

# Markov Logik Netze (MLN)

Prof. Dr.-Ing Rüdiger Dillmann

Prof. Dr.-Ing. Marius Zöllner



**Markov Logic:**  
A Unifying Language for  
Information and Knowledge  
Management

Pedro Domingos  
Dept. of Computer Science & Eng.  
University of Washington

Joint work with Stanley Kok, Daniel Lowd,  
Hoifung Poon, Matt Richardson, Parag Singla,  
Marc Sumner, and Jue Wang



[P. Domingos]



Forschungszentrum Karlsruhe  
in der Helmholtz-Gemeinschaft



Universität Karlsruhe (TH)  
Forschungsuniversität • gegründet 1825

# .... 2008 ... University of Wahington

- The UW will lead a multi-institutional group pushing the limits of computers' ability to interpret data and ultimately predict the behavior of complex systems
- ...
- “A complex monitoring system has far too many pieces of information for any one person to look at,” said principal investigator Pedro Domingos, a UW associate professor of computer science and engineering. “This award lets us do the research to develop a system for the military to look at all the available information that might be valuable and use it to predict behavior.”
- The basic approach is the same as that of fictional detective Sherlock Holmes: using the powers of reasoning to discover the best explanation for a set of clues.

This is an archived article.

April 17, 2008

## UW to lead \$6.25 million project creating electronic Sherlock Holmes

Hannah Hickey  
News and Information

Posted under: Uncategorized



Pedro Domingos

The UW will lead a multi-institutional group pushing the limits of computers' ability to interpret data and ultimately predict the behavior of complex systems. The project, involving seven U.S. universities, has received a \$6.25 million, 5-year grant from the Department of Defense.

“A complex monitoring system has far too many pieces of information for any one person to look at,” said principal investigator Pedro Domingos, a UW associate professor of computer science and engineering. “This award lets us do the research to develop a system for the military to look at all the available information that might be valuable and use it to predict behavior.”

The basic approach is the same as that of fictional detective Sherlock Holmes: using the powers of reasoning to discover the best explanation for a set of clues. But today's reasoning can't be done by a single, pipe-smoking sleuth. The modern military has millions of possible clues, including sensors on soldiers, satellite maps, road monitors, aerial drones and written observations from reconnaissance missions. The Army Research Office that provided the grant wants to make sense of this information in order to make decisions and predict an adversary's next moves.

Although machine learning and artificial intelligence are increasingly used in Web commerce and financial investment, the way those situations are handled today is relatively simple compared to what this project will attempt.

# Inhalt der heutigen Vorlesung

- Motivation
- Hintergrund
- Markov Logik
- Inferenz
- Lernen
- Software
- Anwendungen
- Diskussion
- Literatur

Foliensatz enthält vertiefende Algorithmen („Exkurs“) → nicht prüfungsrelevant

# Markov Logik

- Grundidee MLN:  
Vereint Prädikatenlogik erster Ordnung und probabilistische (grafische) Modelle
  - Prädikatenlogik erster Ordnung → gut wenn strukturierte Information
  - Probabilistische Modelle → gut für unsichere verrauschte Informationen (ggf. unstrukturiert)
- Benötigt werden effiziente Algorithmen für
  - Inferenz (für die Auswertung von Anfragen)
  - Lernen von Modellen

# Markov Logik

- Syntax: Gewichtete logische Formeln (Prädikatenlogik -PL- erster Ordnung)
- Semantik: Template für Erzeugung von Markov Netzen
- Probabilistische Inferenz (WalkSAT, MCMC, KBMC)
- Lernen für Gewichte und Formeln (Voted perceptron, Pseudo-likelihood, Induktive Logische Programmierung, ILP)
- Software: Alchemy, Tuffy, ...
- Anwendungen:
  - Informationsgewinnung,
  - Web mining ....
  - Ontologieverfeinerung
  - Klassifikation
  - Assistenzfunktionen

# Inhalt

- Motivation
- Hintergrund
- Markov Logik
- Inferenz
- Lernen
- Software
- Anwendungen
- Diskussion
- Literatur

- Symbole: Konstanten, Variablen, Funktionen, Prädikate, Operatoren, Quantoren ...  
Bsp.: Anna,  $x$ , MotherOf( $x$ ), Friends( $x,y$ )
- Wissensbasis: Formeln
- Belegung (Grounding): Ersetzen der Variablen durch Konstanten  
Bsp.: Friends (Anna, Bob)
- Potentielle **Welten** (Modell, Interpretation): Zuordnung von Wahrheitswerten zu belegten Prädikaten

# Prädikatenlogik erster Ordnung

- Bsp.: mit der Formel  $\forall x : Smoking(x) \Rightarrow Cancer(x)$  und der Menge der Konstanten  $\{A\}$

$$\begin{aligned}\forall x : Smoking(x) \Rightarrow Cancer(x) &\equiv \\ \forall x : \neg Smoking(x) \vee Cancer(x) &\end{aligned}$$

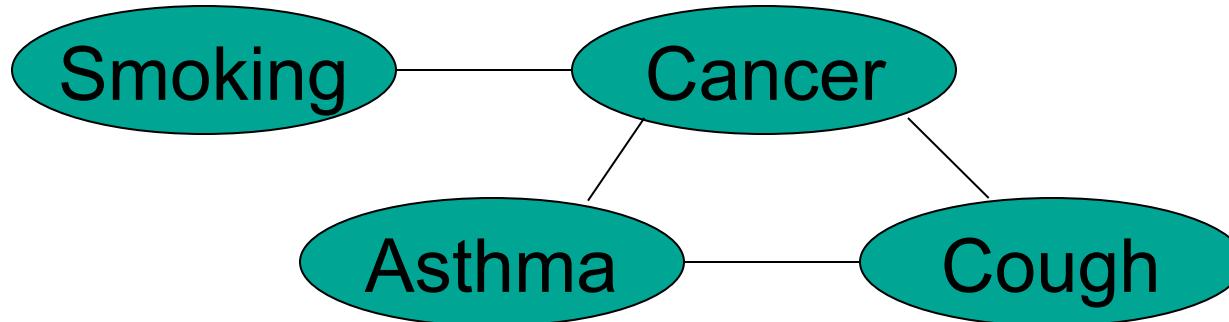
- gibt es eine Belegung und 4 potentielle Interpretationen/Welten

$$\{S(A), C(A)\} : \{\text{false}, \text{false}\}, \{\text{true}, \text{false}\}, \{\text{false}, \text{true}\}, \{\text{true}, \text{true}\}$$

- Davon eine unmögliche, d.h. die die Formel nicht erfüllt

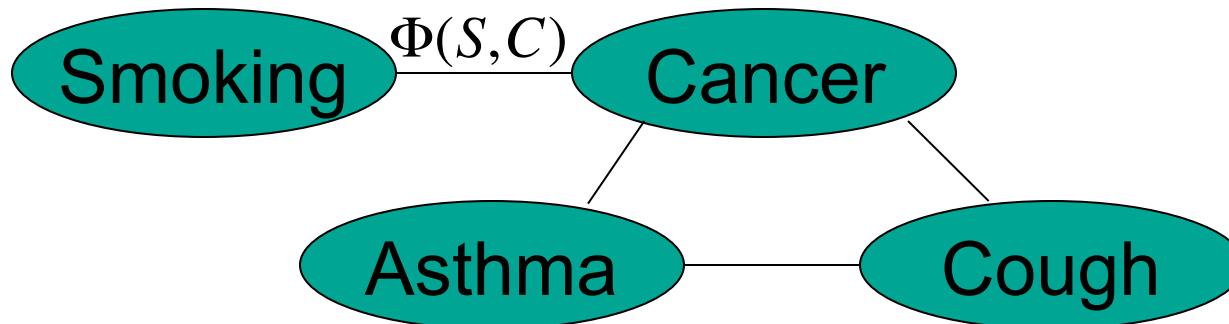
$$\{S(A), C(A)\} : \{\text{true}, \text{false}\}$$

- Gegebene Zufallsvariablen



- Durch s.g. Probabilistische Graphische Modelle (PGM) lassen sich die Verbundwahrscheinlichkeitsverteilungen über die Mengen von Zufallsvariablen beschreiben
  - Knoten repräsentieren Zufallsvariablen
  - Graph kodiert Abhängigkeiten zwischen Zufallsvariablen
- Gerichtete azyklische Graphen → Bayesche Netze (Kausalität)
- Ungerichtete Graphen → Markov Netze (Korrelationen)

## ■ Ungerichteter Graph (graphisches Modell)



- Potentialfunktionen für s.g. Cliques (vollständig verbundener Teilgraph) abh. vom Zustand der Zufallsvariablen definieren Verbundwahrscheinlichkeit:

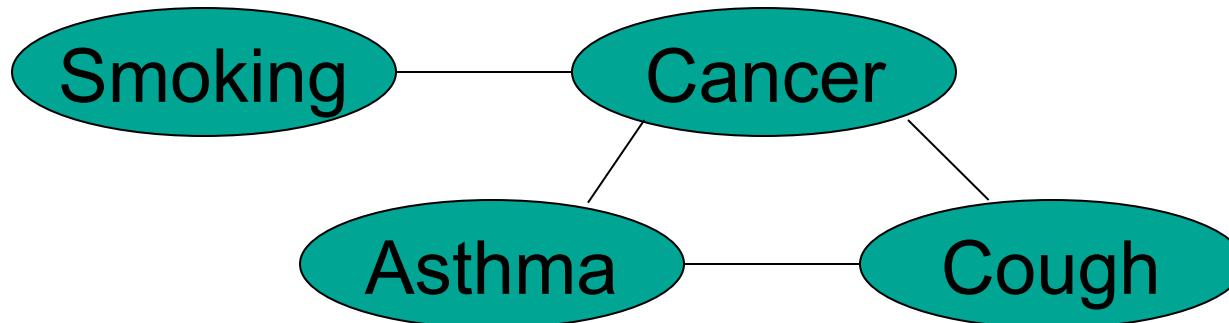
$$P(x) = \frac{1}{Z} \prod_c \Phi_c(x_c)$$

$$Z = \sum_x \prod_c \Phi_c(x_c)$$

z.B. binäre diskrete Zufallsvariablen

Smoking	Cancer	Φ(S,C)
false	false	4.5
false	true	4.5
true	false	1
true	true	4.5

## ■ Ungerichteter Graph (graphisches Modell)



## ■ Log-linear Modell (hier für binäre Zufallsvariablen), kompakte Darstellung:

$$P(x) = \frac{1}{Z} \exp\left(\sum_i w_i f_i(x)\right), \quad \begin{array}{l} f_i - \text{Zustand der } i\text{-ten Clique } c \\ w_i - \log(\Phi_c) \end{array}$$

Gewicht des Feature  $i$     Feature  $i$  (binär)

■ z.B.:  $w_1 = 1.5 = \log(4.5)$

$$f_1(\text{Smoking}, \text{Cancer}) = \begin{cases} 1 & \text{if } \neg \text{Smoking} \vee \text{Cancer} \\ 0 & \text{otherwise} \end{cases}$$

# Inhalt

- Motivation
- Hintergrund
- **Markov Logik**
- Inferenz
- Lernen
- Software
- Anwendungen
- Diskussion
- Literatur

# Grundidee: Markov Logik

- Logische Wissensbasis → strikte Entscheidungen (**hard constraints**) bzgl. Raum der potentiellen Welten
- Aufweichen (**soft constraints**):  
Wenn eine Welt eine Regel (Formel) verletzt wird diese Welt weniger wahrscheinlich aber nicht unmöglich
- Grundidee: Jede Formel wird gewichtet  
(je höher das Gewicht umso stärker ist der Einfluss dieser Formel)
- Gesamtwahrscheinlichkeit („Möglichkeit“) einer Welt:

$$P(\text{Welt}) \propto \exp\left( \sum_{\text{Formel aus Welt}} \text{Gewicht der Formel die erfüllt ist} \right)$$

