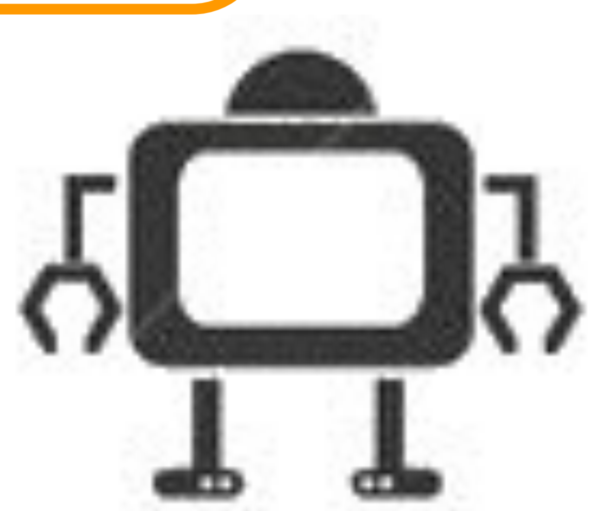
Probabilistic Logic Programming and its Applications

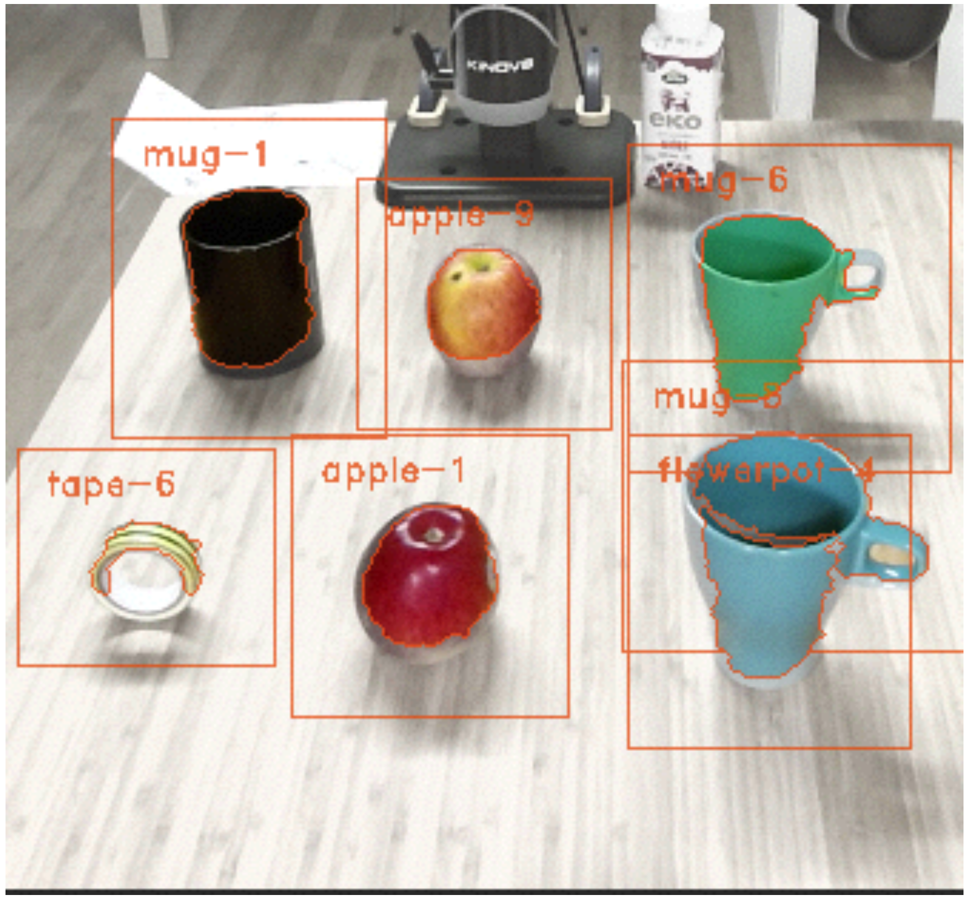
Luc De Raedt

with many slides from Angelika Kimmig

Motivation

Pick the red apple
next to the black mug
and cut it into slices





Among other things, this requires

1. Identifying which objects “apple” and “mug” refer to
2. Understanding the relation “next to”
3. Knowing that it is possible to cut an apple and how to cut it

