Grundlagen der Künstlichen Intelligenz

25. Constraint-Satisfaction-Probleme: Zerlegungsmethoden

Malte Helmert

Universität Basel

25. April 2014

Constraint-Satisfaction-Probleme: Überblick

Kapitelüberblick Constraint-Satisfaction-Probleme:

- 19.–20. Einführung
- 21.–23. Kernalgorithmen
- 24.–25. Problemstruktur
  - 24. Constraint-Graphen
  - 25. Zerlegungsmethoden

Zerlegungsmethoden

Kompliziertere Graphen

Was, wenn der Constraint-Graph kein Baum ist und nicht in Komponenten zerfällt?

- Idee 1: Konditionierung
- Idee 2: Baumzerlegung

Konditionierung

Konditionierung

Konditionierung

Idee: Führe Backtracking mit Forward Checking durch, bis der Constraint-Graph eingeschränkt auf die restlichen Variablen in Komponenten zerfällt bzw. ein Baum ist.

Restproblem $\rightsquigarrow$ Algorithmus für einfache Constraint-Graphen

Konditionierung

Konditionierung

Idee: Führe Backtracking mit Forward Checking durch, bis der Constraint-Graph eingeschränkt auf die restlichen Variablen in Komponenten zerfällt bzw. ein Baum ist.

Restproblem $\rightsquigarrow$ Algorithmus für einfache Constraint-Graphen

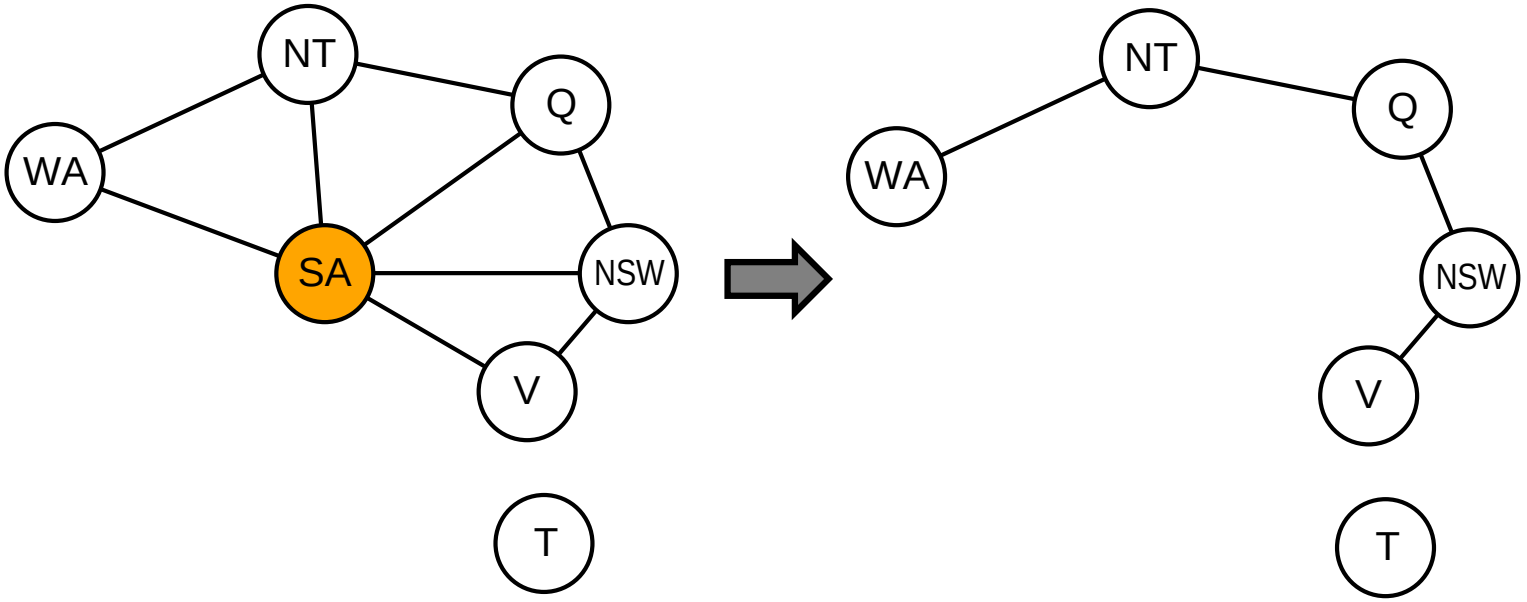
Cutset conditioning: wähle Variablenordnung so, dass vordere Variablen möglichst kleines Cutset bilden (Variablenmenge, ohne die der Constraint-Graph zykelfrei ist).

