

Informatik-Seminar SS2010: Methoden der KI

Thema: Funktionsweise und Anwendung eines JTMS

Vortragender: Frederik Ladewig

# 0. Übersicht

---

- 1. Einführung TMS
- 2. JTMS Einführung
- 3. JTMS Algorithmus
- 4. Fazit

# 1. Einführung TMS

Beispiel Lichtschalter:

$s$  = Schalter ist geschlossen

$\neg s$  = Schalter ist offen

$k$  = Kabel ist in Ordnung

$\neg k$  = Kabel ist defekt

$l$  = Lampe ist an

$\neg l$  = Lampe ist aus

Regeln:

$$s \rightarrow l$$

$$s \wedge \neg k \rightarrow \neg l$$

## 2. JTMS Einführung

---

### Truth Maintenance Network

Ein truth maintenance network  $T$  besteht aus einer Menge von Knoten (nodes)  $N$  und einer Menge von Begründungen (justifications)  $J$ .

$$T = (N, J)$$

### Knoten

Ein Knoten repräsentiert eine elementare Aussage und wird in einem TMN als Rechteck dargestellt.

### Begründung

Begründungen werden in einem TMN als Kreis dargestellt. Eine Begründung  $J$  besteht aus einer Menge von IN-Knoten, einer Menge von OUT-Knoten und einer Konsequenz  $n$ .

$$J = (I | O \rightarrow n)$$

### IN-Liste

Menge der Knoten, die akzeptiert werden müssen, damit auch die Begründung akzeptiert werden kann.

### OUT-Liste

Eine Begründung kann nur akzeptiert werden, wenn kein Knoten der OUT-Liste akzeptiert wird.

### Prämisse

Eine Prämisse ist eine Begründung mit leerer In- und Out-Liste.

$$J = (\emptyset | \emptyset \rightarrow n)$$

## 2. JTMS Einführung

---

Beispiel:

$$T = (N, J)$$

$$N = \{A, B, C\}$$

$$J_0 = (C | A \rightarrow B)$$

$$J_1 = (C | B \rightarrow A)$$

### Modell

Ein Modell  $M$  bezüglich eines TMN  $T=(N,J)$  ist eine Menge von Knoten. Die Knoten in  $M$  werden als IN bezeichnet, alle anderen Knoten in  $N$  sind OUT.

### Gültigkeit von Begründungen

Es sein  $M$  ein Modell eines TMN. Eine Begründung  $(I|O \rightarrow n)$  heißt gültig in  $M$  wenn  $I \subseteq M$  und  $O \cap M = \emptyset$  ist.

### Fundiertes Modell

Eine Knotenmenge heißt fundiert bezüglich eines TMNs, wenn es eine vollständige Ordnung der Elemente in  $M$  gibt, so dass für jedes  $n_j \in M$  gilt: Es gibt eine in  $M$  gültige Begründung  $(I|O \rightarrow n_j) \in J$  von  $n$  derart, dass  $I \subseteq \{n_1, \dots, n_{j-1}\}$ . Eine solche Begründung wird stützende Begründung genannt.

### Abgeschlossenheit

Eine Knotenmenge heißt abgeschlossen bezüglich eines TMNs, wenn jede Konsequenz einer in dem Modell  $M$  gültigen Begründung in  $M$  enthalten ist.

### Zulässiges Modell

Als zulässiges Modell wird ein Modell bezeichnet, welches sowohl fundiert, als auch abgeschlossen ist.

## Ziel des Algorithmus

Die eigentliche Aufgabe eines JTMS-Verfahrens ist das Etablieren eines neuen zulässigen Systemzustandes, wenn neue Informationen dem System hinzugefügt wurden, durch die der bisherige Status unhaltbar geworden ist. Bei einer neu hinzugefügten Begründung versucht der JTMS Algorithmus also aus einem gegebenen zulässigen Modell ein neues zulässiges Modell zu erstellen, welches die neue Begründung berücksichtigt.

## Hinzufügen von neuen Knoten

Neue Knoten können problemlos in ein bestehendes System eingefügt werden, da sie zunächst von keiner Begründung betroffen sind und damit in jedem Fall OUT sind.

## 3. JTMS Algorithmus

---

**J(n)**

Menge der Begründungen, die n als Konsequenz haben.

**SJ(n)**

Stützende Begründung des Knotens n.

