Kapitel 3: Constraints

Teil 2

(Dieser Foliensatz basiert teilweise auf Material von Mirjam Minor, Humboldt-Universität Berlin, WS 2000/01)

Nachtrag 1/2 zu Folie 11 von Teil 1

Backtracking-Suche

Die Zustandsübergangsoperatoren sind kommutativ, d.h. es macht keinen Unterschied, ob wir zuerst $x \leftarrow v$ zuweisen und dann $y \leftarrow w$ oder umgekehrt.

Daher beschränken wir uns bei jeder Verzweigung im Suchbaum auf die verschiedenen Belegungen von nur einer Variablen.

Sind alle Belegungen dieser Variablen unerfüllbar, gehen wir eine Ebene im Suchbaum zurück $\rightarrow$ Backtracking.

Nachtrag 2/2 zu Folie 11 von Teil 1

Weitere Heuristiken

Heuristik der maximal eingeschränkten Variablen: Bevorzuge die Variable mit dem größten noch möglichen Wertebereich. Kann der H. des minimalen Konflikts vorgeschaltet werden.

Heuristik der minimalen Breitenordnung: Wähle in dem noch verbleibenden Constraint-Graphen die Ecke mit dem höchsten Eckengrad. Kann bspw. an die anderen Heuristiken angeschlossen werden.

Constraint-Propagierung

Idee: Constraints propagieren und so sukzessive die Wertebereiche der Variablen einschränken (und damit den Suchraum verkleinern).

Kann als vorbereitender Schritt vor der Suche eingesetzt werden oder nach jedem Suchschritt durchgeführt werden.

Bsp. an Tafel

Das Constraintnetz $\mathcal{C} = \{C_1, C_2, C_3\}$ sei wie folgt gegeben:
$V = \{x, y, z, w\}$, $Dom(x) = Dom(y) = Dom(z) = Dom(w) = \{1, 2, 3, 4\}$
$C_1 = \{[x, y] : x > y\}$ über $V_1 = \{x, y\}$
$C_2 = \{[y, z] : y > z\}$ über $V_2 = \{y, z\}$
$C_3 = \{[z, w] : z > w\}$ über $V_3 = \{z, w\}$

– Was passiert bei Tiefensuche?

Konsistenz von Problemen

Wir betrachten nur zweistellige CSPs, da sich alle anderen darauf zurückführen lassen.

Eine Variable a heisst *konsistent*, wenn jede ihrer Belegungen alle einstelligen Constraints der Form $c(x)$ löst. Ein CSP heisst *knotenkonsistent* (oder *1-konsistent*), wenn jede Variable konsistent ist.

Ein zweistelliges Constraint $c(x, y)$ heisst konsistent, wenn sich jede Belegung $\{x \leftarrow v\}$ zu $\{x \leftarrow v, y \leftarrow w\}$ ohne Verstoß gegen $C(x, y)$ erweitern lässt. Ein CSP heisst *kantenkonsistent* (oder *lokal konstistent*), wenn es knotenkonsistent ist und wenn jedes zweistellige Constraint konsistent ist.

Constraint-Propagierung

Ziel der Propagierung: Konstruktion eines kleineren kantenkonsistenten CSPs durch Streichung von Werten aus den Wertebereichen.

Ist ein Constraint $c(x, y)$ nicht konsistent, gibt es (nach Def.) eine Belegung $\{x \leftarrow v\}$, die sich durch kein w zu $\{x \leftarrow v, y \leftarrow w\}$ ohne Verstoß gegen $C(x, y)$ erweitern lässt. v kann dann aus dem Wertebereich von x gestrichen werden.

(Passiert dies innerhalb eines Suchbaums, muss ggf. beim Backtracking das Streichen rückgängig gemacht werden.)

$\longrightarrow$ Tafelbeispiel

Ergebnis der Constraint-Propagierung

Gelingt es nicht, ein kantenkonsistentes CSP zu erzeugen, ist das ursprüngliche Problem unlösbar.

Gelingt es, so folgt nicht notwendigerweise, dass das Problem lösbar ist. Aber zumindest ist der Suchraum verringert worden.