ReGround Project’s Hypotheses

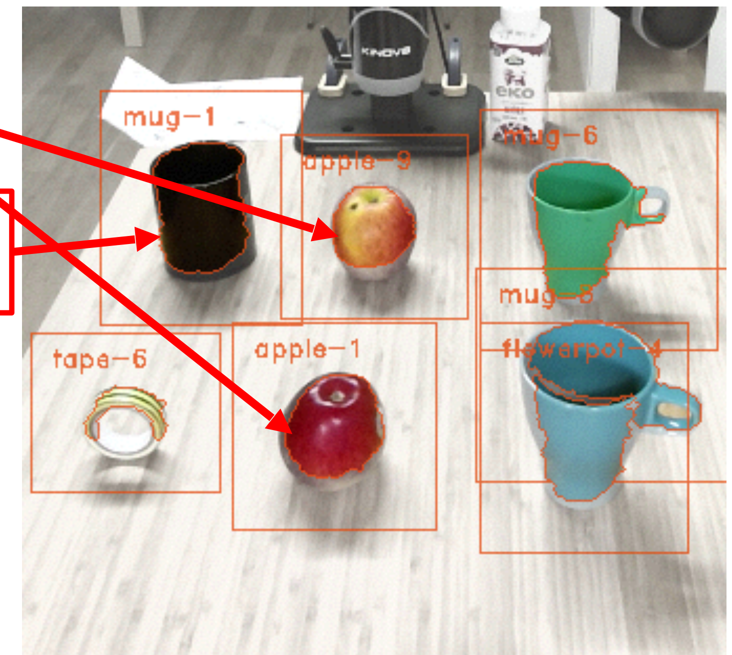
Spatial relation

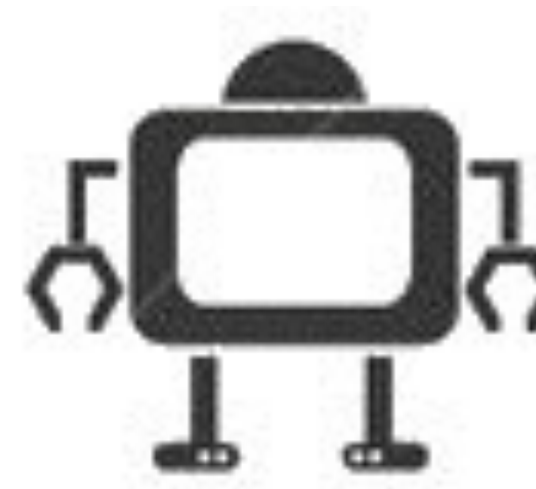
Apple affords cutting

Language

1. Pick the red apple next to the black mug
2. Cut it into slices

Vision





Four tenets of ReGround

1. Grounding requires integrating from multiple modalities
2. Identifying symbols is only the first step of grounding
3. Learning affordances is the second, underexplored step of grounding
4. Exploiting relationships and affordances will improve grounding