**Supp(n)**

Für die stützende Begründung  $SJ(n) = (I|O \rightarrow n)$  gilt  $Supp(n) = I \cup O$

# 3. JTMS Algorithmus

---

## **Ant(n)**

Die stützenden Knoten  $\text{Supp}(n)$  eines IN Knotens werden auch Antezedenzen genannt.

## **Cons(n)**

$\text{Cons}(n)$  umfasst alle Knoten, die  $n$  in ihrer Begründung erwähnen.

## **ACons(n)**

Affected consequences sind alle Knoten, die  $n$  in ihrer stützenden Begründung erwähnen.

# 3. JTMS Algorithmus

---

## **Supp\*(n)**

Die Vorfahren eines Knoten n ist der transitive Abschluss der stützenden Knoten von n.

## **Ant\*(n)**

Die Fundamente sind der transitive Abschluss der Antezedenzen von n.

## **ACons\*(n)**

Die Auswirkungen sind der transitive Abschluss der betroffenen Konsequenzen von n.

## 3. JTMS Algorithmus

---

Hinzufügen einer neuen Begründung:  $J_0 = (I_0 | O_0 \rightarrow n_0)$

1. If Label( $n_0$ ) = IN then HALT;  
If  $J_0$  ungültig in M then HALT;
2. If Acons( $n_0$ ) = leer then  
Füge  $n_0$  als größtes Element in M ein; HALT
3. //Label( $n_0$ ) = out  
// $J_0$  ist gültig in M  
// Acons( $n_0$ ) ist nicht leer  
L := Acons\*( $n_0$ ) +  $n_0$   
Markiere jeden Knoten in L mit 'unknown'

## 3. JTMS Algorithmus

---

- 4. for n in L do (4A)
- 4A. if Label(n)='unknown' then
  - If Es gibt eine fundiert gültige Begründung für n then
    - Label(n):=IN
    - Wende 4A auf alle Konsequenzen von n an
  - Elseif es gibt nur fundiert ungültige Begründungen für n then
    - Label(n):=OUT
    - Wende 4A auf alle Konsequenzen von n an
  - Else
    - Bestimmung des status wird auf 5 verschoben

## 3. JTMS Algorithmus

```
5. for n in L do (5A)

5A. if Label(n)='unknown' then
    If Es gibt eine nicht-fundiert gültige Begründung J'=(I'|O'->n) then
        If Acons(n) = nicht leer then
            For n' in Acons(n)+n do
                Label(n):='unknown'
                Wende 5A auf n' an
        Else
            Label(n):= in
            For n' in O' do
                If Label(n')=unknown then
                    Label(n'):=out
            For n' in Cons(n) do
                If Label(n')=unknown then 5A
    Else //Alle Begründungen sind nicht-fundiert ungültig
        Label(n):=out
        For J' in J(n) do
            Wähle n' in I' mit Label(n')=unknown
            Label(n')=OUT
```

# 3. JTMS Algorithmus

## Dependency Directed Backtracking

$n_{\perp}$  = Widerspruchsknoten

$MaxAnn(n_{\perp})$  = Maximale Annahmen

$J_{\perp}$  = Begründungen der maximalen Annahmen

## 3. JTMS Algorithmus

### DDB Algorithmus

1. Ist  $MaxAnn(n_{\perp}) = \emptyset$ , so liegt ein unlösbarer Widerspruch vor.
2. Wähle eine der maximalen Annahmen  $n_A \in MaxAnn(n_{\perp}) = \emptyset$   
Wähle einen (OUT-) Knoten  $n^*$  in der OUT-Liste der stützenden Begründungen von  $n_A$ .
3. Füge eine neue Begründung der Form  $(I_{\perp} | O_{\perp} \rightarrow n^{\circ})$  hinzu, wobei  
 $I_{\perp}$  = Vereinigung der IN-Listen von  $J_{\perp}$  und  
 $O_{\perp}$  = (Vereinigung der OUT-Listen von  $J_{\perp}$ )- $\{n^*\}$ .
4. JTMS Algorithmus Schritte 1-5
5. Ist  $n_{\perp}$  auch im neuen Modell IN so wende DDB auf das neue Modell an.
6. Ist der Status von  $n_{\perp}$  schließlich OUT so ist das DDB-Verfahren Erfolgreich abgeschlossen worden.