MAXPLAN

Ein neuer Ansatz für probabilistisches Planen
(Stephen M. Majercik & Michael L. Littman 1998)

Kurzfassung / Folien

Eric Auer

8. April 2000

Das Beste aus beiden Welten

Klassische KI-Planer:
Planen in grossen deterministischen Welten

Operations research Techniken:
Planen in kleinen probabilistischen Welten

MAXPLAN:
Probabilistisches Planproblem als **E-MAJSAT** Erfüllbarkeitsproblem lösen $\Rightarrow$ effiziente Verarbeitung

E-MAJSAT Problem:
Maximiere die Erfüllungswahrscheinlichkeit einer KNF-Formel.

Ansatz bei MAXPLAN:
Wahlvariablen repräsentieren Pläne, Zufallsvariablen Wahrscheinlichkeiten.

Problemrepräsentation bei MAXPLAN

Probabilistisches Planproblem:

$< S, s_0, A, G >$

S: Zustände

s_0: Startzustand

A: Aktionen

G: Zielzustände ($G \subseteq S$)

Eingabe:

Sequentieller Effektbaum (**ST**), wird in ein E-MAJSAT Problem umgeformt.

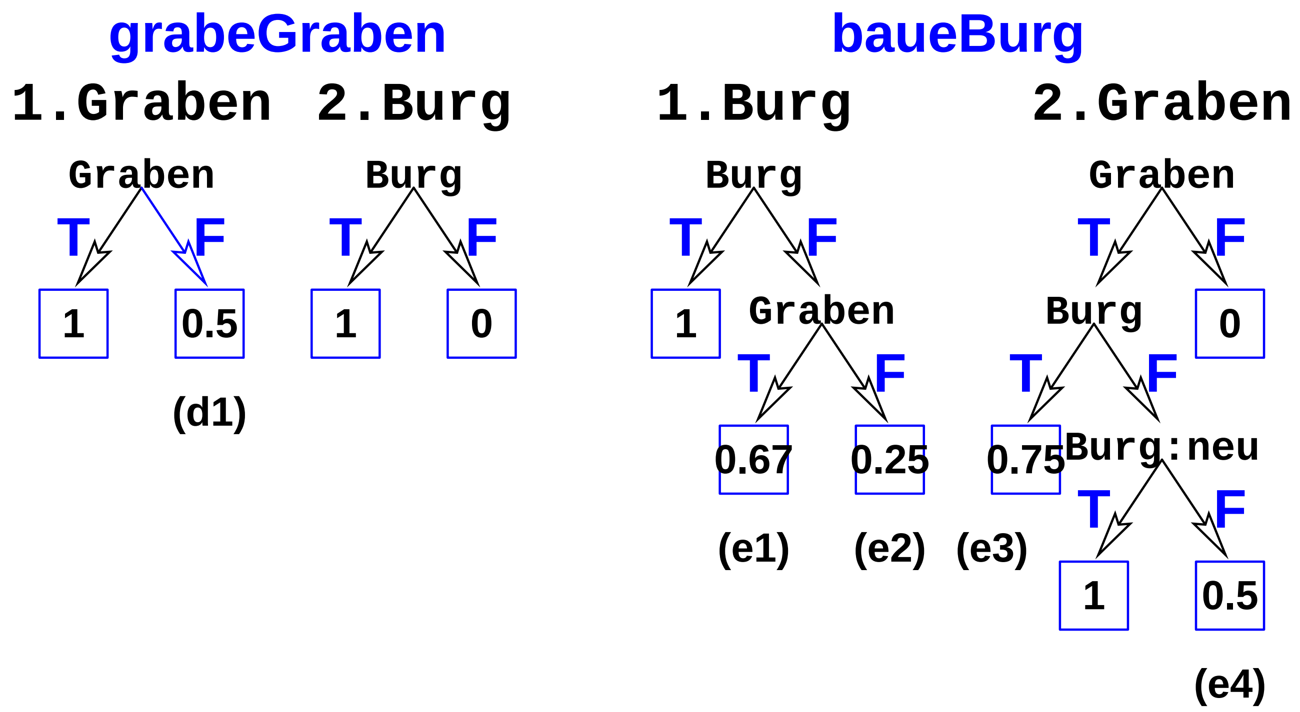
Beispiel „Sandburg-67":

Aufgabe: Baue eine Sandburg.