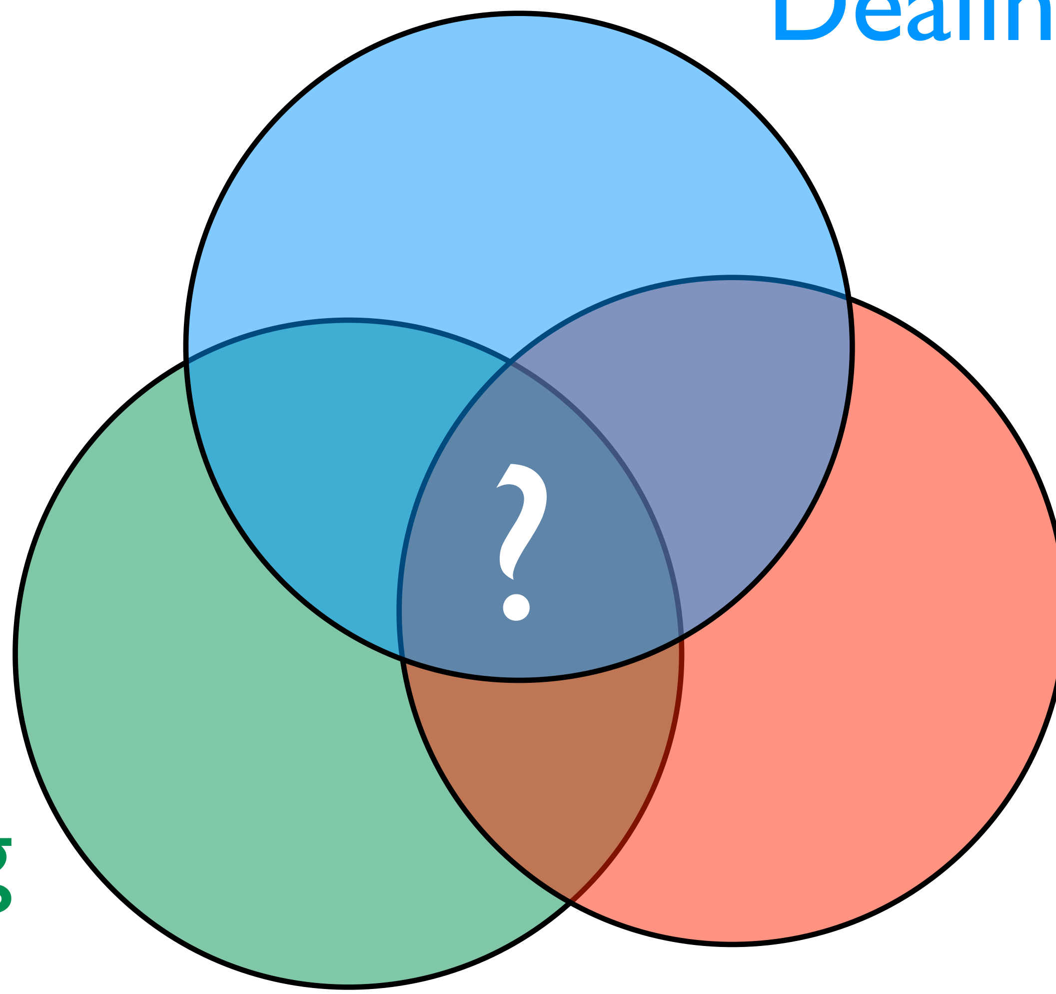
A key question in AI:

Dealing with uncertainty

- probability theory
- graphical models
- ...

Reasoning with relational data

- logic
- databases
- programming
- ...