# Definition Markov Logik Netz

## ■ Syntax:

Ein **Markov Logik Netz (MLN)** ist eine Menge von Tupeln  $(F_i, w_i)$ , wobei:

- $F_i$  ist eine Formel in Prädikatenlogik erster Ordnung
- $w_i$  ist eine reelle Zahl (Gewicht)

## ■ Semantik:

Mit einer Menge von Konstanten wird daraus ein Markov Netz definiert, mit:

- Je einem Knoten für jede mögliche Belegung, jedes Prädikats des MLN
- Je einem Feature für jede Belegung (grounding) jeder Formel  $F_i$  des MLN mit dem entsprechenden Gewicht  $w_i$   
(Formeln sind dabei Disjunktionsterme / Klauseln → ggf. Umformung nötig)

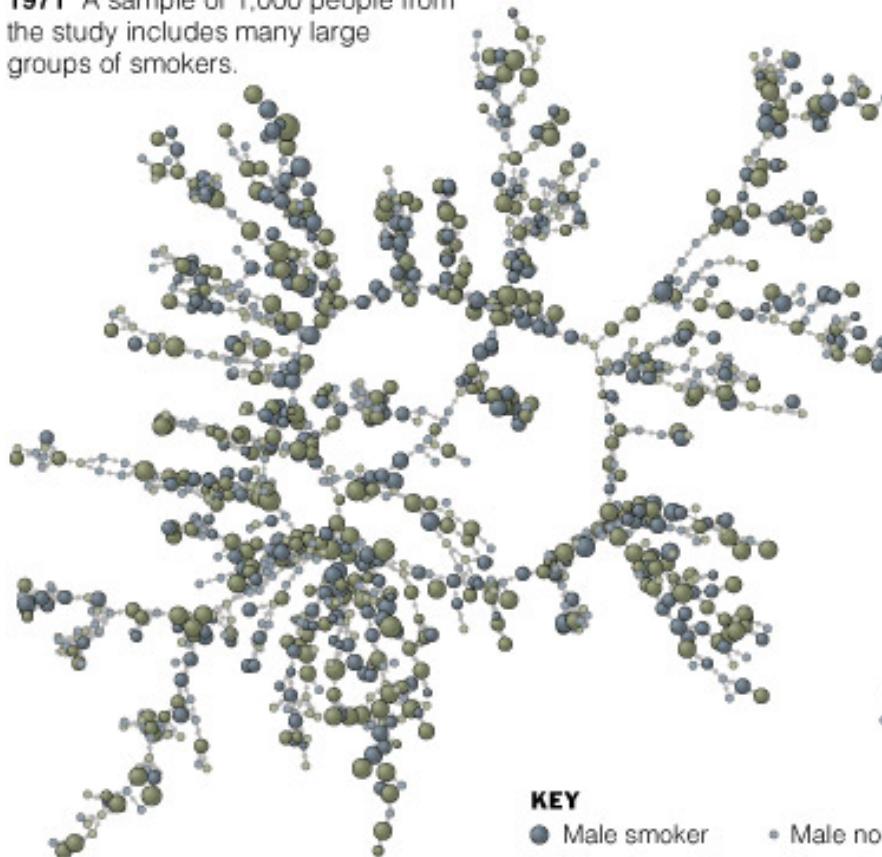
# Beispiel: Friends & Smokers

## Hintergrund

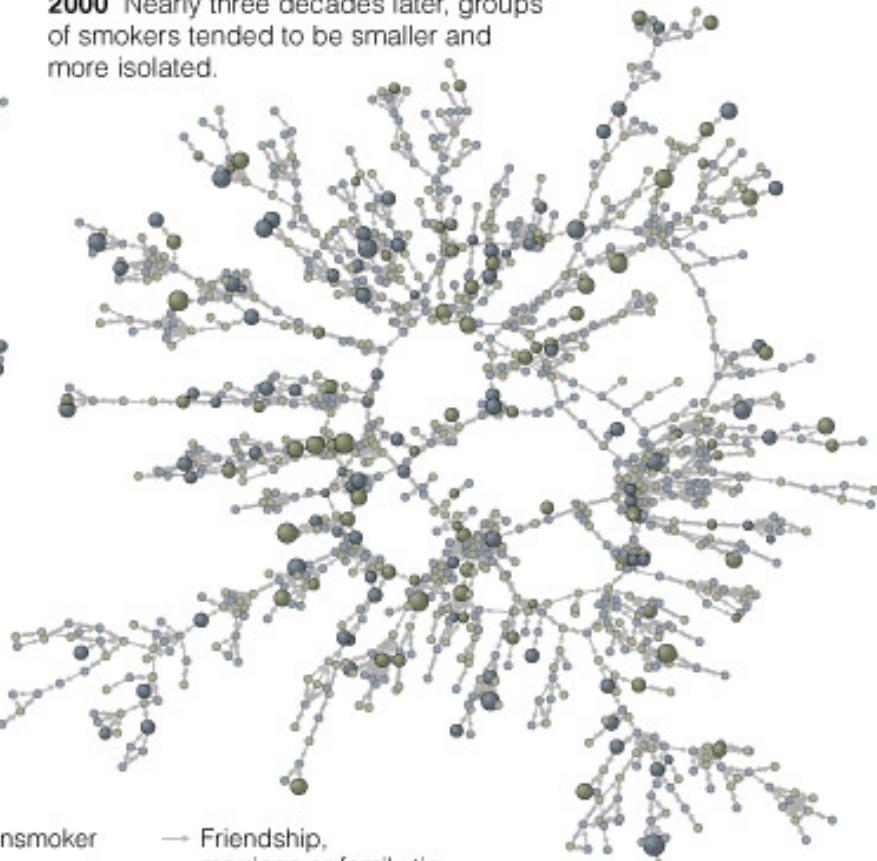
### Smoking and Quitting in Groups

Researchers studying a network of 12,067 people found that smokers and nonsmokers tended to cluster in groups of close friends and family members. As more people quit over the decades, remaining groups of smokers were increasingly pushed to the periphery of the social network.

**1971** A sample of 1,000 people from the study includes many large groups of smokers.



**2000** Nearly three decades later, groups of smokers tended to be smaller and more isolated.



Sources: New England Journal of Medicine;  
Dr. Nicholas A. Christakis; James H. Fowler

#### KEY

- Male smoker      • Male nonsmoker      → Friendship,  
● Female smoker      • Female nonsmoker      marriage or family tie

Circle size is proportional to the number of cigarettes smoked per day.

THE NEW YORK TIMES

# Beispiel: Friends & Smokers

## Zusammenhänge (Regeln)

Smoking causes cancer.

Friends have similar smoking habits.

# Beispiel: Friends & Smokers

## Prädikatenlogische Formeln

$$\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$$
$$\forall x, y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$$

„Schön“ aber nicht wirklich korrekt

→ Nicht jeder Raucher hat Krebs

→ Nicht jeder dessen Freund/-in raucht, raucht ebenfalls

# Beispiel: Friends & Smokers

## MLN: Gewichte

1.5

 $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$ 

1.1

 $\forall x, y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$ 

Stärkere Aussage

# Beispiel: Friends & Smokers

## MLN: Konstanten

1.5  $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$

1.1  $\forall x, y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$

Zwei Konstanten: **Anna** (A) und **Bob** (B)

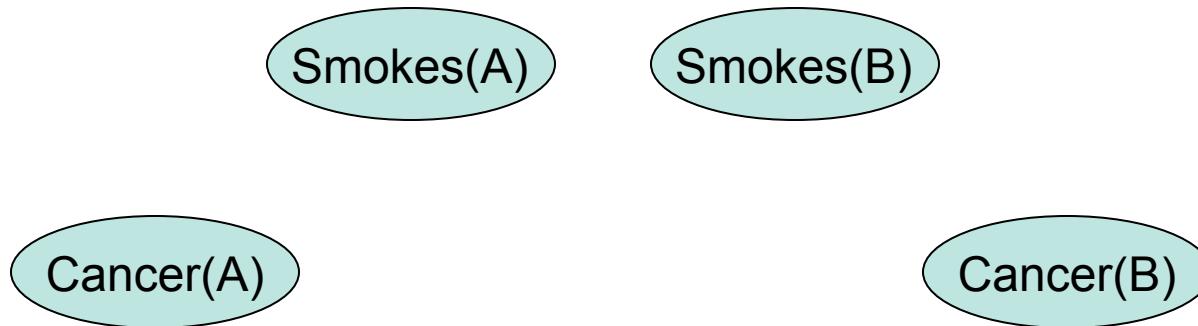
# Beispiel: Friends & Smokers

## MLN → Markov Netz: Belegungen

1.5  $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$

1.1  $\forall x, y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$

Zwei Konstanten: **Anna** (A) und **Bob** (B)



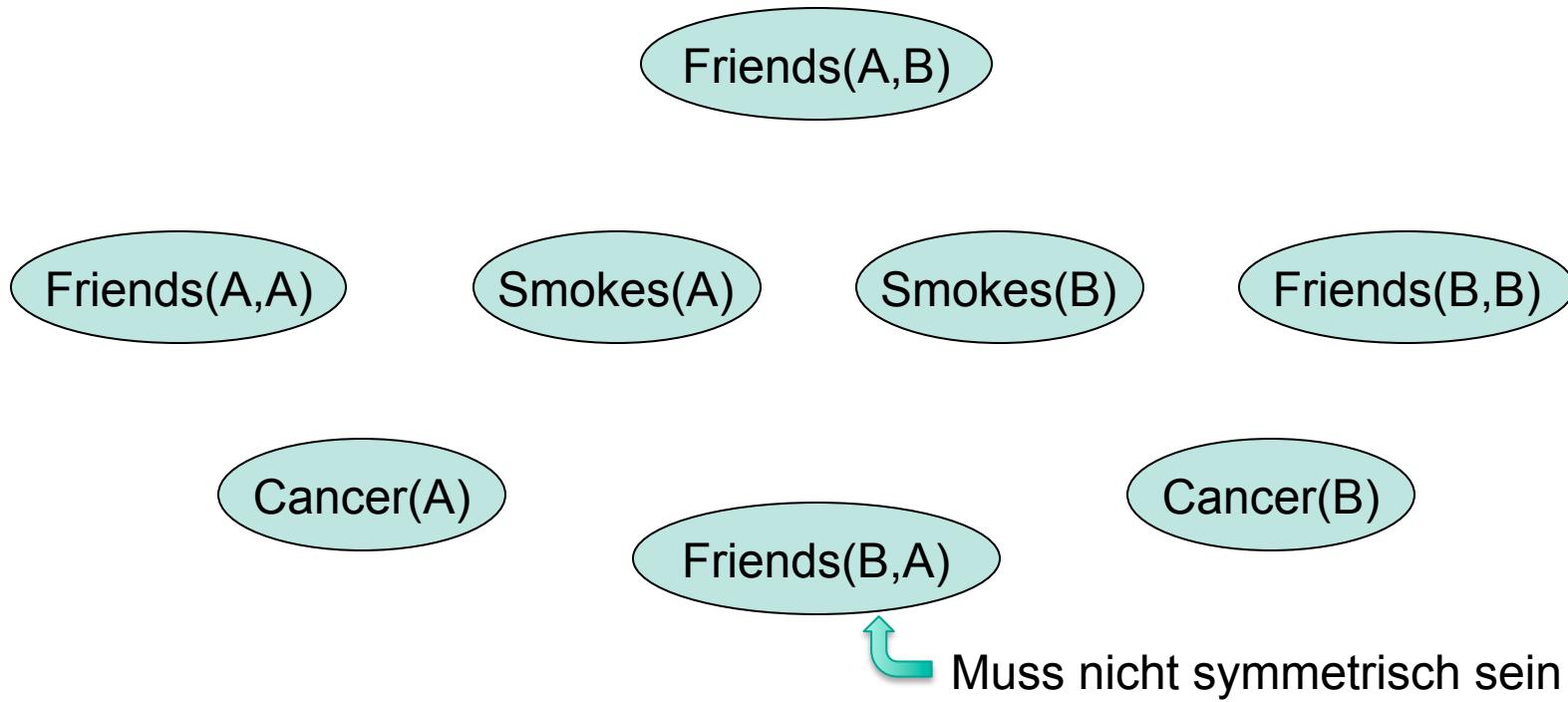
# Beispiel: Friends & Smokers

## MLN → Markov Netz: Belegungen

1.5  $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$

1.1  $\forall x, y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$

Zwei Konstanten: **Anna** (A) und **Bob** (B)



