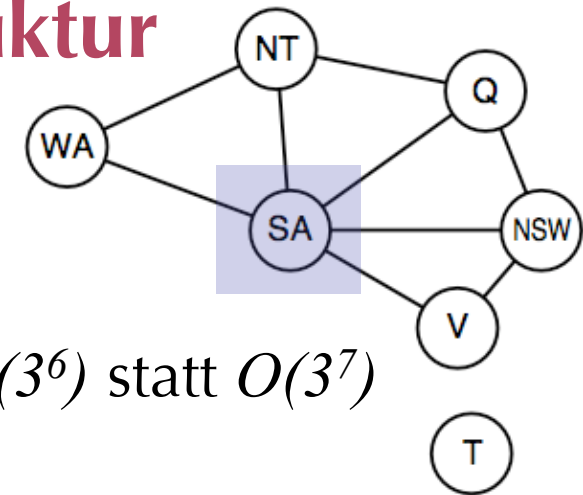


Ausnutzen der Netzstruktur



Im Australien-Färbeproblem Tasmanien unabhängig vom Rest färbbar

⇒ Lösungskomplexität eigentlich $O(3^6+3^1)=O(3^6)$ statt $O(3^7)$

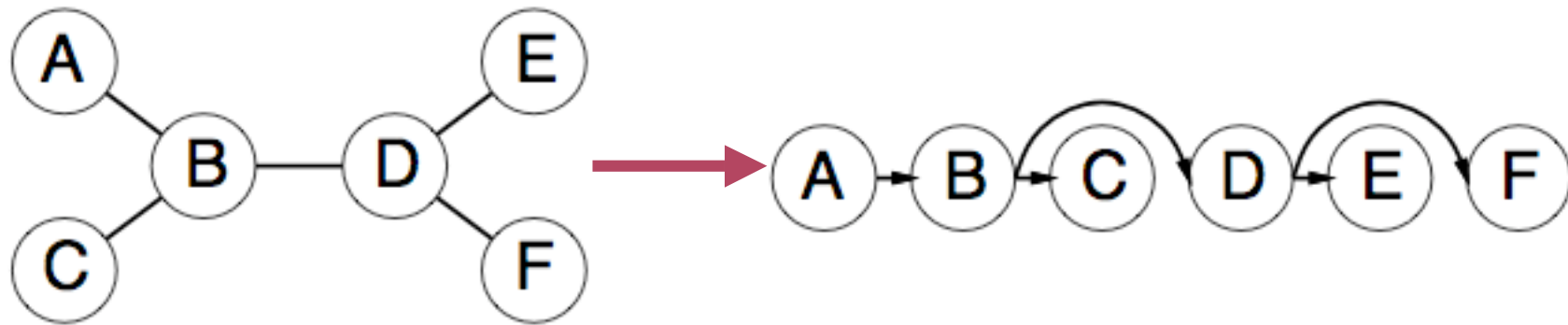
Zerlegbarkeit von Problemen als allgemeine Struktur-Heuristik mit exponentiellem Spareffekt!
(Aus $O(d^n)$ mach $O(z \times d^{n/z})$ für z „gleich große“ Zerlegungen!)
decomposability, near decomposability

Schwächere Form der Strukturvereinfachung: **Constraint-Bäume**

- Lösungsalgorithmus der Komplexität $O(nd^2)$ (nächste Folie)
- Überführe Constraint-Graph in Constraint-Baum (*cycle cutset*)
- Verschneide Baum-Lösung mit Lösung der entfernten Knoten

Lösungsalgorithmus für zyklenfreie Binär-CSPs

1. Wähle eine Variable als Wurzelknoten X_1 ;
ordne alle anderen Variablen linear als $\langle X_1, \dots, X_n \rangle$, sodass jeder Elternknoten allen seinen Nachfolgern vorangeht



2. **for** j **from** n **down to** 2 **do**
 REMOVE-INCONSISTENT(Parent(X_j), X_j);
 falls Wertebereich Parent(X_j) leer, **return** failure
3. **for** j **from** 1 **to** n weise X_j konsistent mit Parent(X_j) zu