Kantenkonsistenz-Algorithmus AC–3

Erzeugt ein CSP mit möglicherweise reduzierten Wertebereichen:

Eingabe: *csp*, ein binäres CSP mit Variablen $\{x_1, x_2, \ldots, x_n\}$

Lokale Var.: *queue*, eine Schlange mit Kanten; enthält anfangs alle Kanten von *csp*

While *queue* nicht leer do
- $(x_i, x_j) \leftarrow$ Remove-first(*queue*)
- if Remove-inconsistent-values(x_i, x_j) then
  - for each x_k in Neighbors(x_i) do
    - füge (x_k, x_i) zur *queue* hinzu.

function Remove-inconsistent-values (x_i, x_j) returns *true* bei Entfernung eines Wertes

  removed $\leftarrow$ *false*

  for each v in $dom(x_i)$ do

    wenn für kein $w \in dom(x_j)$ das Paar (v, w) das Constraint zwischen x_i und x_j erfüllt

      dann lösche v aus $dom(x_i)$; *removed* $\leftarrow$ *true*

  return *removed*

Bildverstehen als Constraint-Problem

Aufgabe: Zweidimensionales Abbild (Linienzeichnung) eines ebenflächig begrenzten dreidimensionalen Körpers (Polyeder) interpretieren.

Voraussetzungen an Zeichnung:

- keine Schatten oder Bruchlinien
- verdeckte Kanten sind nicht sichtbar
- alle Eckpunkte sind Schnittpunkte genau drei aufeinandertreffender Flächen
- "Allgemeiner Beobachtungstandpunkt": bei geringen Ortsveränderungen darf kein Schnittpunkt seinen Typ wechseln

Vom Objekt bis zur internen Repräsentation sind in der Praxis mehrere Schritte nötig: Objekt → Bild → Pixel → Kontrastbereiche → Kanten → interne Repräsentation

Zweck:
- Existiert eine physikalisch mögliche Interpretation ?
- zulässige Interpretation finden
- Objekte identifizieren (insbes. Aussenkanten)

Voraussetzungen sind in Praxis nicht immer erfüllt (z.B.: Spitze Cheops-Pyramide, Würfel frontal gesehen)

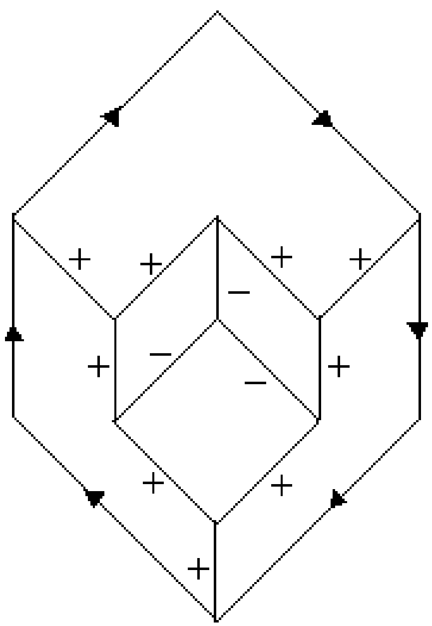
Mögliche Linientypen

Begrenzungslinien: Linien, bei denen eine der angrenzenden Flächen verdeckt ist – die äußeren Kanten des Objekts. Kennzeichnung mit einem Pfeil „→“. Die Pfeile der Begrenzungslinien sind so gerichtet, daß die Körperfläche immer rechts liegt.

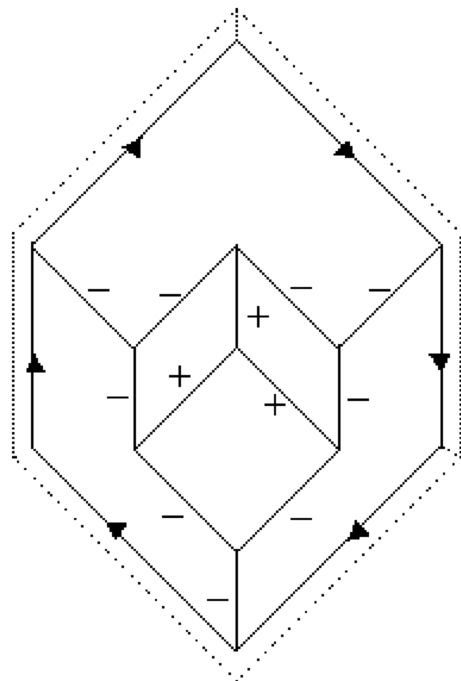
Innenlinien: Kanten im Inneren des Objekts

- **Konkave Linien:** Beide Begrenzungsflächen schließen den Beobachtungsstandpunkt ein. Ein Beispiel wäre der Blick in ein geöffnetes Buch. Kennzeichnung mit einem „–“.
- **Konvexe Linien:** Beide Grenzflächen sind vom Beobachter abgewandt, wie bei einem Würfel. Kennzeichnung mit einem „+“.