Tipp: Baue einen schützenden Burggraben.

$$
\begin{aligned}
S = \{ \quad & (\neg\texttt{Burg}, \neg\texttt{Graben}),\ (\neg\texttt{Burg}, \texttt{Graben}),\\
& (\texttt{Burg}, \neg\texttt{Graben}),\ (\texttt{Burg}, \texttt{Graben})\ \}\\
s_0 = (\quad & \neg\texttt{Burg}, \neg\texttt{Graben}\ )\\
G = \{ \quad & (\texttt{Burg}, \neg\texttt{Graben}),\ (\texttt{Burg}, \texttt{Graben})\ \}\\
A = \{ \quad & \mathsf{baueBurg},\ \mathsf{grabeGraben}\ \}
\end{aligned}
$$

ST für das Sandburg-67 Problem



Beispielklausel:

Wenn es zur Zeit 0 keinen Graben gab ($\neg\mathtt{Graben}_0$), macht $\mathsf{grabeGraben}_1$ das Prädikat $\mathtt{Graben}_1$ mit Wahrscheinlichkeit $d_{1,1}$ wahr.

Als Implikation:

$$\mathsf{grabeGraben}_1 \wedge \overline{\mathtt{Graben}_0} \wedge d_{1,1} \Rightarrow \mathtt{Graben}_1$$

Fertige Klausel:

$$\overline{\mathsf{grabeGraben}_1} \vee \mathtt{Graben}_0 \vee \overline{d_{1,1}} \vee \mathtt{Graben}_1$$

$\boxed{\text{KNF Darstellung: Variablen}}$

Anpassung des ST: Nichtdeterministische Blätter durch probabilistische Knoten ersetzen:

Aus: $\boxed{0.67(e_1)}$ wird so: $e_1 \begin{array}{l} T \nearrow \boxed{1.0} \\ F \searrow \boxed{0.0} \end{array}$

Aktionen: $\forall a_i \in A$: $a_{i,1} \dots a_{i,N}$

Zustand: $\forall p_i \in P$: $p_{0,1} \dots p_{i,N}$
(P: Zustandsprädikate)

Unsicherheit: $\forall r_i \in R$: $r_{1,1} \dots r_{i,N}$

Macht zusammen:

$$V = AN + P(N+1) + RN$$

Variablen in der KNF Formel (E-MAJSAT Form)

Gesamtzahl Klauseln:

$$P + P + (1 + \binom{A}{2})N + 2N \sum_{i=1}^{A} L_i$$

$(\binom{A}{2} = \frac{1}{2}(A^2 + A)\ )$

KNF Darstellung: Beispiel

KNF für Sandburg-67 mit 1 Zeitschritt:
(*nicht auswendiglernen* $\ddot{\smile}$ *!*)

$(\overline{\mathtt{Graben}_0}) \;\wedge\; (\overline{\mathtt{Burg}_0}) \;\wedge$ $\quad\leftarrow s_0$
$(\mathtt{Burg}_1) \;\wedge$ $\quad\leftarrow$ Ziel

$(\mathrm{grabeGraben}_1 \vee \mathrm{baueBurg}_1) \;\wedge$ $\quad\leftarrow$ min. 1 Aktion
$(\overline{\mathrm{grabeGraben}_1} \vee \overline{\mathrm{baueBurg}_1}) \;\wedge$ $\quad\leftarrow$ nie 2 Aktionen

$(\overline{\mathrm{grabeGraben}_1} \vee \overline{\mathtt{Graben}_0} \vee \mathtt{Graben}_1) \;\wedge$
$(\overline{\mathrm{grabeGraben}_1} \vee \mathtt{Graben}_0 \vee \overline{d_{1,1}} \vee \mathtt{Graben}_1) \;\wedge$
$(\overline{\mathrm{grabeGraben}_1} \vee \mathtt{Graben}_0 \vee d_{1,1} \vee \overline{\mathtt{Graben}_1}) \;\wedge$

$(\overline{\mathrm{baueBurg}_1} \vee \overline{\mathtt{Graben}_0} \vee \overline{\mathtt{Burg}_0} \vee \overline{e_{3,1}} \vee \mathtt{Graben}_1) \;\wedge$
$(\overline{\mathrm{baueBurg}_1} \vee \overline{\mathtt{Graben}_0} \vee \overline{\mathtt{Burg}_0} \vee e_{3,1} \vee \overline{\mathtt{Graben}_1}) \;\wedge$