# Beispiel: Friends & Smokers

## MLN → Markov Netz: ungerichteter Graph

1.5

$$\forall x \neg Smokes(x) \vee Cancer(x)$$

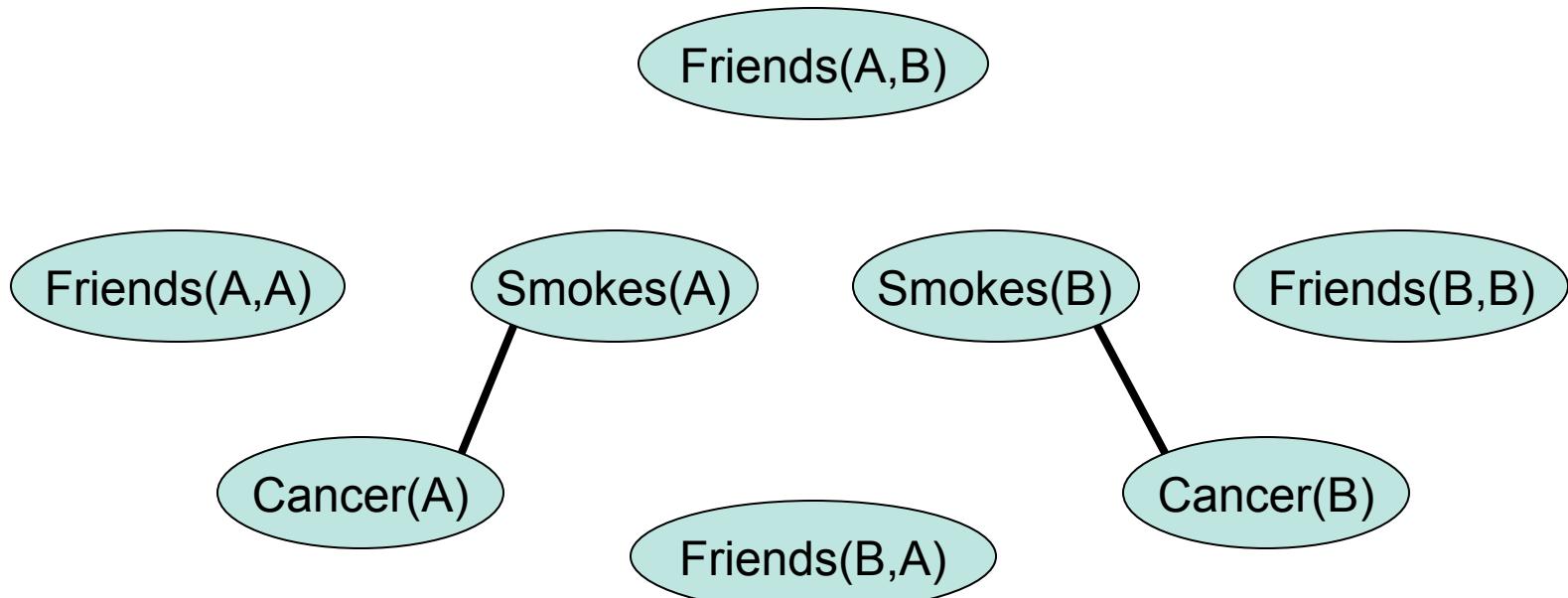
1.1

$$\forall x, y \neg Friends(x, y) \vee (\neg Smokes(x) \vee Smokes(y))$$

$$\forall x, y \neg Friends(x, y) \vee (Smokes(x) \vee \neg Smokes(y))$$

Klauseln!!

Zwei Konstanten: **Anna** (A) und **Bob** (B)



# Beispiel: Friends & Smokers

## MLN → Markov Netz: ungerichteter Graph

1.5

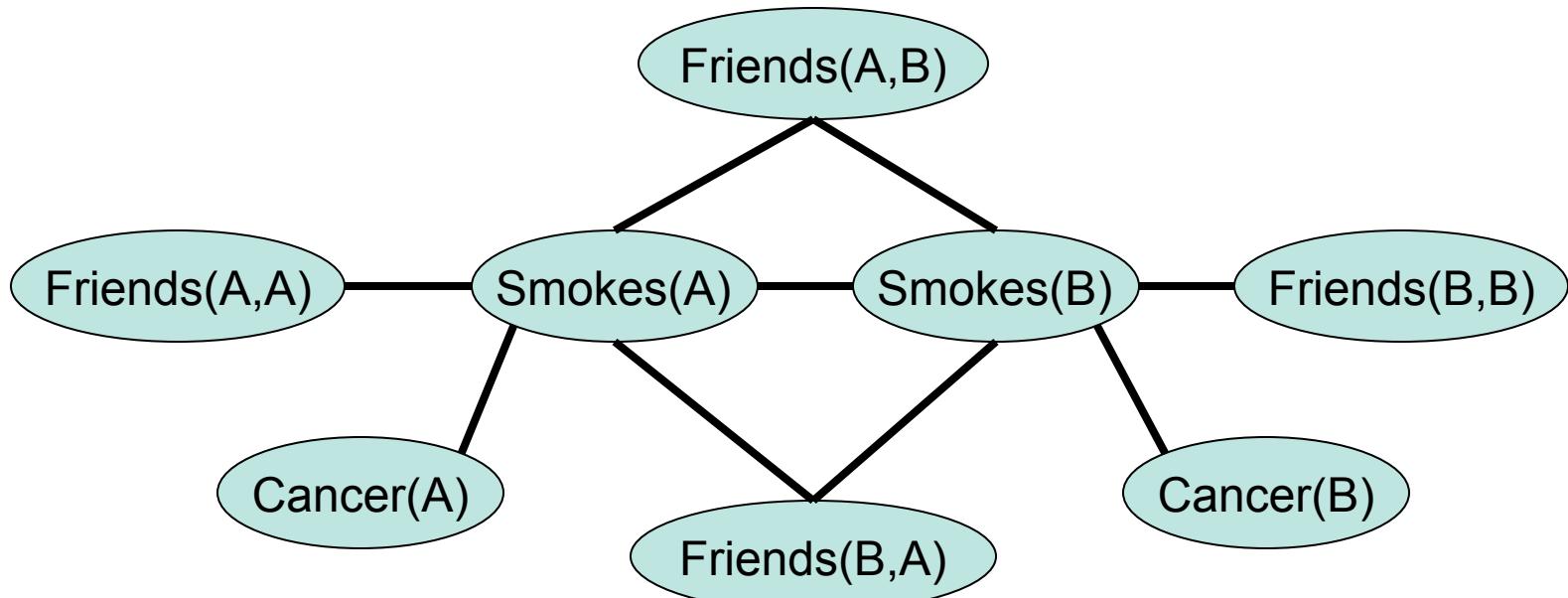
$$\forall x \neg \text{Smokes}(x) \vee \text{Cancer}(x)$$

1.1

$$\forall x, y \neg \text{Friends}(x, y) \vee (\neg \text{Smokes}(x) \vee \text{Smokes}(y))$$

$$\forall x, y \neg \text{Friends}(x, y) \vee (\text{Smokes}(x) \vee \neg \text{Smokes}(y))$$

Zwei Konstanten: **Anna** (A) und **Bob** (B)



# Beispiel: Friends & Smokers

## MLN → Markov Netz: ungerichteter Graph

1.5

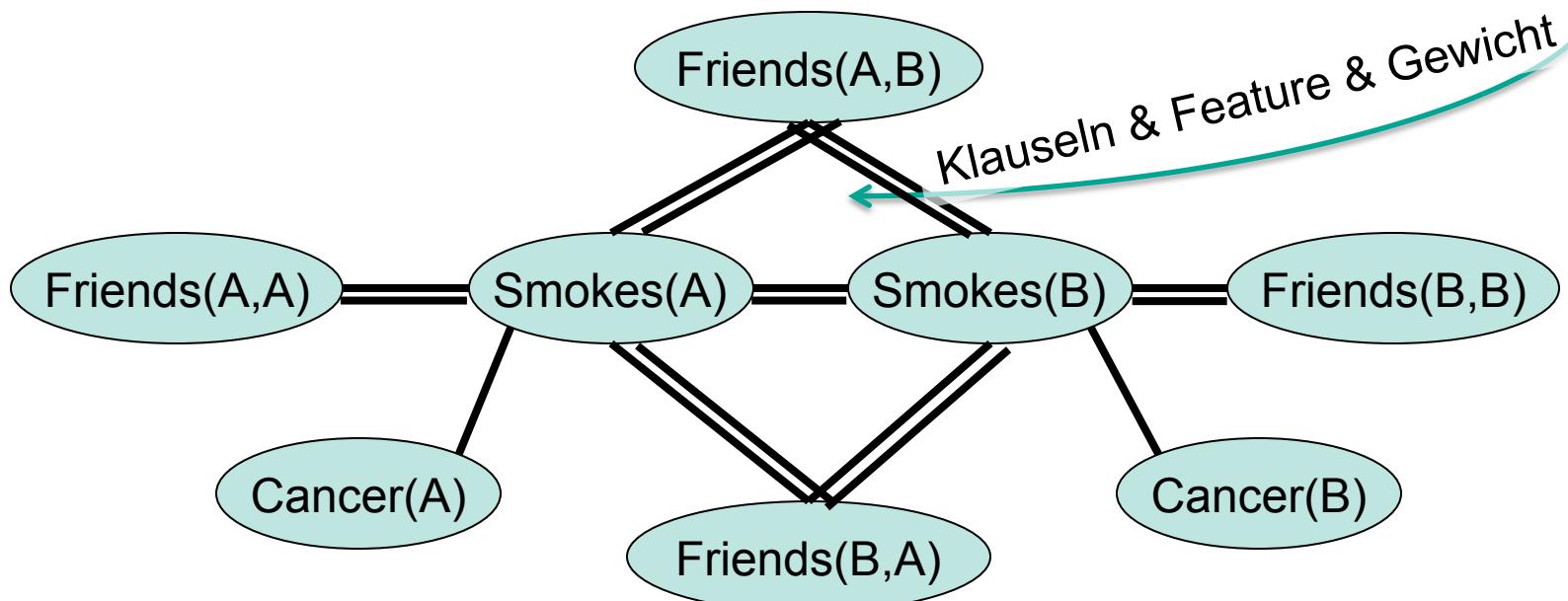
$$\forall x \neg \text{Smokes}(x) \vee \text{Cancer}(x)$$