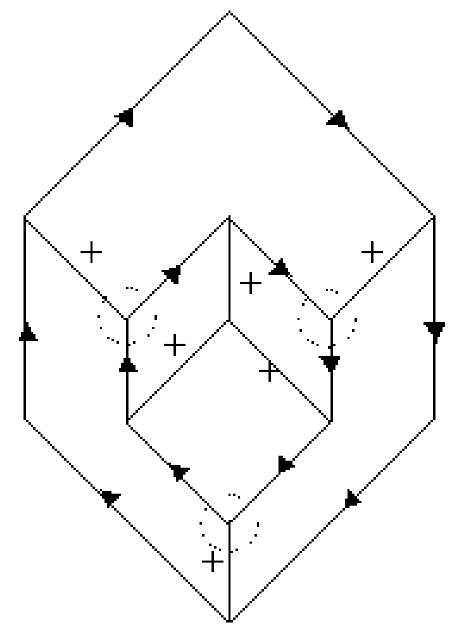
Beispiele zur Beschriftung



Großer Würfel mit herausgeschnittenem kleinen Würfel



Kleiner Würfel in einer Ecke (hier fehlt eigentlich der umgebende Körper)



Großer Würfel mit herausstehendem kleinen Würfel (hier treffen sich in einigen Schnittpunkten mehr als 3 Flächen)

Arten von Schnittpunkten und ihre Beschriftung

4 Typen von Schnittpunkten (aufgrund der Voraussetzungen)

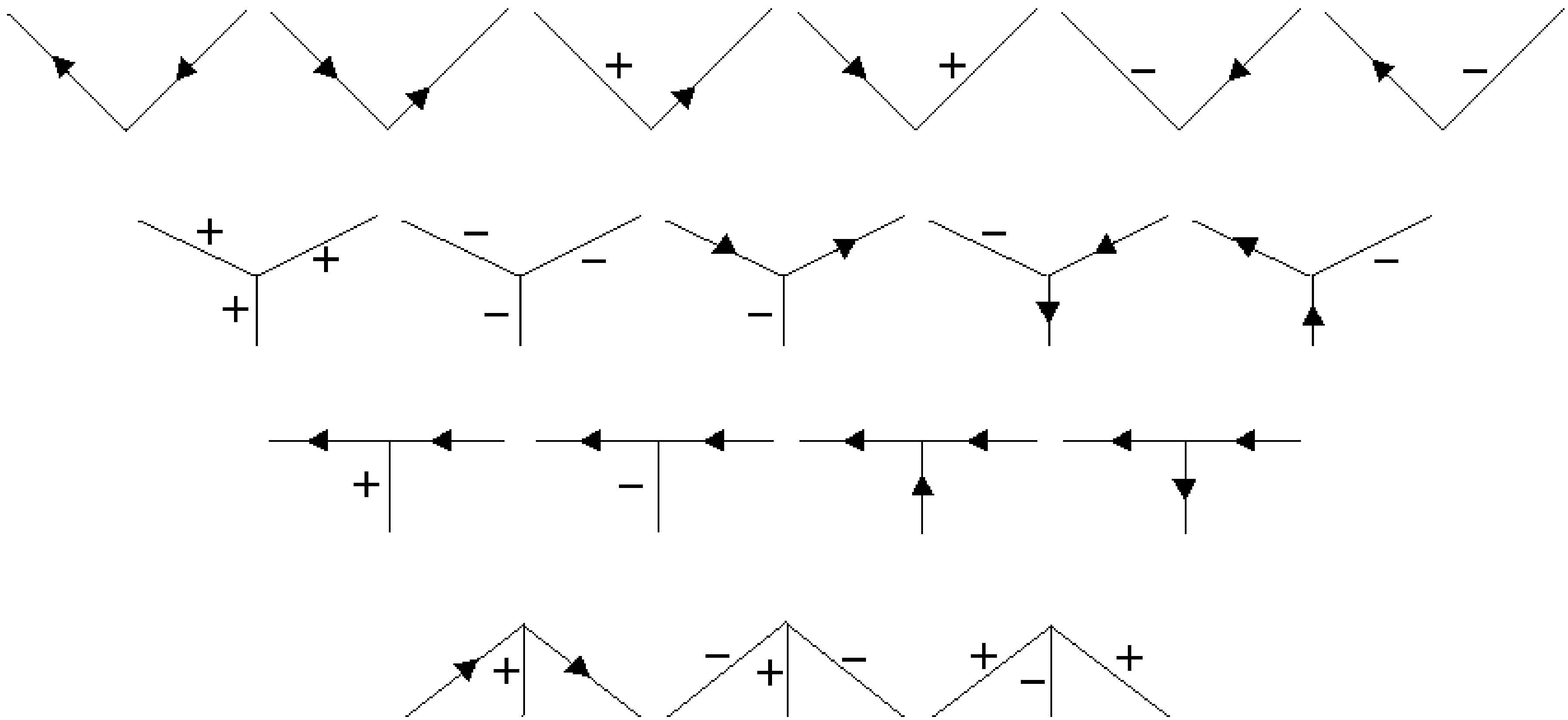
Ecke | T-Stück | Gabel | Pfeil

3 Typen von Linien

außen | außen | konvex | konkav

Es gibt für jede Kante jedes Knotens vier mögliche Beschriftungen. Insgesamt also $4^2 + 4^3 + 4^3 + 4^3 = 208$ Möglichkeiten.