$(\overline{\mathrm{b.B.}_1} \vee \overline{\mathtt{Graben}_0} \vee \mathtt{Burg}_0 \vee \overline{\mathtt{Burg}_1} \vee \mathtt{Graben}_1) \;\wedge$
$(\overline{\mathrm{b.B.}_1} \vee \overline{\mathtt{Graben}_0} \vee \mathtt{Burg}_0 \vee \mathtt{Burg}_1 \vee \overline{e_{4,1}} \vee \mathtt{Graben}_1) \;\wedge$
$(\overline{\mathrm{b.B.}_1} \vee \overline{\mathtt{Graben}_0} \vee \mathtt{Burg}_0 \vee \mathtt{Burg}_1 \vee e_{4,1} \vee \overline{\mathtt{Graben}_1}) \;\wedge$

$(\overline{\mathrm{baueBurg}_1} \vee \mathtt{Graben}_0 \vee \overline{\mathtt{Graben}_1}) \;\wedge$

$(\overline{\mathrm{grabeGraben}_1} \vee \overline{\mathtt{Burg}_0} \vee \mathtt{Burg}_1) \;\wedge$
$(\overline{\mathrm{grabeGraben}_1} \vee \mathtt{Burg}_0 \vee \overline{\mathtt{Burg}_1}) \;\wedge$

$(\overline{\mathrm{baueBurg}_1} \vee \overline{\mathtt{Burg}_0} \vee \mathtt{Burg}_1) \;\wedge$ $\quad\leftarrow$ 1 ST-Pfad

$(\overline{\mathrm{baueBurg}_1} \vee \mathtt{Burg}_0 \vee \overline{\mathtt{Graben}_0} \vee \overline{e_{1,1}} \vee \mathtt{Burg}_1) \;\wedge$
$(\overline{\mathrm{baueBurg}_1} \vee \mathtt{Burg}_0 \vee \overline{\mathtt{Graben}_0} \vee e_{1,1} \vee \overline{\mathtt{Burg}_1}) \;\wedge$

$(\overline{\mathrm{baueBurg}_1} \vee \mathtt{Burg}_0 \vee \mathtt{Graben}_0 \vee \overline{e_{2,1}} \vee \mathtt{Burg}_1) \;\wedge$
$(\overline{\mathrm{baueBurg}_1} \vee \mathtt{Burg}_0 \vee \mathtt{Graben}_0 \vee e_{2,1} \vee \overline{\mathtt{Burg}_1})$

Problemlöser: Algorithmus

Ausgangspunkt: DPLL-Algorithmus für Erfüllbarkeitsprobleme
(Davis-Putnam-Logemann-Loveland)

Aufgabe: Für alle Pläne die Erfolgswahrscheinlichkeit messen $\Rightarrow$ Sieger?

Strategie: Suche in Binärbaum mit Pruning.

Begriffe:

Aktive Variable:
hat noch keinen festgelegtem Wert.

Aktive Klausel: noch nicht erfüllte Klausel.

Aktuelle KNF Teilformel:
bestimmt durch aktive Klauseln / Variablen.

Wert einer KNF Teilformel:
Maximum der Erfolgswahrscheinlichkeiten ($\sum_S \ldots$) für alle möglichen Pläne D.

$$\rightarrow \quad \max_D \sum_S \Big(\prod_{v_i \in CH} \pi_i^{v_i} * (1-\pi_i)^{1-v_i} \Big)$$

Problemlöser: DPLL-Pruningregeln

Unwichtige Variablen belegen
*(***IRR***, irrelevant variable removal):*
Aktive Variable ohne aktive Klauseln $\Rightarrow$ true.

Atomare Klauseln erfüllen
*(***UNIT***, unit propagation):*
Aktive Variable in atomarer Klausel: trivial.

Eindeutige Variablen entscheiden
*(***PURE***, purification):*
Aktive Variable in einheitlicher Richtung ist auch naheliegend.

Planbewerter: Heuristiken

Motivation:
Struktur des Binärbaumes optimieren.

Vergleich:
Drei Verzweigungsheuristiken wurden an einem gegebenen Sandburg-67 Plan verglichen.

Zufall (random):
Zufällige Ordnung der Variablen.

Gier (sats most clauses):
Versuche viele Klauseln zu erfüllen.