1.1

$$\forall x, y \neg \text{Friends}(x, y) \vee (\neg \text{Smokes}(x) \vee \text{Smokes}(y))$$

$$\forall x, y \neg \text{Friends}(x, y) \vee (\text{Smokes}(x) \vee \neg \text{Smokes}(y))$$

Zwei Konstanten: **Anna** (A) und **Bob** (B)



# Inhalt

- Motivation
- Hintergrund
- Markov Logik
- Inferenz
- Lernen
- Software
- Anwendungen
- Diskussion
- Literatur

# Markov Logik Netz

- MLN ist eine **Schablone (Template)** für den Aufbau eines Markov Netzes
- Wahrscheinlichkeit einer Welt  $X$ :

$$P(X) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(X)\right)$$

Gewicht der Formel  $i$

Anzahl der wahren Belegungen  
der Formel (Klausel)  $i$  in  $X$

- Typisierte Variablen und Konstanten reduzieren sehr stark die Größe des aufgebauten Markov Netzes (reduziert Belegung)
- Möglichkeit der Nutzung von:
  - Funktionen, Quantoren, etc.
  - Endliche und kontinuierliche Domänen

# Beispiel: Friends & Smokers

## MLN → Markov Netz: Verbundwahrscheinlichkeit

1.5

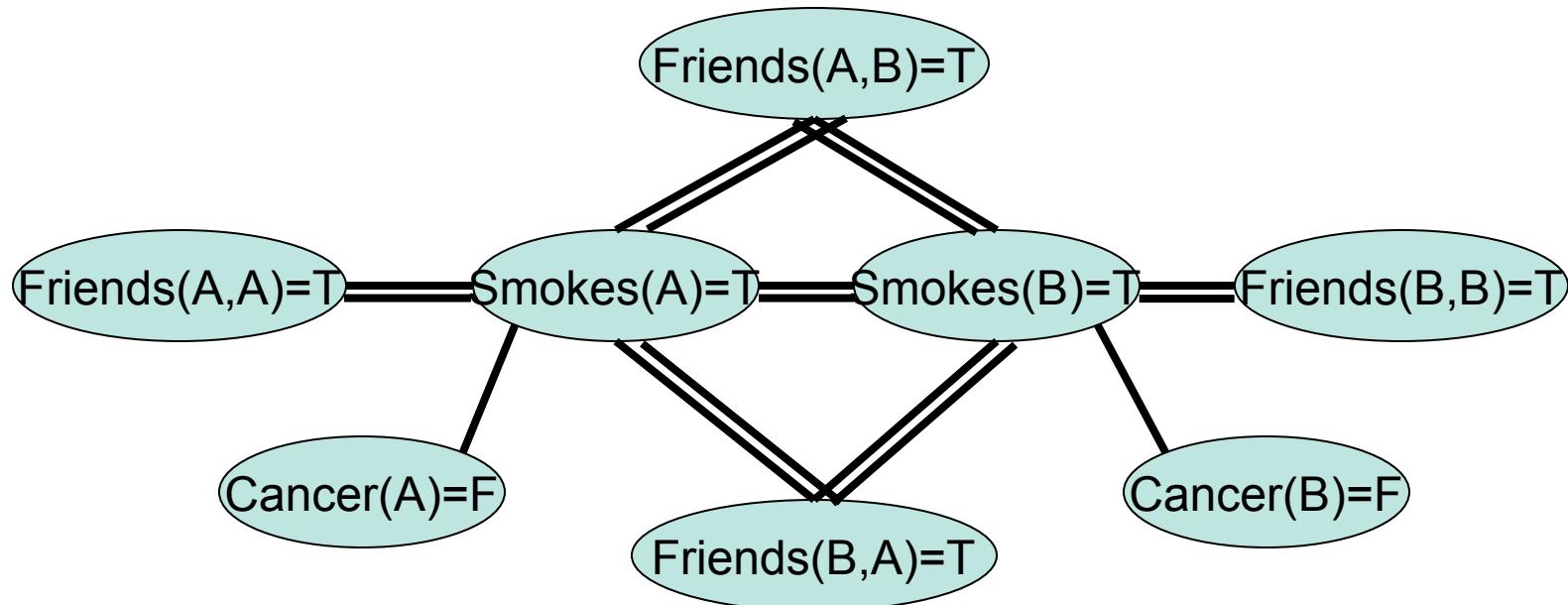
$$\forall x \neg Smokes(x) \vee Cancer(x)$$

1.1

$$\forall x, y \neg Friends(x, y) \vee (\neg Smokes(x) \vee Smokes(y))$$

$$\forall x, y \neg Friends(x, y) \vee (Smokes(x) \vee \neg Smokes(y))$$

Zwei Konstanten: **Anna** (A) und **Bob** (B)



$$P(S(A) = T, S(B) = T, F(A, B) = T, \dots, C(A) = F, C(B) = F) = \frac{1}{Z} \exp(4 * 1.1 + 4 * 1.1 + 0)$$

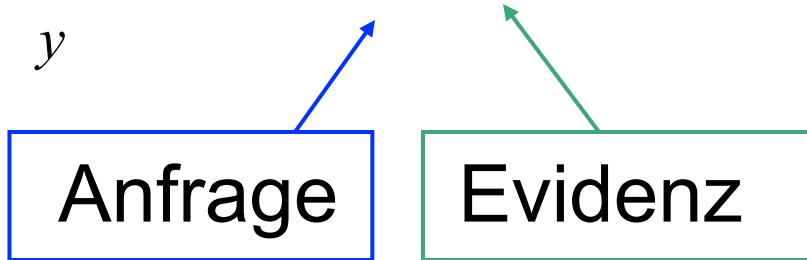
# Vergleich zu Prädikatenlogik erster Ordnung

- Wenn unendliche Gewichte  
→ MLN = reine Prädikatenlogik erster Ordnung (beweisbar)
  
- Wenn erfüllbare Wissensbasis, d.h. Welten s.d. alle Formeln wahr sind  
→ erfüllende Interpretationen (Welten) sind die Modalwerte der Verteilung  
→ d.h. dass auch im Falle von verrauschten Daten die Welten, die durch die PL ausdrückt werden enthalten sind
  
- Vorteil: Markov Logik erlaubt Widersprüche unter den Formeln  
→ wichtig für reale verrauschte Daten oder unvollständiges Modell

- **Problem:** Finden des wahrscheinlichsten Zustands der Welt (Variablen  $y$ ), gegeben Evidenzen  $x$  (wahrscheinlichste Erklärung für  $y$ )
- Bekannter Ansatz: Maximum a-posteriori Ansatz bzw. most probable estimate

$$\arg \max_y P(y | x)$$

Anfrage      Evidenz



- **Problem:** Finden des wahrscheinlichsten Zustands der Welt (Variablen  $y$ ), gegeben Evidenzen  $x$  (wahrscheinlichste Erklärung für  $y$ )
- Entsprechend MLN ersetzen:

$$\arg \max_y \frac{1}{Z} \exp \left( \sum_i w_i n_i(x, y) \right)$$

exp – monotone steigende Funkt.

$$\arg \max_y \sum_i w_i n_i(x, y)$$



- **Problem:** Finden des wahrscheinlichsten Zustands der Welt (Variablen  $y$ ), gegeben Evidenzen  $x$  (wahrscheinlichste Erklärung für  $y$ )

$$\arg \max_y \sum_i w_i n_i(x, y)$$

- Dies ist bekannt als „weighted MaxSAT“ Problem
  - SAT = satisfied belief / satisfiability - solver → alle Formeln erfüllen
  - MaxSAT → so viele Formeln wie möglich erfüllen
  - Weighted MaxSat → max. gewichtete Anzahl von Formel erfüllen
- Vorteile:
  - Standard - Verfahren → SAT solver (z.B., MaxWalkSAT [Kautz et al., 1997] )
  - Potentiell schneller als rein logische Inferenz (!)

# WalkSAT Algorithmus

- Ziel:  
Finden einer Interpretation (Wahrheitswert – Zuordnung, truth assignments) der Variablen um alle Klauseln (gleichzeitig) zu erfüllen
- Klauseln = Disjunktion von Literalen (Verallgemeinerung: jede Formeln in disjunktive Form konvertierbar)

# WalkSAT Algorithmus (Exkurs)

```
for  $i \leftarrow 1$  to  $\text{max-tries}$  do
    solution = random truth assignment
    for  $j \leftarrow 1$  to  $\text{max-flips}$  do
        if all clauses satisfied then
            return solution
         $c \leftarrow$  random unsatisfied clause
        with probability  $p$ 
            flip a random variable in  $c$ 
        else
            flip variable in  $c$  that maximizes
            number of satisfied clauses
    return failure
```

# Weighted MaxWalkSAT Algorithmus (Exkurs)

```
for  $i \leftarrow 1$  to  $max\text{-tries}$  do
    solution = random truth assignment
    for  $j \leftarrow 1$  to  $max\text{-flips}$  do
        if  $\sum$  weights(sat. clauses) > threshold then
            return solution
         $c \leftarrow$  random unsatisfied clause
        with probability  $p$ 
            flip a random variable in  $c$ 
        else
            flip variable in  $c$  that maximizes
             $\sum$  weights(sat. clauses)
    return failure, best solution found
```

# Aber ... Speicherkomplexität

## ■ Problem:

- Wenn es  $n$  Konstanten gibt (z.B. jede Person oder jedes Wort einer Sprache)
- und Klausel mit (max.) Anzahl  $c$  unterschiedlichen Variablen
- Dann benötigt das belegte Netz  $O(n^c)$  Speicher → kann schnell, sehr stark explodieren
- z.B.: 1000 Konstanten (nicht viel z.B. für Personen) und  $c=3$  (nicht viel) → 1 Mrd. belegter Klauseln(!)
- Z.B.  $6 \cdot 10^9$  Personen auf der Welt daher potentiell  $6^2 \cdot 10^{18}$  belegte Klauseln (z.B. Freunde) aber nur wenige sind relevant

## ■ Praktikable Lösung:

- Nutzung der „dünnen“ Belegung (sparseness);  
→ Klauseln werden nur dünn belegt (ground clauses lazily)
- Exkurs: LazySAT Algorithmus [Singla & Domingos, 2006]

- Häufige Fragestellung: Anfragen bzgl. Erfüllung einer Formel (gegeben MLN und Konstanten C)

$$P(\text{Formel} \mid \text{MLN}, C) = ?$$

- entspricht „einfach“ der Summe der Wahrscheinlichkeiten der Welten in denen die Formel erfüllt ist
- Aber: große (exponentielle) Anzahl von Welten
- Lösung:
  - MCMC (markov chain monte carlo) Ansatz:
    - stochastisches Abtasten der Welten (Wiederholtes sampeln von Belegungen für eine/jede Variable in abh. von MAP)
    - Test ob Formel erfüllt ist
  - Anteil (Faktor) entspricht der gesuchten Wahrscheinlichkeit  $P$

```
state ← random truth assignment
for  $i \leftarrow 1$  to num-samples do
    for each variable  $x$ 
        sample  $x$  according to  $P(x|neighbors(x))$ 
        state ← state with new value of  $x$ 
    P( $F$ ) ← fraction of states in which  $F$  is true
```

# Aber ... MCMC oft nicht ausreichend (Exkurs)

## ■ Problem:

- Für rein deterministische Abhängigkeiten „funktioniert“ MCMC nicht weil der Zustandsraum nicht zusammenhängend ist und MCMC nur in einem Teilraum des Zustandsraumes auswertet (auch nahezu deterministische Abhängigkeiten machen Inferenz sehr langsam)