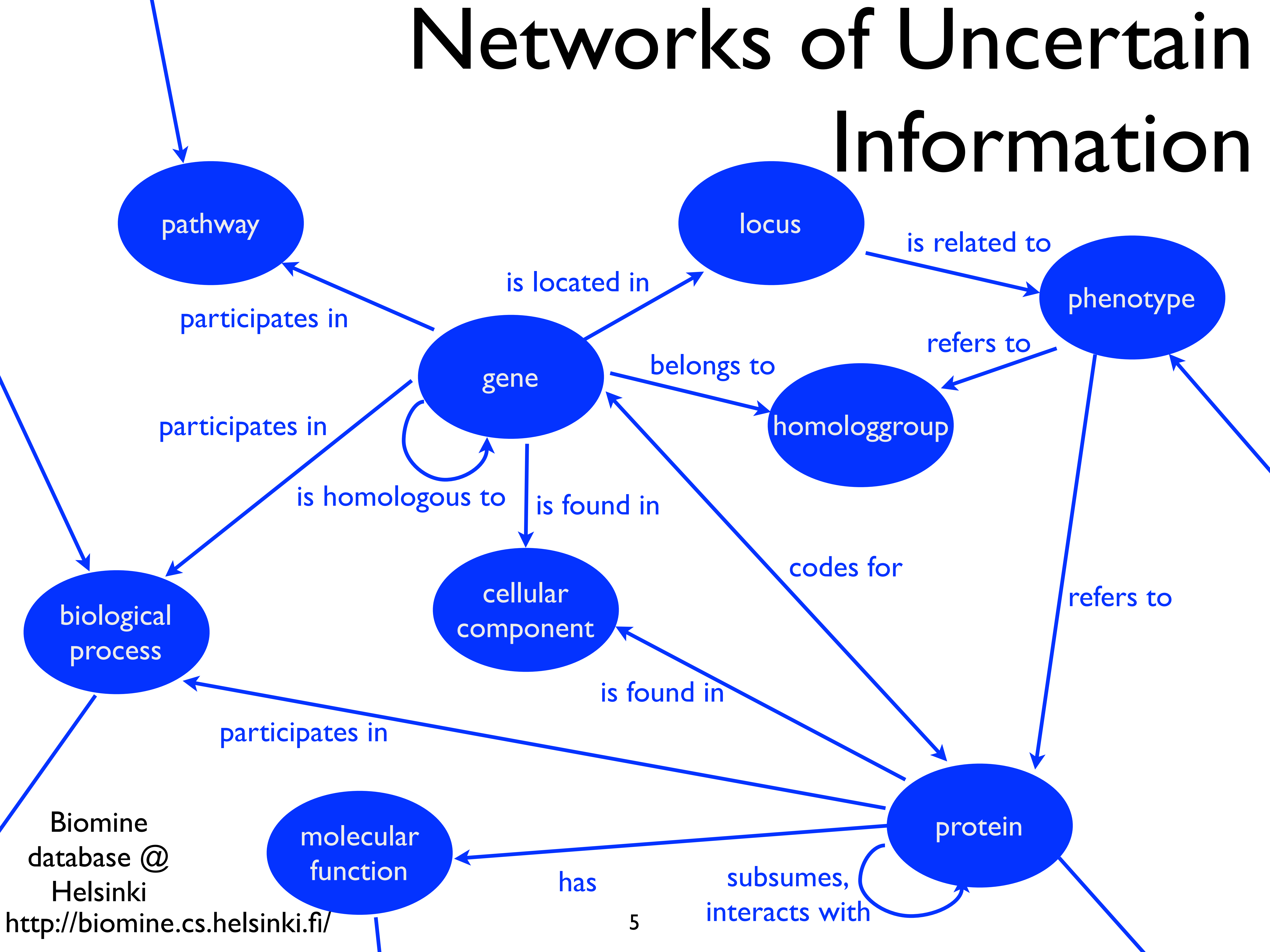


Learning

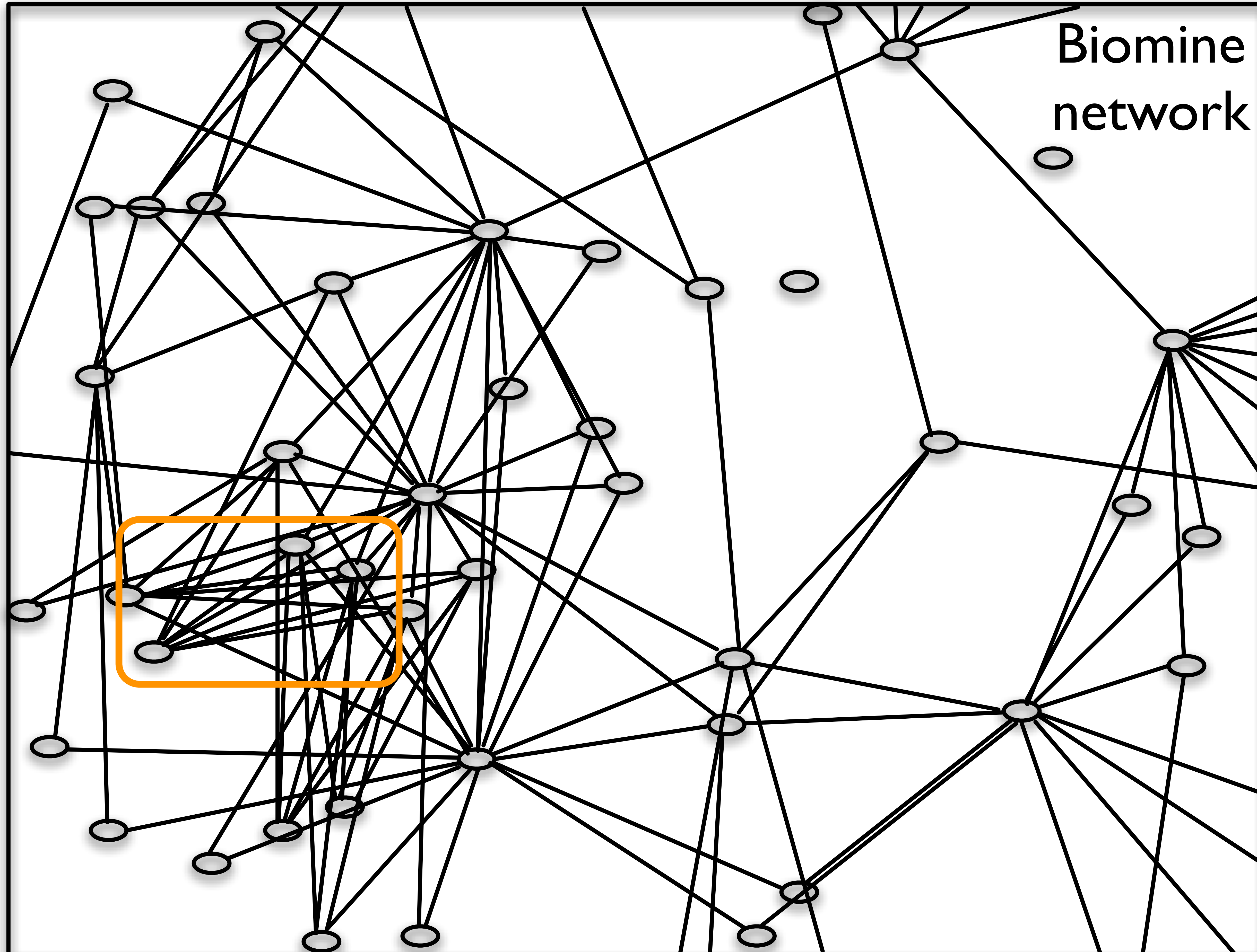
- parameters
- structure