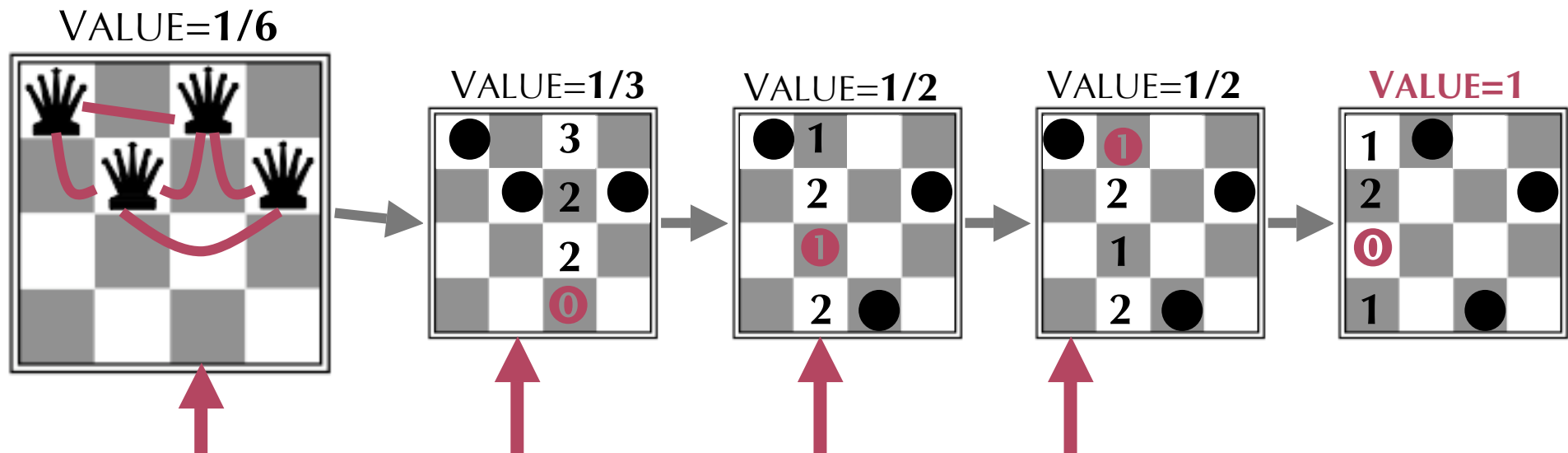
Komplexität $O(nd^2)$ (2 Läufe durch n Variable, max. d^2 Wertepaare)

Lokale Suche für CSPs: Min-Conflicts

Verwende Constraintnetz als Zustandsbewertungsfunktion bei lokaler Suche: $VALUE(n) = 1/(1 + \#Constraintverletzungen)$

Vergleiche n-Damen-Beispiel für Bergsteige-Suche

Verändere aktuellen Knoten so, dass für zufällig gewählte Variable die Anzahl der Constraintverletzungen minimiert wird.



Min-Conflicts

```
function MIN-CONFLICTS(csp,max_steps) returns a solution or failure
inputs: csp, a constraint satisfaction problem
           max_steps, the number of steps allowed before giving up

current ← an initial complete assignment for csp
for i=1 to max_steps do
    if current is a solution for csp then return current
    var ← a randomly chosen, conflicted variable from VARIABLES[csp]
    value ← a value v for var that minimizes CONFLICTS(var, v, current, csp)
    set var=value in current
return failure
```

Vergleiche Algorithmus Bergsteigen,
möglicherweise mit gleich bewertetem Nachfolger

Zusammenfassung Constraints

- Constraintnetze repräsentieren Variablen, zwischen denen Relationen gelten
- Eine Lösung besteht in einer gültigen Belegung aller Variablen
- Constraint-Algorithmen sollen eine effiziente Suche nach einer solchen Belegung realisieren
- Varianten lokaler Konsistenz stellen Zwischenschritte zu einer Lösung dar, die auch anderen Suchverfahren dienen können
- Wie bei der allgemeinen Suche gibt es systematische und lokale Verfahren, eine Lösung zu suchen
- Praktisch werden Constraint-Verfahren oft auf Zuordnungs-, Konfigurations- und Scheduling-Probleme angewendet
→ *Constraint Logic Programming, CLP*

Gliederung

1. KI im Allgemeinen und in dieser Vorlesung
 2. Heuristische Suche
 - 3. Logik und Inferenz**
 4. Wissensrepräsentation
 5. Handlungsplanung
 6. Lernen
 7. Sprachverarbeitung
 8. Umgebungswahrnehmung
- 1. Aussagenlogik**
 - 2. Resolution in der Aussagenlogik**
 - 3. Prädikatenlogik 1.Stufe**
 - 4. Resolution in der Prädikatenlogik**

1. Aussagenlogik

Logik – Was soll das?