## ■ Lösung:

- Kombination von MCMC mit WalkSAT
  - Wähle abhängig vom WalkSAT Strategie nächste Abtastung → effiziente Sprünge im Zustandsraum möglich
  - MC-SAT Algorithmus [Poon & Domingos, 2006]

- Weitere Anfrage: Erfüllung einer Formel gegeben andere Formel (gegeben MLN und Konstanten C)

$$P(\textcolor{red}{Formel}_1 \mid \textcolor{blue}{Formel}_2, \textit{MLN}, C) = ?$$

- Idee: In MCMC Berücksichtigen der Welten, die die bedingende Formel<sub>2</sub> erfüllen ← Aufwendig bei großen Netzen
- Alternative → Aufbau nur des relevanten, belegten Netzes
- Bsp.: Frage ob Smoking(Anna) erfüllt ist → nur Personen berücksichtigen (und entsprechende Formeln) die Freunde von Anna sind, Rest der Welt ist egal

- Weitere Anfrage: Erfüllung einer Formel gegeben andere Formel (gegeben MLN und Konstanten C)

$$P(\textcolor{red}{Formel}_1 \mid \textcolor{blue}{Formel}_2, \textit{MLN}, C) = ?$$

- Idee: In MCMC Berücksichtigen der Welten die bedingende Formel erfüllen ← Aufwendig bei großen Netzen
- Alternative → Aufbau nur des relevanten, belegten Netzes
  - Konstruiere das minimale notwendige Netz aus Atomen der Anfrage ( $Formel_1$ ) bis die Evidenzen (geg. auch durch Atome der  $Formel_2$  erreicht werden) ← Markov blanket
  - Verallgemeinerung des KBMC – knowledge based model construction
  - Dann Anwenden der MCMC – Inferenz
- Erweiterung s.g. lifted inference [Singla & Domingos, 2008]

```
network ← Ø
queue ← query nodes
repeat
    node ← front(queue)
    remove node from queue
    add node to network
    if node not in evidence then
        add neighbors(node) to queue
until queue = Ø
```

# Inhalt

- Motivation
- Hintergrund
- Markov Logik
- Inferenz
- **Lernen**
- Software
- Anwendungen
- Diskussion
- Literatur

## ■ Gegeben

- Daten (beobachtete Welten)  
z.B. in einer relationalen Form (Datenbank)
- Geschlossene Welt - Annahme, d.h. jedes Atom, das nicht in den Daten /Datenbank ist, ist *falsch*  
(sonst erweiterte Verfahren wie EM)

## ■ Lernen

- Parameter (Gewichte):
  - Formeln sind gegeben, Gewichte sind unbekannt
  - Ansätze
    - Generativ Lernen
    - Diskriminatives Lernen
- Struktur
  - Formeln und Gewichte sind unbekannt (oder nur teilweise vorhanden)

# Generatives Lernen von Gewichten

- Finden der Gewichte die am wahrscheinlichsten die Daten generiert haben
- Maximum likelihood  $\rightarrow \log(P)$  maximieren

$$\frac{\partial}{\partial w_i} \log P_w(x) = n_i(x) - E_w[n_i(x)] = n_i(x) - \sum_{x'} P_w(X = x') n_i(x')$$

Anz. der wahren Belegungen  
der Klausel  $i$  in den Daten

Erwartete Anz. der wahren Belegungen  
entsprechend dem inferierten Modell

- Std. Maximierungsalgorithmus – z.B: Gradientenverfahren
- Konvex, d.h. keine lokalen Maxima, Initialisierung beliebig
- Aber: Benötig Inferenz um die Erwartung zu bestimmen – in jedem Schritt des Gradientenverfahrens  $\rightarrow$  langsam !

# Pseudo-Likelihood (Exkurs)

- Üblicher Trick: Wenn das zu optimierende Kriterium zu schwierig ist – Finden eines einfacheren Kriteriums, z.B.:

$$PL(x) \equiv \prod_i P(x_i \mid neighbors(x_i))$$

- Benötigt keine vollständige Inferenz in jedem Schritt
- Konsistent
- Oft verwendet  $\leftarrow$  Standard Verfahren
- Aber PL ggf nicht gut geeignet für lange Inferenz-Ketten (wg. lokaler Sicht – siehe MCMC)

- Maximieren der bedingten Wahrscheinlichkeit (conditional likelihood) der Anfrage ( $y$ ) gegeben Evidenzen ( $x$ ) – wenn vorab bekannt welches Evidenzen sind (üblicherweise der Fall)

$$\frac{\partial}{\partial w_i} \log P_w(y | x) = n_i(x, y) - E_w[n_i(x, y)]$$

Anz. der wahren Belegungen  
der Klausel  $i$  in den Daten

Erwartete Anz. der wahren Belegungen  
entsprechend dem inferierten Modell

- Vorteil – wenn Evidenzen bekannt fallen viele Inferenzschritte weg
- ➔ Weitere Optimierung (Exkurs): nur bezüglich der MAP Werte von  $y$  optimieren: „Voted Perceptron Algorithmus“

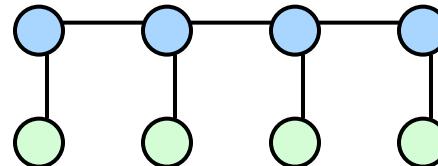
# Voted Perceptron (Exkurs)

- Ursprünglich für das diskriminative Trainieren von HMMs [Collins, 2002]
- Annahme, dass das Netz ein lineare Kette ist

```

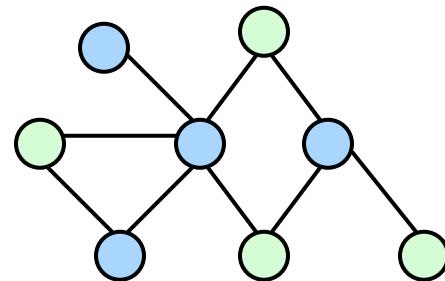
 $w_i \leftarrow 0$ 
for  $t \leftarrow 1$  to  $T$  do
     $y_{MAP} \leftarrow \text{Viterbi}(x)$ 
     $w_i \leftarrow w_i + \eta [\text{count}_i(y_{Data}) - \text{count}_i(y_{MAP})]$ 
return  $\sum_t w_i / T$ 

```



- HMMs sind ein Spezialfall der MLNs
- Ersetzte Viterbi durch MaxWalkSAT
- Netz kann ein beliebiger Graph sein

```
wi ← 0
for t ← 1 to T do
    yMAP ← MaxWalkSAT(x)
    wi ← wi + η [counti(yData) – counti(yMAP)]
return ∑t wi / T
```



## Typischer (heuristischer) Algorithmus

- **Initialisierung:** Atomare Klauseln oder vorgegebene Wissensbasis
- **Iterieren**
  - **Ändern durch Operatoren:** Add/Löschen v. Literalen, Negieren, ...
  - **Lernen der Gewichte des MLN**
  - **Evaluationsfunktion für MLN durch:**
    - Güte des Modells für Daten → Verbundwahrscheinlichkeit
    - Strukturbewertung (prior): Bestrafung wenn Struktur zu sehr von einer def. Vorgabe abweicht (zur Reduzierung von Overfitting)
- **Suche nach neuen Kandidaten (Änderung)**
  - **Verschiedene Methoden (Exkurs):**
    - Beam [Kok & Domingos, 2005]
    - Shortest-first [Kok & Domingos, 2005]
    - Bottom-up [Mihalkova & Mooney, 2007]

# Inhalt

- Motivation
- Hintergrund
- Markov Logik
- Inferenz
- Lernen
- Software → Alchemy [alchemy.cs.washington.edu](http://alchemy.cs.washington.edu)
- Anwendungen → Tuffy <http://i.stanford.edu/hazy/hazy/tuffy/>  
→ ...
- Diskussion
- Literatur

# Anwendung: Informationsgewinnung

- Typisches Beispiel: Zitierungen analysieren  
Strukturierte Information in unstrukturierter Form (Text) mit Rauschen
- Datensätze sind z.B.:

Parag Singla and Pedro Domingos, “Memory-Efficient Inference in Relational Domains” (AAAI-06).

Singla, P., & Domingos, P. (2006). Memory-efficient inference in relational domains. In Proceedings of the Twenty-First National Conference on Artificial Intelligence (pp. 500-505). Boston, MA: AAAI Press.

H. Poon & P. Domingos, Sound and Efficient Inference with Probabilistic and Deterministic Dependencies”, in Proc. AAAI-06, Boston, MA, 2006.

P. Hoifung (2006). Efficient inference. In Proceedings of the Twenty-First National Conference on Artificial Intelligence.

# Problemstellung: Segmentierung

Parag Singla and Pedro Domingos, “Memory-Efficient Inference in Relational Domains” (AAAI-06).

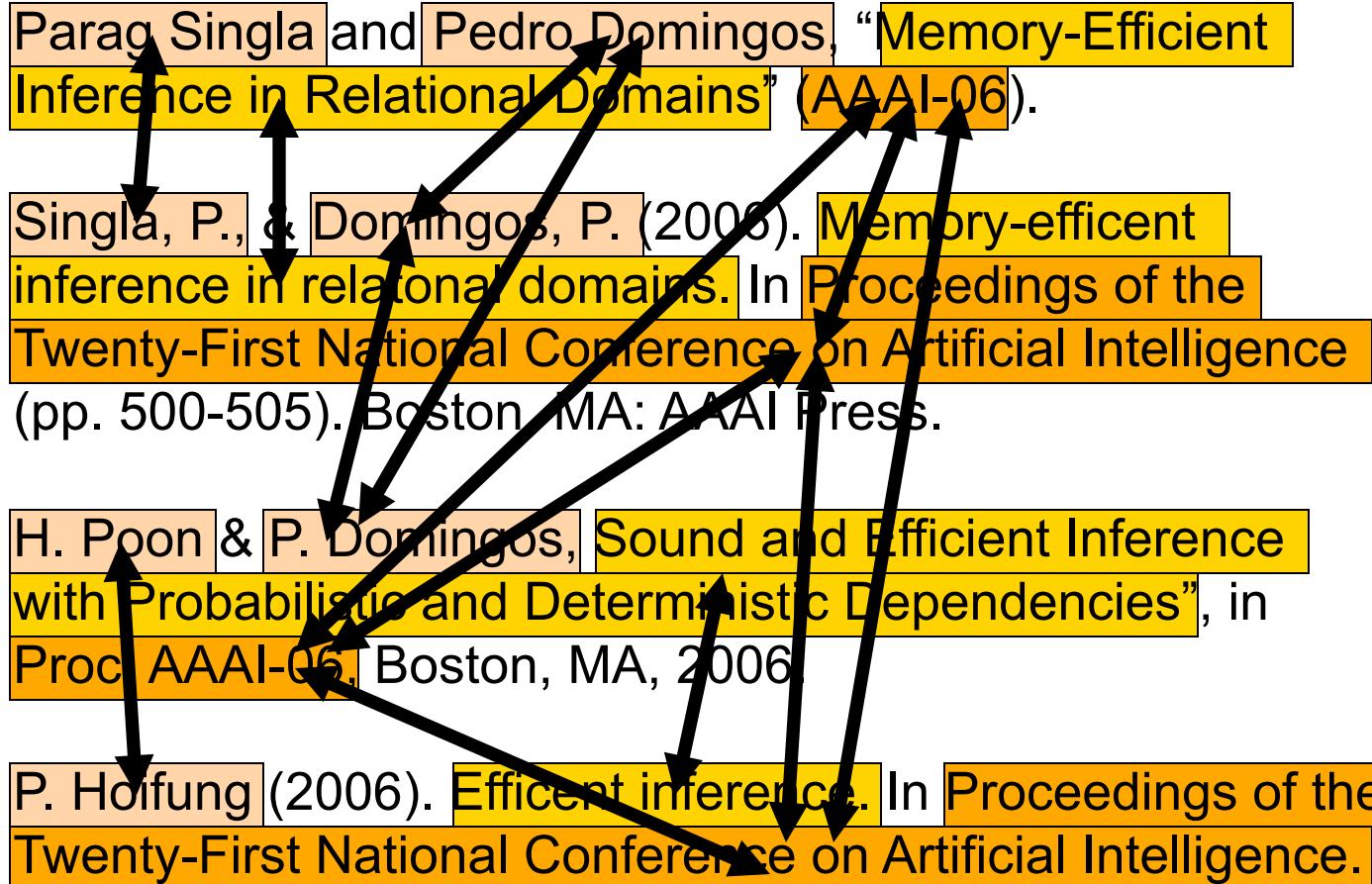
Singla, P., & Domingos, P. (2006). Memory-efficient inference in relational domains. In Proceedings of the Twenty-First National Conference on Artificial Intelligence (pp. 500-505). Boston, MA: AAAI Press.