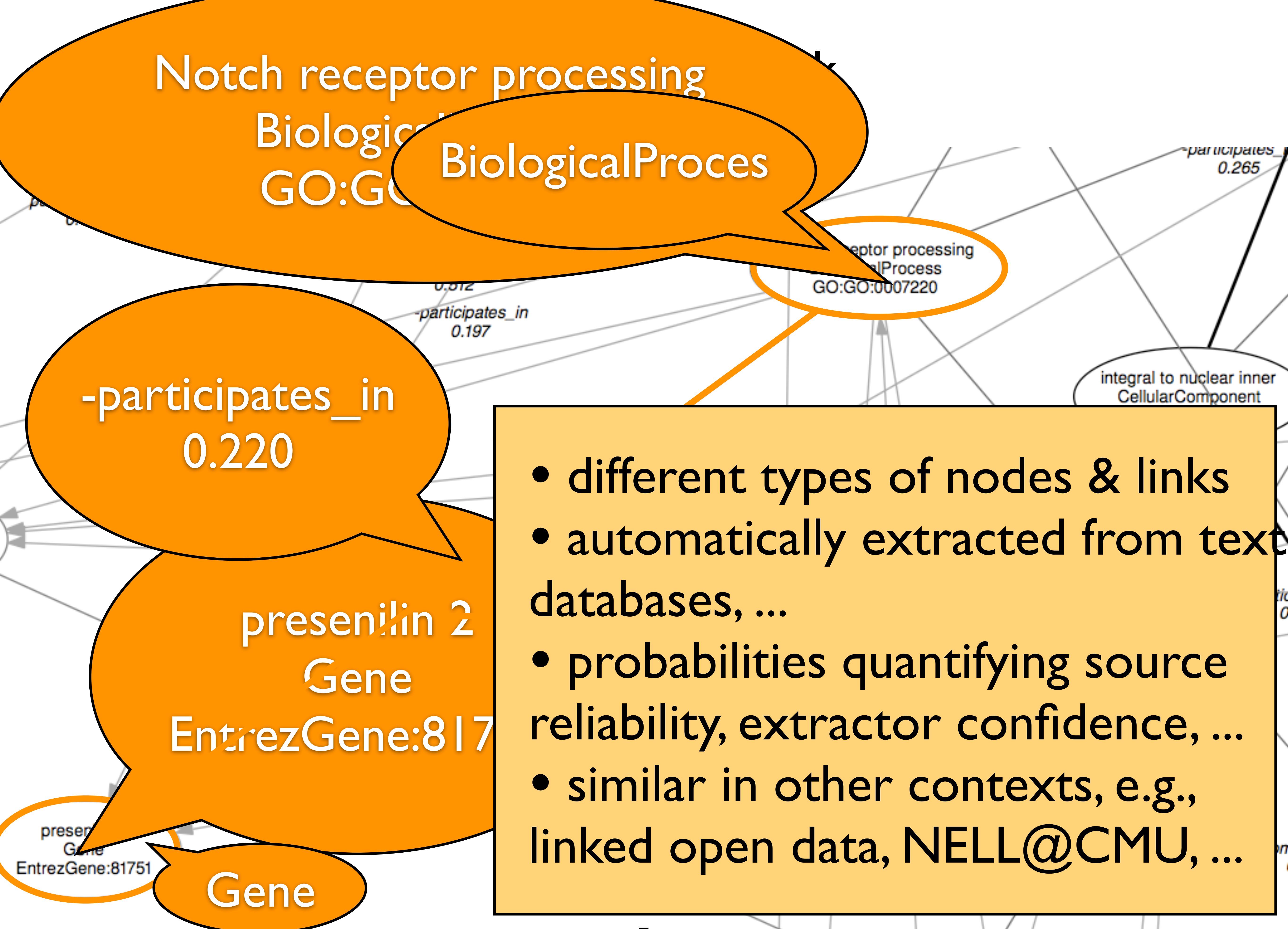
Statistical relational learning, probabilistic logic learning, probabilistic programming, ...

