Es gibt keine nichttrivialen Constraints, die **nicht** innerhalb eines Teilproblems liegen. □

Unzusammenhängende Constraint-Graphen: Beispiel

Beispiel: Tasmanien separat vom Rest von Australien einfärben



Weiteres Beispiel:

Zerfalle ein Problem mit $k = 2$, $n = 30$ in drei Teile der Grösse 10.

Ersparnis?

27.3 Bäume

Bäume als Constraint-Graphen

Satz (Bäume als Constraint-Graphen)

Sei \mathcal{C} ein Constraint-Netz mit n Variablen und maximaler Wertebereichsgrösse k , dessen Constraint-Graph ein **Baum** oder **Wald** ist (d. h. keine Zyklen enthält).

Dann kann \mathcal{C} in Zeit $O(nk^2)$ gelöst werden (bzw. die Nichtexistenz einer Lösung bewiesen werden).

Zahlenbeispiel: $k = 5$, $n = 10$

$\rightsquigarrow k^n = 9765625$, $nk^2 = 250$

Bäume als Constraint-Graphen: Algorithmus

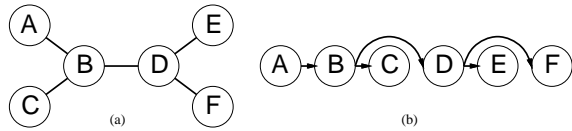
Algorithmus für Baum:

- ▶ Bilde gerichteten Baum, indem beliebige Variable zur Wurzel erklärt wird.
- ▶ Ordne Variablen v_1, \dots, v_n so, dass Vater immer vor Kindern kommt.
- ▶ Für $i \in \langle n, n-1, \dots, 2 \rangle$: rufe **revise**($v_{\text{parent}(i)}, v_i$) auf
 \rightsquigarrow jeder Knoten ist kantenkonsistent in Bezug auf Kinder
- ▶ wenn dabei ein Wertebereich leer wird: Problem unlösbar
- ▶ Ansonsten löse mit **BacktrackingWithInference** mit Variablenordnung v_1, \dots, v_n und Forward Checking
 \rightsquigarrow Lösung wird **ohne Backtracking-Schritte** gefunden

Beweis: \rightsquigarrow Übungen

Bäume als Constraint-Graphen: Beispiel

Constraint Netz \rightsquigarrow gerichteter Baum, geordnet:



Revise-Schritte:

- ▶ revise(D, F)
- ▶ revise(D, E)
- ▶ revise(B, D)
- ▶ revise(B, C)
- ▶ revise(A, B)

Lösungsfindung:

Backtracking mit Ordnung $A \prec B \prec C \prec D \prec E \prec F$

27.4 Zusammenfassung

Zusammenfassung

- ▶ Constraint-Netze mit vielen Variablen können einfach zu lösen sein, wenn sie eine einfache **Struktur** haben.
- ▶ **Constraint-Graph** formalisiert diese Struktur
 - ▶ **mehrere Zusammenhangskomponenten:** Lösung als **separate** Instanzen für jede Komponente
 - ▶ **Baum:** Lösungsalgorithmus mit **linearem Zeitaufwand** in der Anzahl der Variablen