Zeit (time-ordered):
Wähle die zeitlich nächste aktive Variable.

Planbewerter: Erste Optimierungen

Messergebnisse:

„Zeit" ist die beste Heuristik.

Heuristik	in 1 Minute max.	in 10 Minuten
Zufall	8 Schritte	10 Schritte
Gier	11 Schritte	13 Schritte
Zeit	15 Schritte	19 Schritte

Weniger Pruning:

Keine gute Idee — schon 5-Schritt-Pläne brauchen dann Tage zur Bewertung.

Lernen von „ENUM":

Memoisierung der Teilformeln bringt grosse Vorteile $\Rightarrow$ 110 statt 18 Schritte in X Sekunden.

Planbewerter: Weitere Optimierungen

Pruning bremst Memoisierung:

Pruning-regeln	Planlänge in 20 Sekunden	Rechenzeit für 100 Schritte
alle DPLL	$\approx$ 50	$\approx$ 140s
ohne IRR	$\approx$ 60	$\approx$ 90s
nur UNIT	$\approx$ 95	$\approx$ 25s
keine	$\approx$ 100	$\approx$ 20s

(*„nur UNIT" ist oft besser als „kein Pruning".*)

Das Ergebnis:
MAXPLAN findet jetzt in 12 Sekunden den optimalen 10-Schritt-Plan für Sandburg-67 (*Erfolgswahrscheinlichkeit 97%*):

G-B-G-B-B- G-B-G-B-B
(*B: baueBurg*, *G: grabeGraben*)

Vergleich mit anderen Planern

Die Kandidaten:

BURIDAN
Klassisches Partial-order Planen für probabilistische Probleme.

ENUM
Planaufzählung und dynamische Programmierung für die Planbewertung.

POMDP
Löst teilweise beobachtbare MDP mit inkrementellem Pruning effektiv.

Das Ergebnis:

BURIDAN:
MAXPLAN ist gut 10 mal schneller:

MAXPLAN ohne Pruning:	0.7 Sekunden
MAXPLAN mit Pruning:	0.3 Sekunden
BURIDAN, Vorwärts:	6.9 Sekunden
BURIDAN, Abfrage:	64.9 Sekunden
BURIDAN, Netzwerk:	152.0 Sekunden
BURIDAN, Rückwärts:	6736.0 Sekunden

Welcher Planer skaliert besser?

Leichte Gegner?

ENUM / POMDP zählen alle 2^P Zustände auf, ENUM bewertet alle A^N Pläne.

Nicht immer!

POMDP schlägt MAXPLAN klar bei Sandburg-67, weil es die Unbeobachtbarkeit besser nutzen kann.

Zeit:	MAXPLAN:	POMDP:	ENUM:
0.1 Sek.	7 Schritte	13 Schritte	6 Schritte
10 Sek.	9 Schritte	16 Schritte	50 Schritte

Ein Traum für MAXPLAN:

MAXPLAN löst **Bomben-entschärfen** für 1 bis 10 Pakete in annähernd linearer Zeit.

MAXPLAN:	etwa konstant $\frac{1}{10}$ Sekunde.
POMDP:	In $\frac{1}{2}$ Sekunde 5 Pakete.
ENUM:	In $\frac{1}{2}$ Sekunde 7 Pakete, bis 5 Pakete schneller als MAXPLAN.

Zukünftige Verbesserungen

Skalierung:
Noch mehr Aufgaben sind zu untersuchen.

KNF:
Bessere Repräsentation des Problemes,
elegantere Speicherung der Formel.

Verzweigungsheuristik:
Die Zeit-Heuristik ist manchmal nutzlos.
Natürliche Zeitstruktur ausnutzen.

Memoisierung:
Beim Speicherverbrauch wählerischer sein
und nicht alle Zwischenergebnisse speichern
$\Rightarrow$ **Cache**.

Zukünftige Erweiterungen

Verstärkungslernen:
Kann Verzweigungs- und Cache-Heuristiken verbessern. Problem: Zu wenige typische Probleme als Trainingsaufgaben.

Näherungsverfahren:
Schnell eine Lösung schätzen und später verfeinern, **Anytime-Planen**.

Belief Networks:
Die KNF-Formel als Belief Network betrachten und lösen.

Beobachtbarkeit berücksichtigen:
In beobachtbaren Problemwelten durch Beobachtung bessere Pläne finden.