Networks of Uncertain Information



Biomine database @ Helsinki
http://biomine.cs.helsinki.fi/


Biomine
network



- different types of nodes & links
- automatically extracted from text, databases, ...
- probabilities quantifying source reliability, extractor confidence, ...
- similar in other contexts, e.g., linked open data, NELL@CMU, ...

Example: Information Extraction

Recently-Learned Facts

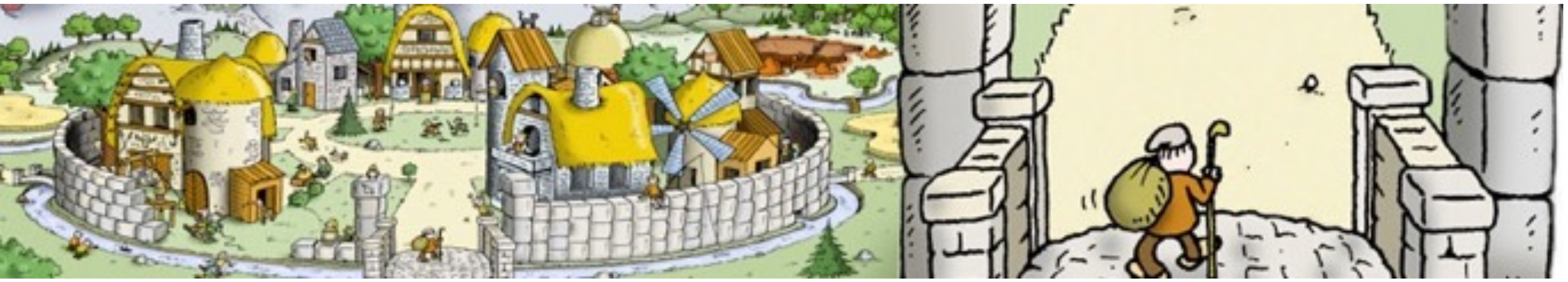
instance	iteration	date learned	confidence
kelly_andrews is a female	826	29-mar-2014	98.7
investment_next_year is an economic sector	829	10-apr-2014	95.3
shibenik is a geopolitical entity that is an organization	829	10-apr-2014	97.2
quality_web_design_work is a character trait	826	29-mar-2014	91.0
mercedes_benz_cls_by_carlsson is an automobile manufacturer	829	10-apr-2014	95.2
social_work is an academic program at the university rutgers_university	827	02-apr-2014	93.8
dante wrote the book the_divine_comedy	826	29-mar-2014	93.8
willie_aames was born in the city los_angeles	831	16-apr-2014	100.0
kitt_peak is a mountain in the state or province arizona	831	16-apr-2014	96.9
greenwich is a park in the city london	831	16-apr-2014	100.0