Die 18 physikalisch möglich Beschriftungen



Beschriftungsverfahren

Für eine gegebene 2-D Zeichnung soll eine konsistente Beschriftung der Kanten gefunden werden, d.h.:

- Ecken gemäß den 18 zulässigen Typen
- längs einer Kante darf sich die Beschriftung nicht ändern

Als Constraint-Problem: Kanten sind die Variablen mit Wertebereich {„←“, „→“, „−“ , „+“}; Constraints ergeben sich an den Ecken (zulässige Typen)

Für realistische Bilder existiert stets (mindestens) eine konsistente Beschriftung. Es gibt manchmal aber auch konsistente Beschriftungen bei unrealistischen Bildern.

In komplexeren Bildern weitere Constraints durch Licht/Schatten und Bruchlinien.

Einfaches Suchverfahren

1. Auswahl einer Ecke, diese beschriften
2. Fortlaufend Nachbarecken beschriften, solange dies möglich ist.
3. Beim Auftreten von Widersprüchen Backtracking (alternative Ecken und Beschriftungen in 2.)

Verfahren von Waltz

1. Alle Ecken mit den dort möglichen Beschriftungstypen versehen
2. Fortlaufend jeweils Paare von Nachbarecken vergleichen:
   Bei beiden Ecken die nicht miteinander vereinbaren Beschriftungstypen entfernen, bis keine Änderungen mehr möglich sind

Zeitliches Schliessen als Constraint-Problem

Aufgabe: für eine Menge von Aussagen über zeitliche Zusammenhänge prüfen ob sie konsistent sind bzw. weitere, nicht explizit gegebene Zusammenhänge finden.

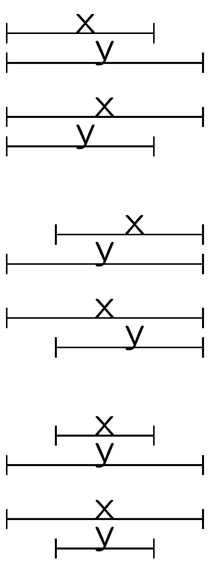
Beispiel: Gegeben seien die Zeitintervalle A, B, C, D mit folgenden Beziehungen:

- A beginnt während B
- B beginnt nicht vor C
- B liegt zeitlich völlig nach D
- C und D beginnen gleichzeitig

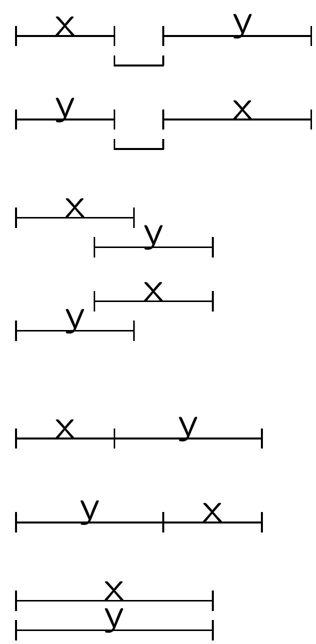
Was läßt sich über die Beziehung zwischen A und D sagen ?

Das Intervallkalkül von Allen

Die folgenden Beziehungen zwischen Zeitintervallen sollen als Relationen über Intervallen definiert werden.



$s(x,y)$	STARTS(x,y)
$s_i(x,y)$	Inverses
$f(x,y)$	FINISHES(x,y)
$f_i(x,y)$	Inverses
$d(x,y)$	DURING(x,y)
$d_i(x,y)$	Inverses



$b(x,y)$	BEFORE(x,y)
$b_i(x,y)$	Inverses
$o(x,y)$	OVERLAPS(x,y)
$o_i(x,y)$	Inverses
$m(x,y)$	MEETS(x,y)
$m_i(x,y)$	Inverses
$e(x,y)$	EQUALS(x,y)

Literatur

G. Görz (Hrsg.): Handbuch der KI

Suche: H. Kaindl: Problemlösen durch heuristische Suche in der AI

Constraints allgemein: E. Rich, C. Knight: Artificial Intelligence

Interpretation von Zeichnungen: P.H. Winston: Künstliche Intelligenz