H. Poon & P. Domingos, Sound and Efficient Inference with Probabilistic and Deterministic Dependencies”, in Proc. AAAI-06, Boston, MA, 2006.

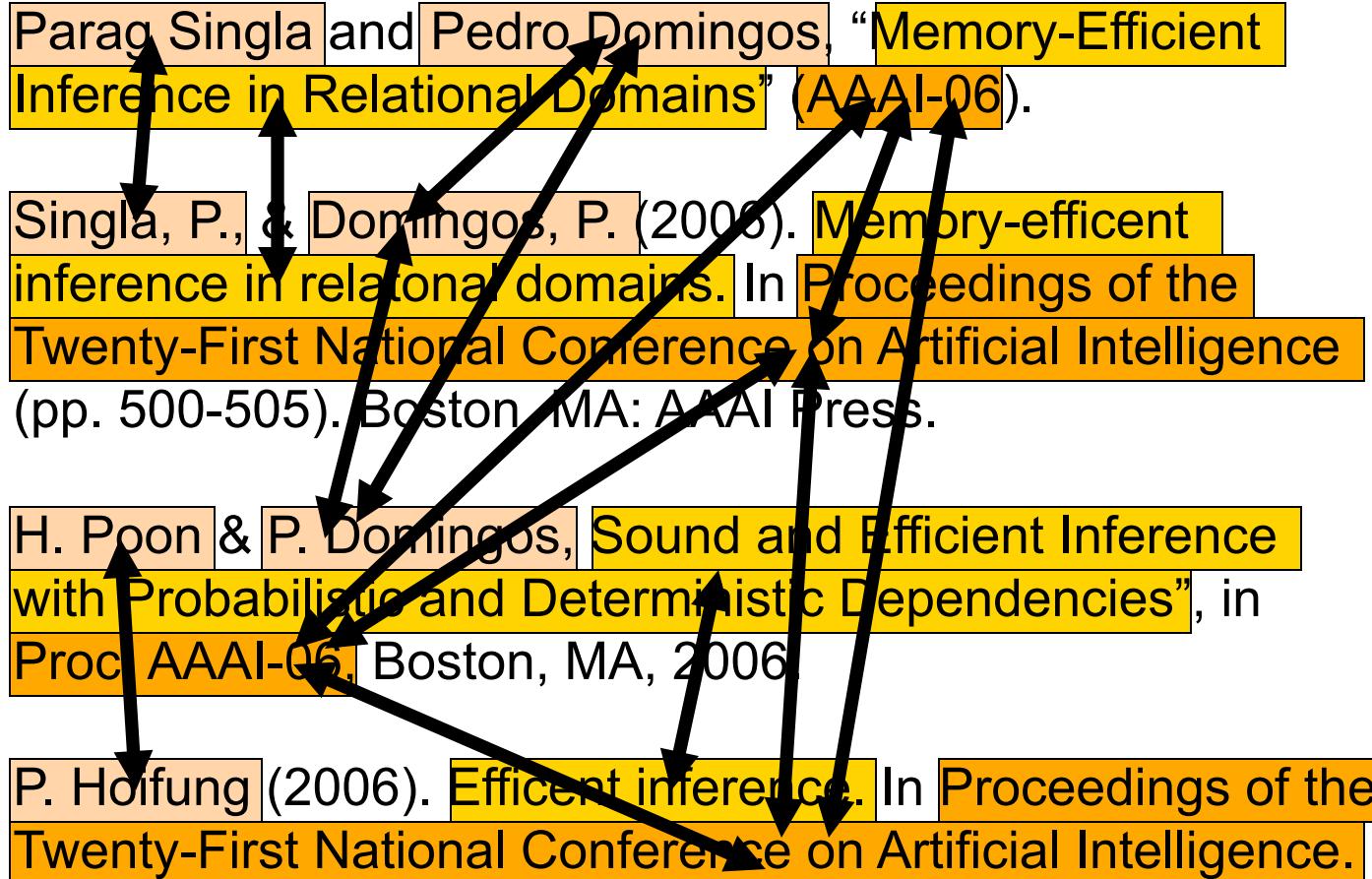
P. Hoifung (2006). Efficient inference. In Proceedings of the Twenty-First National Conference on Artificial Intelligence.

- Author
- Title
- Venue

# Problemstellung: Entitäten auflösen / vergleichen



# Problemstellung: Entitäten / Datensätze auflösen



# Typische Ansätze vs. MLN

## ■ Ein Ansatz (Idee)

### ■ Segmentierung:

- HMM (etc.) um Zeichen einem entsprechende Feld zuzuordnen

### ■ Entitäten auflösen / vergleichen

- Gleiche Felder /Datensätze zu erkennen (Logistische Regression)
- Transitiver Schluss

Im Vergleich dazu :

## ■ MLN (Alchemy) Implementierung: Sieben Formeln ☺

# Typen und Prädikate

```
token = {Parag, Singla, and, Pedro, ...}  
field = {Author, Title, Venue, ...}  
citation = {C1, C2, ...}  
position = {0, 1, 2, ...}
```

Konstanten  
typisiert

```
Token(token, position, citation)  
InField(position, field, citation)  
SameField(field, citation, citation)  
SameCit(citation, citation)
```

# Typen und Prädikate

```
token = {Parag, Singla, and, Pedro, ...}  
field = {Author, Title, Venue}  
citation = {C1, C2, ...}  
position = {0, 1, 2, ...}
```

**Token(token, position, citation)** ← Evidenz

```
InField(position, field, citation)  
SameField(field, citation, citation)  
SameCit(citation, citation)
```

- 
- „token“ ist an einer „Position“ in der „Zitierung“

# Typen und Prädikate

```

token = {Parag, Singla, and, Pedro, ...}
field = {Author, Title, Venue}
citation = {C1, C2, ...}
position = {0, 1, 2, ...}
  
```

Token(token, position, citation)

InField(position, field, citation)  
 SameField(field, citation, citation)  
 SameCit(citation, citation)

Anfragen  
 ← Segmentierung  
 ← Entitäten /  
 Vergleich

- 
- „Feld“ ist an einer „Position“ in der „Zitierung“
  - Felder in den Zitierungen gleich
  - Zitierungen sind gleich

```
Token(+t,i,c) => InField(i,+f,c)
InField(i,+f,c) <=> InField(i+1,+f,c)
f != f' => (!InField(i,+f,c) v !InField(i,+f',c))
```

```
Token(+t,i,c) ^ InField(i,+f,c) ^ Token(+t,i',c')
               ^ InField(i',+f,c') => SameField(+f,c,c')
SameField(+f,c,c') <=> SameCit(c,c')
SameField(f,c,c') ^ SameField(f,c',c")
                  => SameField(f,c,c")
SameCit(c,c') ^ SameCit(c',c") => SameCit(c,c")
```

- 
- Syntax bei Alchemy „+...“ bedeutet das dies für alle Variationen gilt

# Formeln - Segmentierung

**Token(+t,i,c) => InField(i,+f,c)**

**InField(i,+f,c) <=> InField(i+1,+f,c)**

**f != f' => (!InField(i,+f,c) v !InField(i,+f',c))**

**Token(+t,i,c) ^ InField(i,+f,c) ^ Token(+t,i',c')  
^ InField(i',+f,c') => SameField(+f,c,c')**

**SameField(+f,c,c') <=> SameCit(c,c')**

**SameField(f,c,c') ^ SameField(f,c',c'')  
=> SameField(f,c,c'')**

**SameCit(c,c') ^ SameCit(c',c'') => SameCit(c,c'')**

- 
- Wenn „token“ ist an einer „Position“ in der „Zitierung“ dann ist „feld“ an der Position

# Formeln - Segmentierung

**Token(+t,i,c) => InField(i,+f,c)**

**InField(i,+f,c) <=> InField(i+1,+f,c)**

**f != f' => (!InField(i,+f,c) v !InField(i,+f',c))**

**Token(+t,i,c) ^ InField(i,+f,c) ^ Token(+t,i',c')  
^ InField(i',+f,c') => SameField(+f,c,c')**

**SameField(+f,c,c') <=> SameCit(c,c')**

**SameField(f,c,c') ^ SameField(f,c',c'')  
=> SameField(f,c,c'')**

**SameCit(c,c') ^ SameCit(c',c'') => SameCit(c,c'')**

- Gdw. „feld“ ist an einer „Position“ in der „Zitierung“ ist „feld“ auch der folgende Position

# Formeln – Segmentierung, Erweiterung um Interpunktions

**Token(+t,i,c) => InField(i,+f,c)**

**InField(i,+f,c) ^ !Token(".",i,c) <=> InField(i+1,+f,c)**

**f != f' => (!InField(i,+f,c) v !InField(i,+f',c))**

**Token(+t,i,c) ^ InField(i,+f,c) ^ Token(+t,i',c')**

**^ InField(i',+f,c') => SameField(+f,c,c')**

**SameField(+f,c,c') <=> SameCit(c,c')**

**SameField(f,c,c') ^ SameField(f,c',c")**

**=> SameField(f,c,c")**

**SameCit(c,c') ^ SameCit(c',c") => SameCit(c,c")**

# Formeln - Segmentierung

**Token(+t, i, c) => InField(i, +f, c)**

**InField(i, +f, c) <=> InField(i+1, +f, c)**

**f != f' => (!InField(i, +f, c) v !InField(i, +f', c))**

**Token(+t, i, c) ^ InField(i, +f, c) ^ Token(+t, i', c')  
^ InField(i', +f, c') => SameField(+f, c, c')**

**SameField(+f, c, c') <=> SameCit(c, c')**

**SameField(f, c, c') ^ SameField(f, c', c'')  
=> SameField(f, c, c'')**

**SameCit(c, c') ^ SameCit(c', c'') => SameCit(c, c'')**

- Wenn „felder“ nicht gleich sind dann sind sie nicht an der gleichen Position

# Formeln – Entitäten Vergleich

**Token(+t,i,c) => InField(i,+f,c)**  
**InField(i,+f,c) <=> InField(i+1,+f,c)**  
**f != f' => (!InField(i,+f,c) v !InField(i,+f',c))**