instances for many different relations

degree of certainty

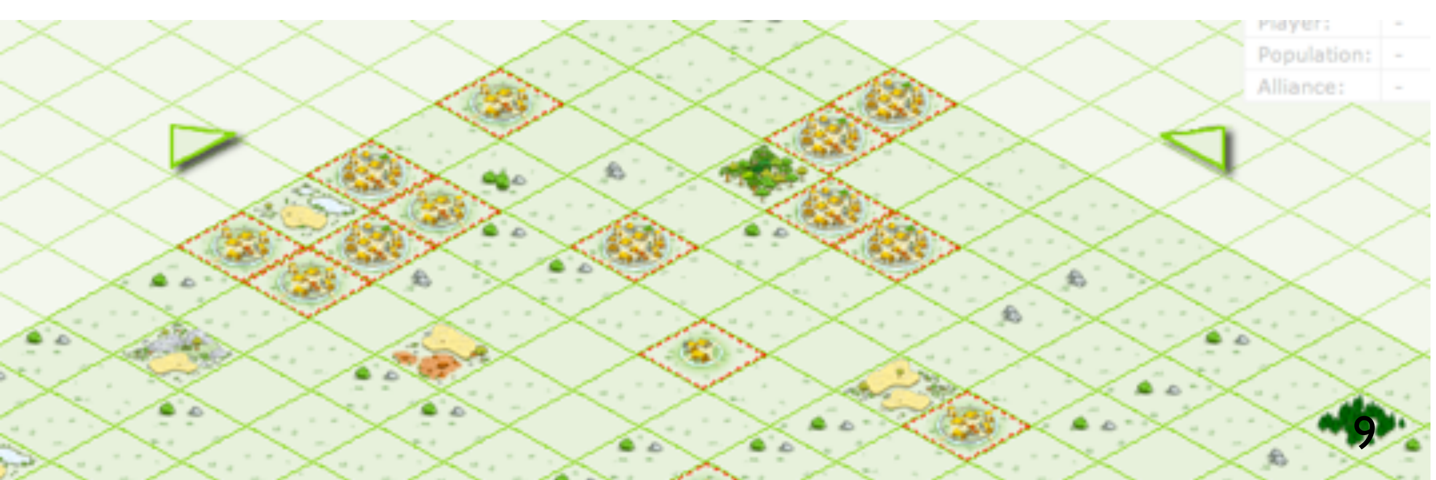
NELL: http://rtw.ml.cmu.edu/rtw/

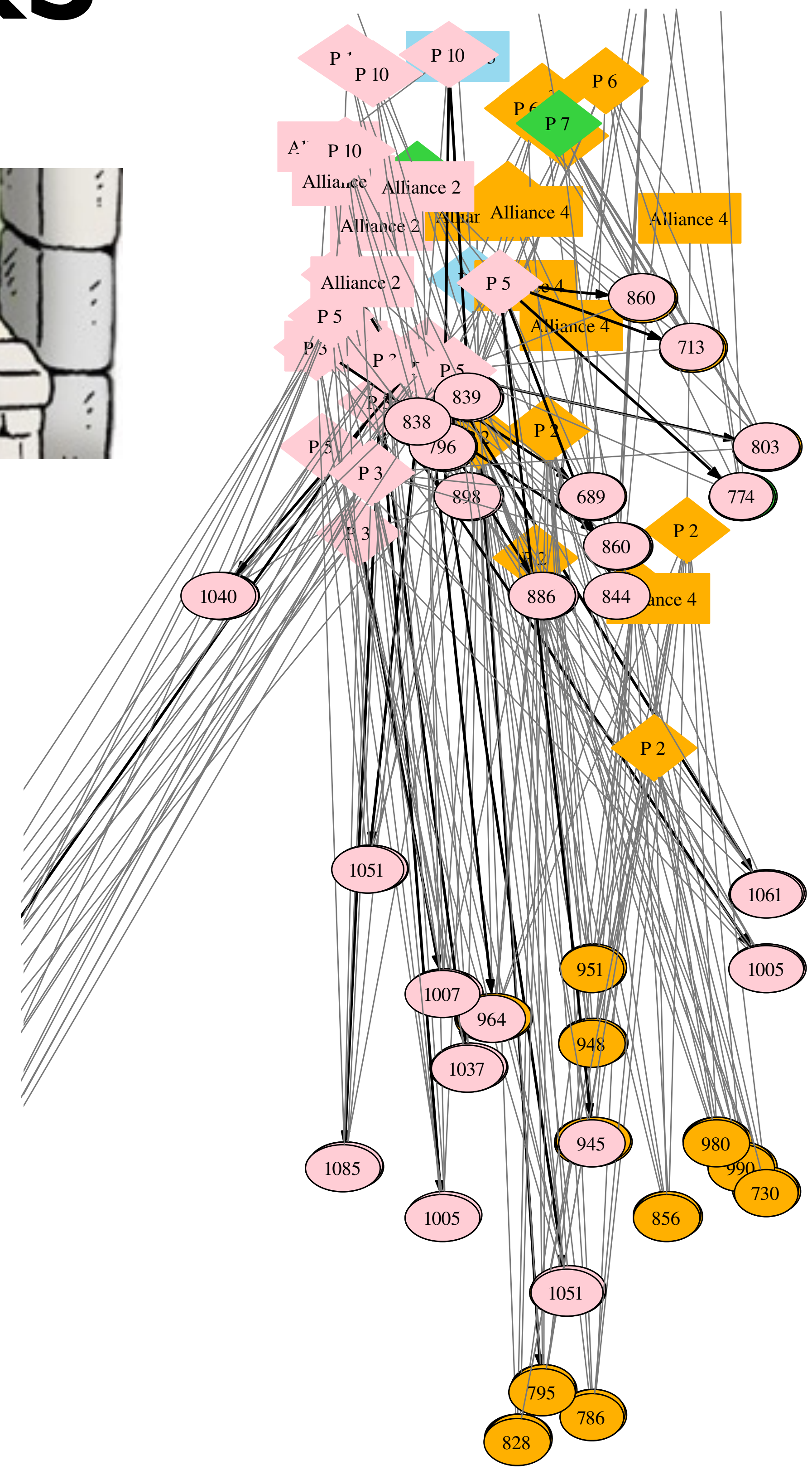
Dynamic networks



Travian: A massively multiplayer real-time strategy game

Can we build a model
of this world ?
Can we use it for playing
better ?