Literatur: U. Schöning: Logik für Informatiker. Spektrum, 52000

- Logik in der Antike (Philosophie, Rhetorik):
Analyse der Korrektheit sprachlicher Schlüsse
- Mathematische (theoretische, symbolische) Logik:
Nutze Formalisierung für diese Analyse

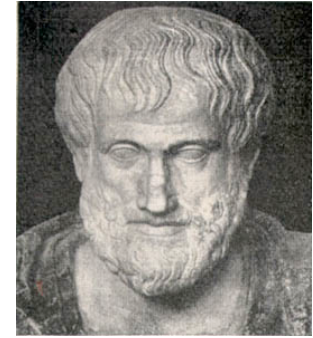
Ein Ziel

Finde Menge von Regeln („**Kalkül**“) zur syntaktischen (inhaltsunabhängigen) Manipulation von Sätzen, sodass:

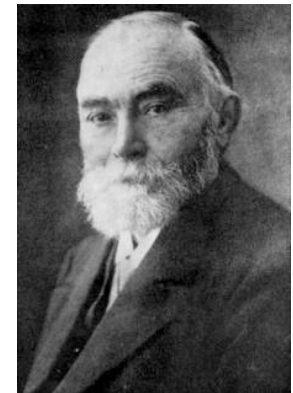
Ein Satz ist genau dann inhaltlich (allgemein-)gültig, wenn er im Kalkül ableitbar ist.

Richtung „wenn gültig, dann ableitbar“: **Vollständigkeit**

Richtung „wenn ableitbar, dann gültig“: **Korrektheit**



Aristoteles
-384 – -322



Gottlob Frege
1848–1925

Aussagen und Wahrheit

Atomare Sprachgebilde, denen man Wahrheitswerte „wahr“ oder „falsch“ zuordnen kann: **Aussagen**

- + Schnee ist weiß
- + Fritz hat einen Onkel, der Lehrer ist und in Kiel wohnt, und zwar am Blücherplatz neben der Kneipe im dritten Stock
- + Gerhard Schröder ist der Papst
- Bringen Sie mir mal ein Bier!

Zweiwertigkeit

- Eine Aussage ist entweder wahr oder falsch (ausgeschlossenes Drittes)
- Eine Aussage ist nie gleichzeitig wahr und falsch (ausgeschlossener Widerspruch)

Ableitungsregeln

... erzeugen aus einer Menge zusammengesetzter Aussagen neue (atomare oder zusammengesetzte) Aussagen

<u>wenn P dann Q</u>	P	<u>wenn es regnet, dann ist die Straße nass</u>	es regnet
Q		die Straße ist nass	

Modus ponens (seit Aristoteles): **korrekt**

<u>wenn P dann nicht Q</u>	<u>wenn es warm ist dann ist es nicht kalt</u>
<u>wenn Q dann nicht P</u>	<u>wenn es kalt ist dann ist es nicht warm</u>

Kontraposition: korrekt

<u>P oder Q</u>	P	<u>es regnet oder die Schranke ist zu</u>	es regnet
nicht Q		die Schranke ist nicht zu	

nicht korrekt!! (je nach Interpretation des „oder“)

Syntax (BNF) der Aussagenlogik

Sentence \rightarrow *AtomicSentence* | *ComplexSentence*

AtomicSentence \rightarrow **True** | **False** | *Symbol*

Symbol \rightarrow **P** | **Q** | **R** | ...

Complex Sentence \rightarrow \neg (*Sentence*)

| (*Sentence* \wedge *Sentence*)

| (*Sentence* \vee *Sentence*)

Abk. für $(\neg(P) \vee Q)$ | (*Sentence* \Rightarrow *Sentence*)

Abk. für $((P \Rightarrow Q) \wedge (Q \Rightarrow P))$ | (*Sentence* \Leftrightarrow *Sentence*)

Junktoren:

Negation, Konjunktion, Disjunktion, Implikation, Ko-Implikation