**Token(+t,i,c) ^ InField(i,+f,c) ^ Token(+t,i',c')  
^ InField(i',+f,c') => SameField(+f,c,c')**

**SameField(+f,c,c') <=> SameCit(c,c')**

**SameField(f,c,c') ^ SameField(f,c',c'')  
=> SameField(f,c,c'')**

**SameCit(c,c') ^ SameCit(c',c'') => SameCit(c,c'')**

# Formeln – Entitäten Vergleich

$$\begin{aligned}
 \text{Token}(+t, i, c) &\Rightarrow \text{InField}(i, +f, c) \\
 \text{InField}(i, +f, c) &\Leftrightarrow \text{InField}(i+1, +f, c) \\
 f \neq f' &\Rightarrow (\neg \text{InField}(i, +f, c) \vee \neg \text{InField}(i, +f', c))
 \end{aligned}$$

$$\begin{aligned}
 \text{Token}(+t, i, c) \wedge \text{InField}(i, +f, c) \wedge \text{Token}(+t, i', c') \\
 \wedge \text{InField}(i', +f, c') &\Rightarrow \text{SameField}(+f, c, c') \\
 \text{SameField}(+f, c, c') &\Leftrightarrow \text{SameCit}(c, c') \\
 \text{SameField}(f, c, c') \wedge \text{SameField}(f, c', c'') \\
 &\Rightarrow \text{SameField}(f, c, c'') \\
 \text{SameCit}(c, c') \wedge \text{SameCit}(c', c'') &\Rightarrow \text{SameCit}(c, c'')
 \end{aligned}$$

- 
- Äquivalenz bringt zusätzliches Wissen denn gdw. 2 Zitierungen gleich sind sind auch die Felder gleich

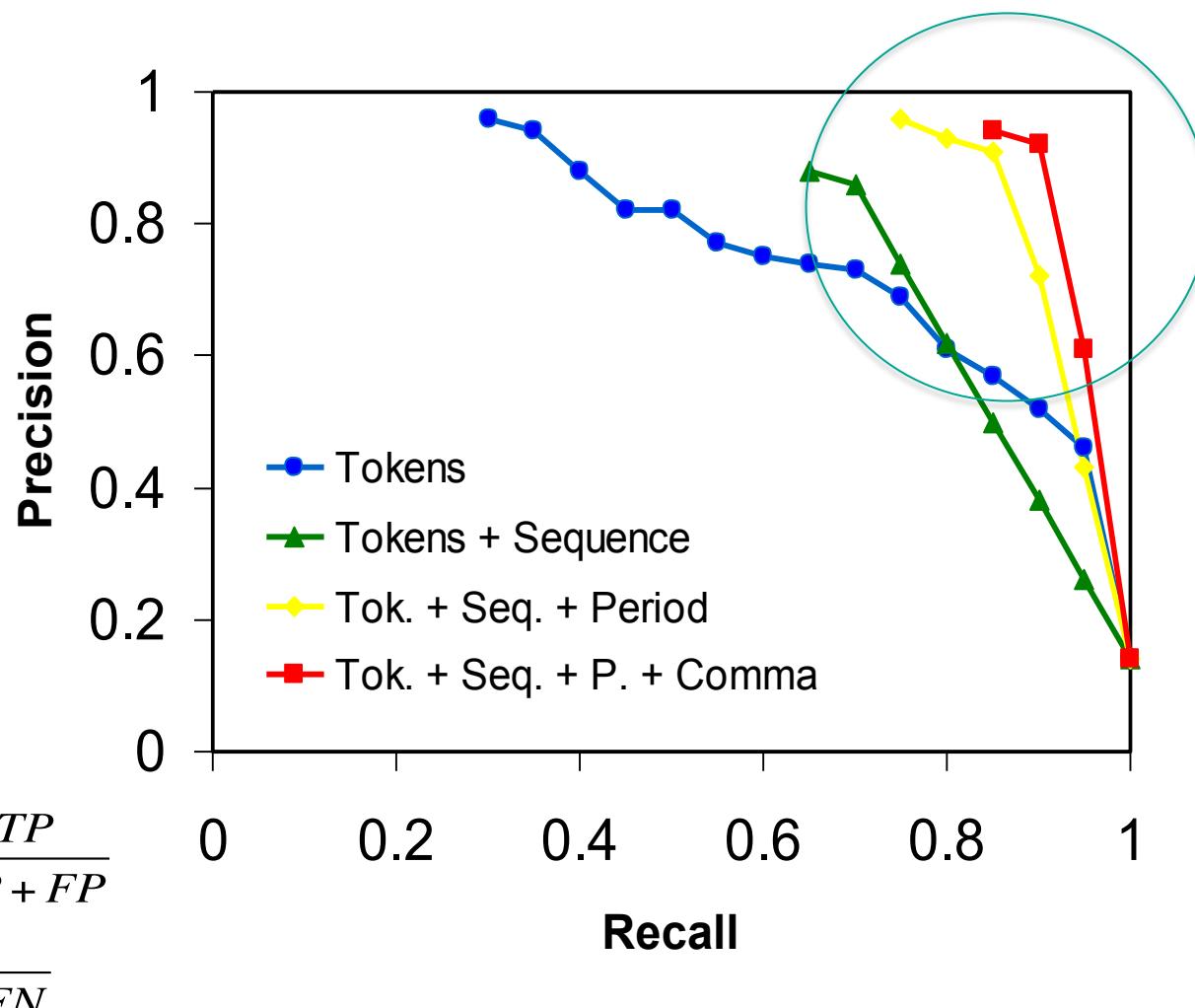
# Formeln - Transitivität

$$\begin{aligned} \text{Token}(+t, i, c) &\Rightarrow \text{InField}(i, +f, c) \\ \text{InField}(i, +f, c) &\Leftrightarrow \text{InField}(i+1, +f, c) \\ f \neq f' &\Rightarrow (\neg \text{InField}(i, +f, c) \vee \neg \text{InField}(i, +f', c)) \end{aligned}$$

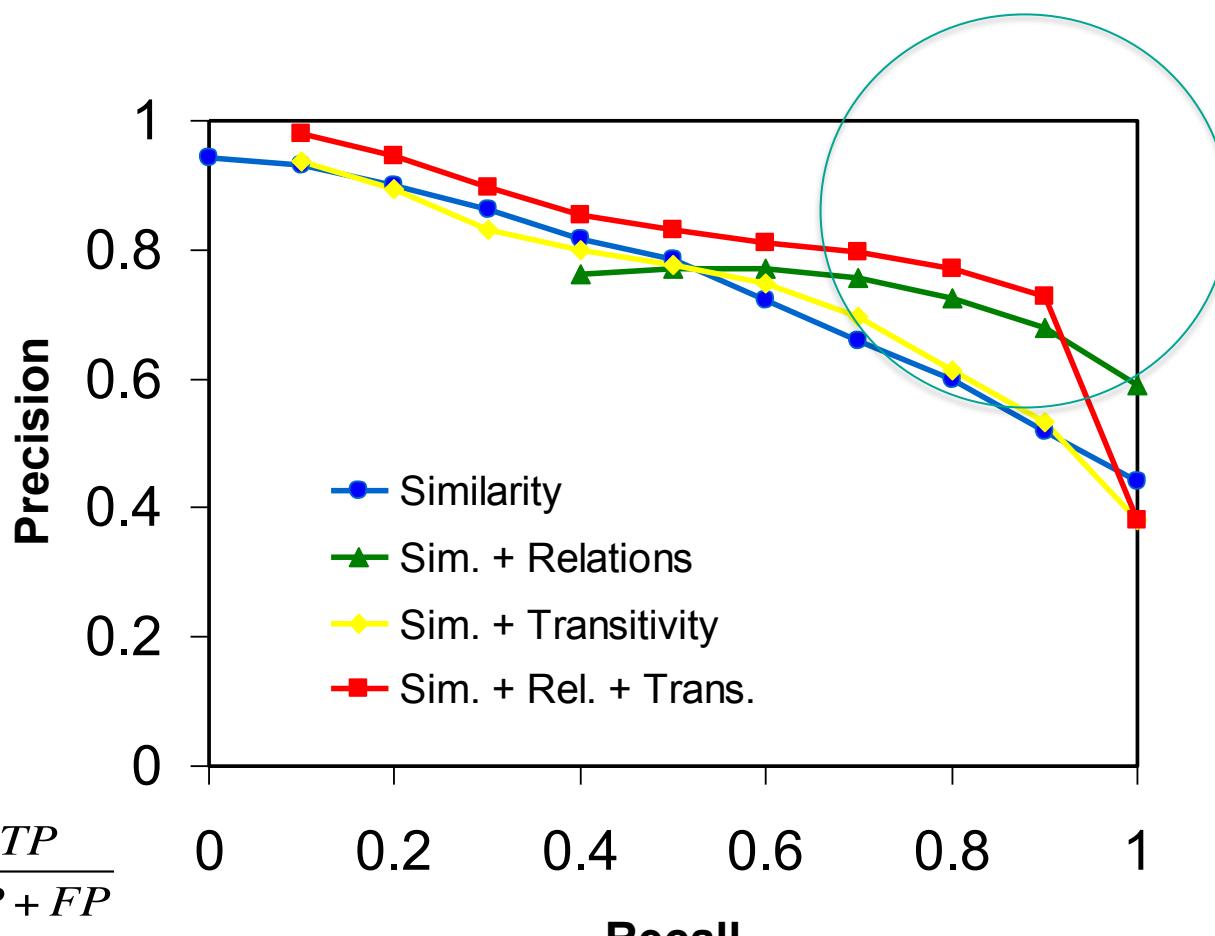
$$\begin{aligned} \text{Token}(+t, i, c) \wedge \text{InField}(i, +f, c) \wedge \text{Token}(+t, i', c') \\ \wedge \text{InField}(i', +f, c') &\Rightarrow \text{SameField}(+f, c, c') \\ \text{SameField}(+f, c, c') &\Leftrightarrow \text{SameCit}(c, c') \\ \text{SameField}(f, c, c') \wedge \text{SameField}(f, c', c'') \\ &\Rightarrow \text{SameField}(f, c, c'') \\ \text{SameCit}(c, c') \wedge \text{SameCit}(c', c'') &\Rightarrow \text{SameCit}(c, c'') \end{aligned}$$

## ■ Transitivität

# Ergebnisse: Segmentierung (Felder zuordnen) auf Cora Datensatz



# Ergebnisse: Matching der Zitierungen



$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

## Gegeben

Markierung von Baustellen

- Gelbe Fahrspurmarkierung
- Warnbaken
- Hinweisschild „Baustelle“

Erkennung mittels

- Bildbasierter Klassifikatoren z.B.: Viola Jones oder
- Segmentierung und merkmalsbasierte SVM – Klassifikation ....