Zeitaufwand: n Variablen, davon $m < n$ im Cutset, Wertebereiche der Grösse k: $O(k^m \cdot (n-m)k^2)$

(Optimale Cutsets zu finden ist seinerseits NP-vollständig.)

Konditionierung: Beispiel

Australien-Beispiel: Cutset der Grösse 1 reicht aus:



Baumzerlegung

Baumzerlegung

Baumzerlegung

Zerlege Constraint-Graph in überlappende Teile, so dass:

- jede Variable in einem Teilproblem auftritt
- jede Kante (nichttrivialer Constraint) in einem Teilproblem auftritt

Teilprobleme werden durch Meta-Kanten zu Wald verbunden.
Der Wald darf frei gewählt werden, solange gilt:

- Für jede Variable ist die Menge der Teilprobleme, in denen diese Variable auftritt, im Wald zusammenhängend.

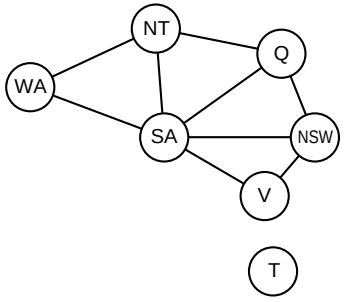
Gute Baumzerlegungen

Ziel: jedes Teilproblem hat möglichst wenige Variablen

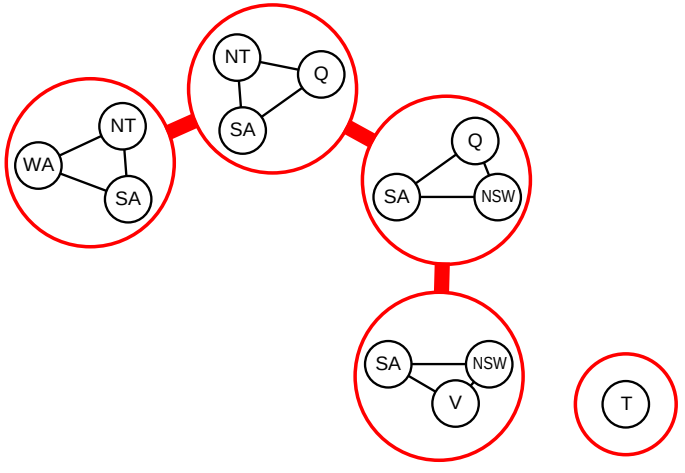
- entscheidend: Teilproblem mit den meisten Variablen
- dessen Variablenzahl minus 1 heisst Weite der Zerlegung
- beste Weite aller Zerlegungen: Baumweite des Graphen (NP-vollständig)

Baumzerlegung: Beispiel

Constraint-Netz:



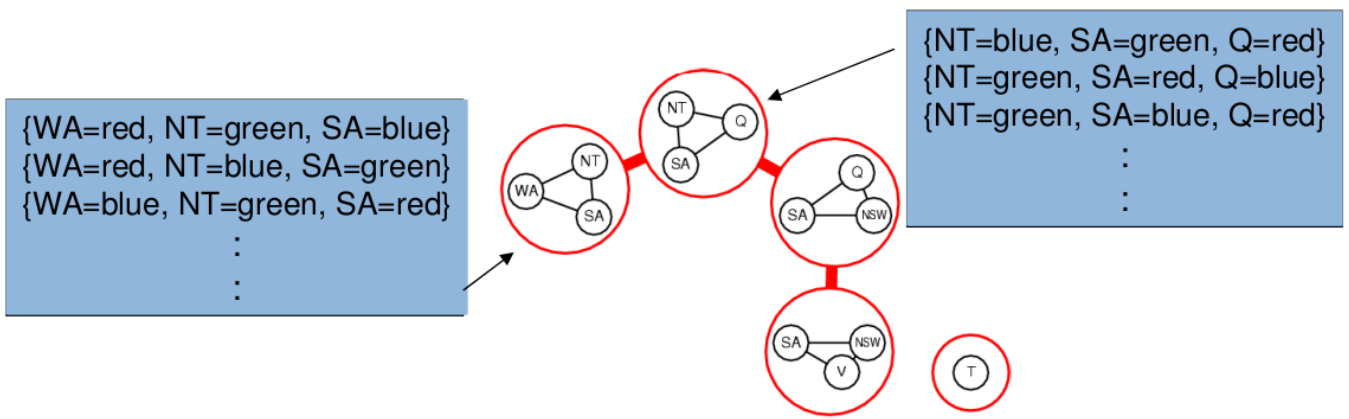
Baumzerlegung:



Baumzerlegung: Algorithmus

Algorithmus:

- Finde alle Lösungen für alle Teilprobleme
  und baue daraus ein baumartiges Meta-Constraint-Netz.
- Constraints darin: Teillösungen müssen kompatibel sein,
  d. h. gemeinsame Variablen identisch belegen.
- Löse Meta-Netz mit Verfahren für baumartige Netze.



Zeitaufwand: $O(nk^{w+1})$ bei Weite w der Zerlegung

(Setzt spezialisierte Version von revise voraus; sonst $O(nk^{2w+2})$.)

Zusammenfassung

Zusammenfassung: Kapitel

- Führe komplizierte Constraint-Graphen auf einfache zurück.
- Cutset-Konditionierung:
  - Wähle möglichst wenige Variablen (Cutset), so dass nach Belegung dieser Variablen Restproblem strukturell einfach ist.
  - Suche über Belegung der Variablen im Cutset
- Baumzerlegung: bilde baumartiges Meta-Constraint-Netz
  - Meta-Variablen: Gruppen von Ursprungsvariablen, die gemeinsam alle Variablen und Constraints abdecken
  - Werte entsprechen konsistenten Belegungen der Gruppe
  - Constraint zwischen überlappenden Gruppen, um Kompatibilität zu sichern
  - Gesamtalgorithmus exponentiell in Weite der Zerlegung (Grösse der grössten Gruppe)

Zusammenfassung: CSPs

Constraint-Satisfaction-Problem (CSP)

allgemeiner Formalismus für Probleme, bei denen

- Variablen so belegt werden müssen, dass
- bestimmte Bedingungen (Constraints) erfüllt sind.

- Algorithmen: Backtracking-Suche + Inferenz
  (z. B. Forward Checking, Kantenkonsistenz, Pfadkonsistenz)
- Variablen- und Wertreihenfolge wichtig
- mehr Effizienz: Ausnutzen der Struktur
  des Constraint-Graphen (unabhängige Komponenten; Bäume)

Weiterführende Themen

weitere, hier nicht behandelte Aspekte:

- Backjumping: Backtracking über mehrere Ebenen auf einmal
- No-Good Learning: Inferieren zusätzlicher Constraints anhand von Information, die beim Backtracking gesammelt wird
- lokale Suchverfahren im Raum der totalen, aber nicht notwendigerweise konsistenten Belegungen
- handhabbare Constraint-Klassen: Identifikation von Constraint-Typen, die polynomielle Algorithmen erlauben
- Verallgemeinerung mit Qualitätsfunktion für Lösungen: Constraint-Optimierungs-Probleme (COP)

$\rightsquigarrow$ mehr als genug Stoff für einsemestrige Vorlesung