## Gesucht

Baustellenerkennung mit MLN

- Markierungsobjekte als Implikatoren
- Gewichte repräsentieren Sicherheit der Information
- Lernen der Gewichte

# MLNs zur Erkennung von Baustellen

```
// Types
time = {"now", "just", "recently"}
color = {"white", "yellow"}
amount = {"none", "some", "many", "lots"}

// Evidence
Beacons(scene, time, amount)
RoadWorkSign(scene, time, amount)
Marking(scene, color, time, amount)

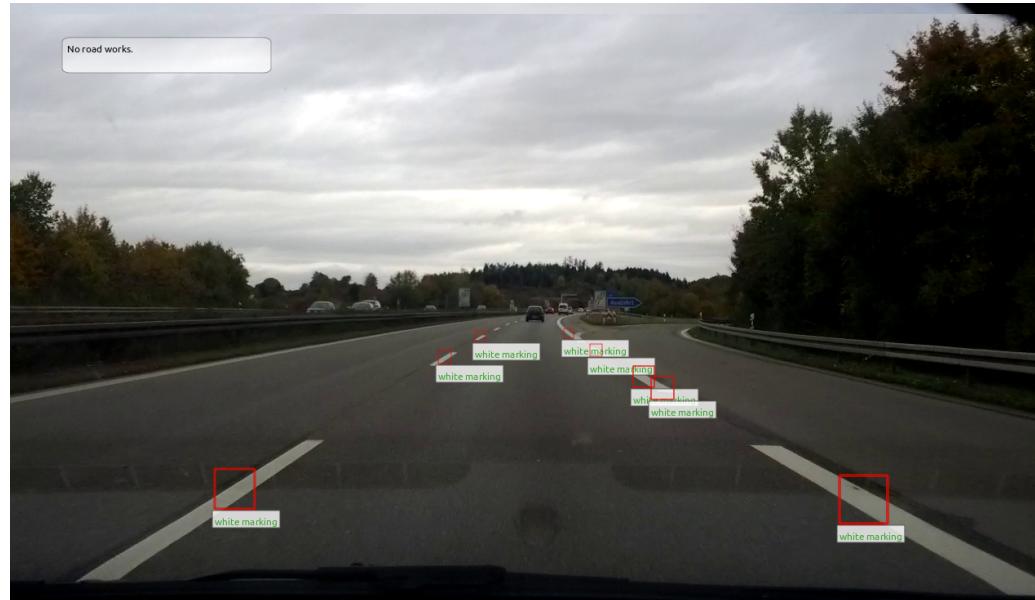
// Anfragen
RoadWorks (x)

// Rule skeletons
Beacons (x, +t, +a) => RoadWorks (x)
RoadWorkSign (x, +t, +a) => RoadWorks (x)
Marking (x, +c, +t, +a) => RoadWorks (x)
```



```
Beacons (133, "now", "many")
Beacons (133, "just", "many")
Beacons (133, "recently", "many")
RoadWorkSign (133, "now", "some")
RoadWorkSign (133, "just", "some")
RoadWorkSign (133, "recently", "some")
Marking (133, "white", "now", "none")
Marking (133, "white", "just", "some")
Marking (133, "white", "recently", "some")
Marking (133, "yellow", "now", "many")
Marking (133, "yellow", "just", "many")
Marking (133, "yellow", "recently", "lots")
```

# Evaluation Baustellenerkennung



## Erkennung

Baustellen	2
Einzelbilder	3838
Erkennungsrate	97.2 %
Laufzeit	7 fps

# Relevanzbestimmung für Ampeln

Gegeben

- Erkennung v. Ampeln und Eigenschaften (Erkennung mittels bildbasierter Segmentierung / Klassifikation / Zustands-, Farbschätzung / ...)
- Ideales Kreuzungsmodell (mit Ampeln)
- Wahrscheinlichste Route

Gesucht: Wie passen die Beobachtungen zum Modell?

- Welche gesehene Ampel entspricht welcher Ampel im Modell
- Wie ist der Zustand für die wahrscheinlichste Route

Herausforderungen:

- Umgang mit eingeschränktem Sichtfeld
- Behandlung von Erkennungsfehlern



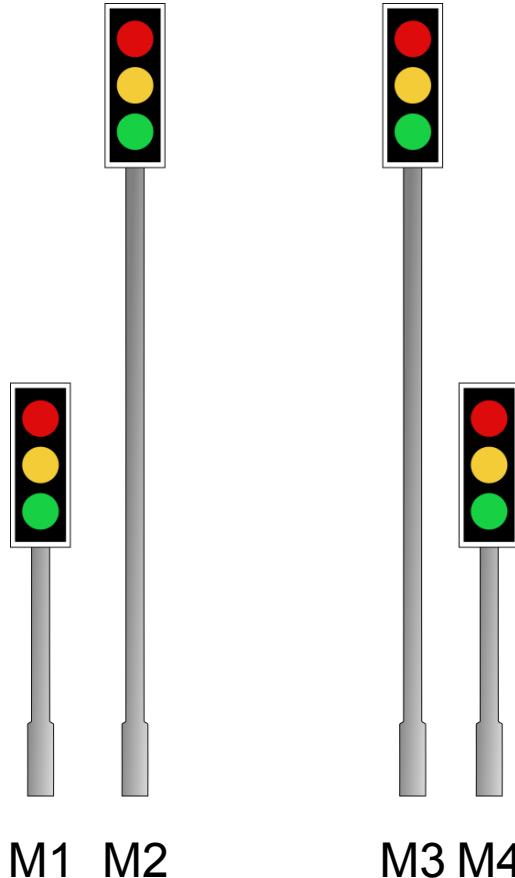
Relevance Estimation of Traffic Elements Using Markov Logic Networks  
 Nienhüser, Gumpp, Zöllner, IEEE Intelligent Transportation Systems, 2011

# Umfelderkennung: Sichtbare Ampeln



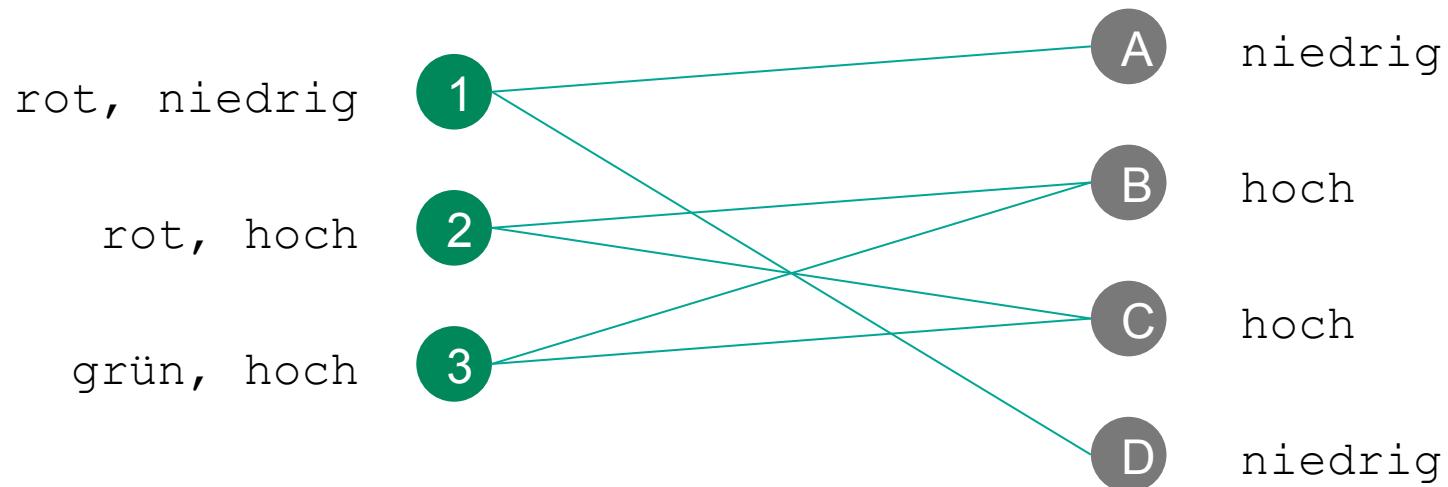
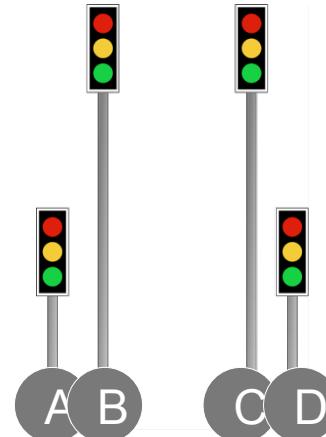
			
<b>Objekt</b>	L1	L2	L3
<b>Farbe</b>	rot	rot	grün
<b>Höhe</b>	niedrig	hoch	hoch
<b>Prädikate</b>	hasColor(L1, red) hasHeight(L1, low)	hasColor(L2, red) hasHeight(L2, high) rightOf(L2, L1)	hasColor(L3, green) hasHeight(L3, high) rightOf(L3, L2)

# Kreuzungsmodell: Wissen über reales Kreuzungslayout

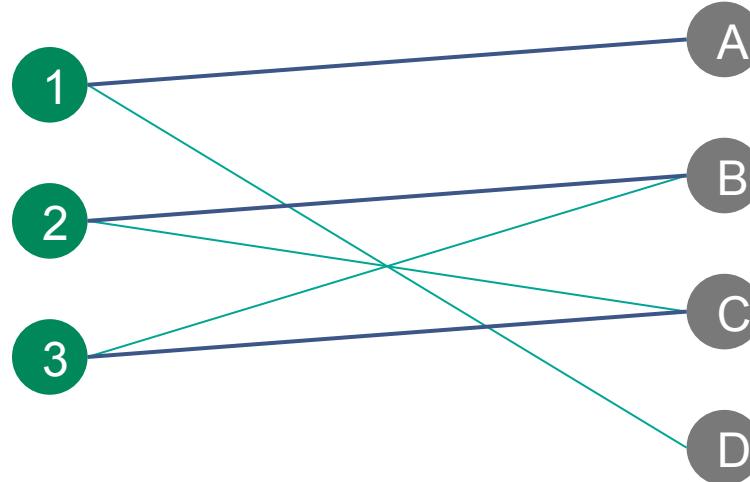


Gruppe	1	1	2	2
Objekt	M1	M2	M3	M4
Höhe	niedrig	hoch	hoch	niedrig
Prädikate	hasHeight (M1, low) sameGroup (M1, M2) leftOf (M2, M1)	...	...	...

# Relevanzbestimmung für Ampeln: Matching



# Relevanzbestimmung für Ampeln



- Domänenwissen schränkt Matching ein: Regeln im MLN
  - Nur planare Zuordnung (keine kreuzenden Kanten)
  - Keine Mehrfachzuordnungen
  - Ampeln unterschiedlicher Farbe können nicht der gleichen Synchronisationsgruppe zugeordnet werden
- Anfrage: Relevanzbestimmung per Inferenz **Matches ( $x, y$ )**

# Evaluation Relevanzbestimmung



Ampelkreuzungen	12
Einzelbilder	7840
Bestimmung relevanter Ampeln	98.8 % (7749)

# Markov Logik Netz (MLN)

Wissensdatenbank



← Lernen



Prob. Weltmodell  
→ Inferenz



*z.B. Umfelderkennung*

Markov Logik Netz

- Prädikatenlogische Formeln
- Gewichte repräsentieren Unsicherheit

Markov Netz

- Ungerichteter Graph
- Kompakte Repräsentation einer Wahrscheinlichkeitsverteilung
- Effiziente approximative Inferenzverfahren

Lernen sehr gut möglich  
Gute Leistungen (Inferenz auf gelerntem MLN)

Forschungsbedarf:

- Skalierung (Laufzeit sehr hoch)
- Robustheit
- Design / Struktur

# LITERATURHINWEISE

# Literatur

- L. Getoor, and B. Taskar, editors: „Introduction to Statistical Relational Learning.“ MIT Press, 2007.
- Pedro Domingos  
<http://homes.cs.washington.edu/~pedrod/>
  - Software → Alchemy
  - Veröffentlichungen (Journals, Papers)
  - Buchkapitel z.B.: Markov Logic: A Unifying Framework for Statistical Relational Learning in **Introduction to Statistical Relational Learning**, 2007
  - Folien
- Pedro Domingos - videolectures  
[http://videolectures.net/pedro\\_domingos/](http://videolectures.net/pedro_domingos/)  
z.B. Keynote ACM '08
- Youtube - Pedro Domingos - Unifying Logical and Statistical AI  
<http://www.youtube.com/watch?v=bW5DzNZgGxY>