[Thon et al, MLJ 11]

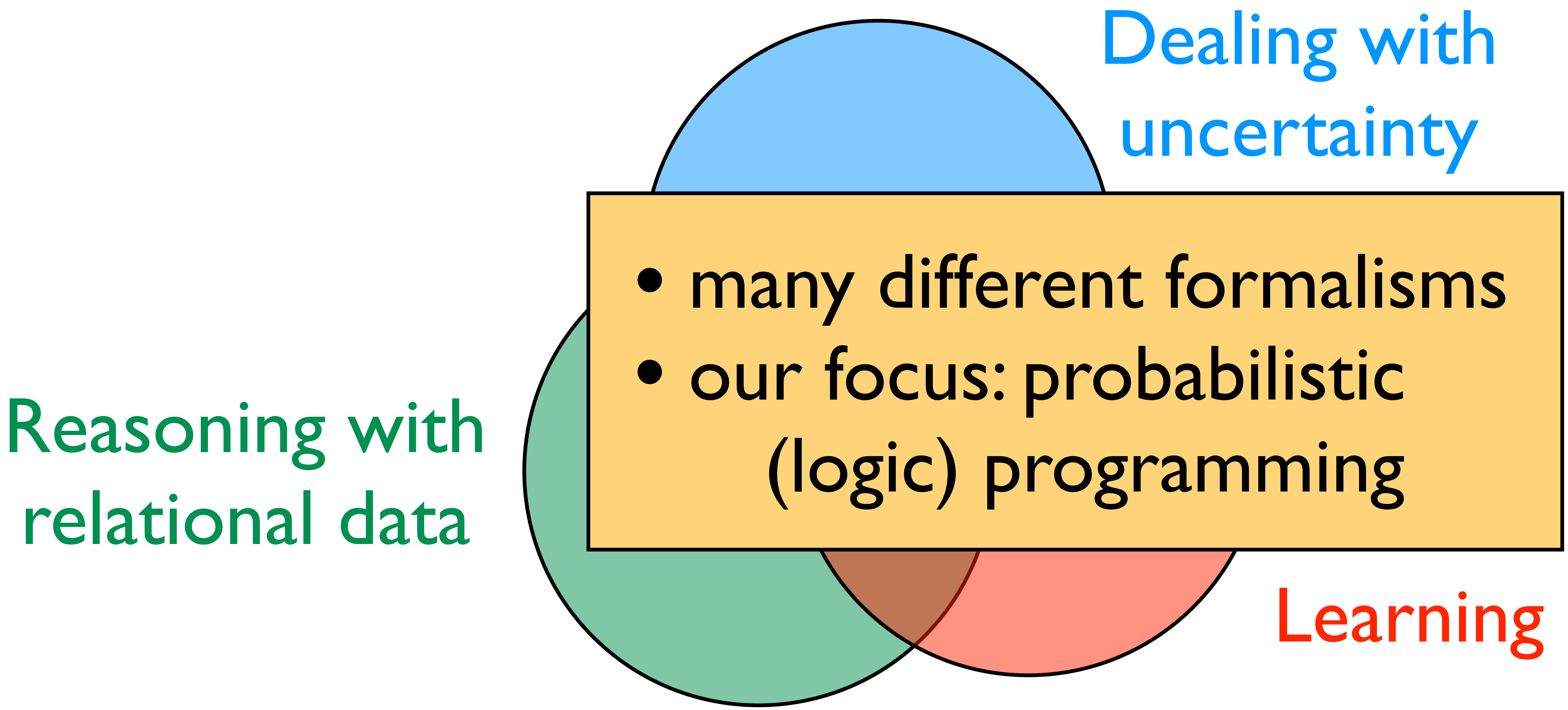
Answering Probability Questions

Mike has a bag of marbles with 4 white, 8 blue, and 6 red marbles. He pulls out one marble from the bag and it is red. What is the probability that the second marble he pulls out of the bag is white?

The answer is 0.235941.

[Dries et al., IJCAI 17]

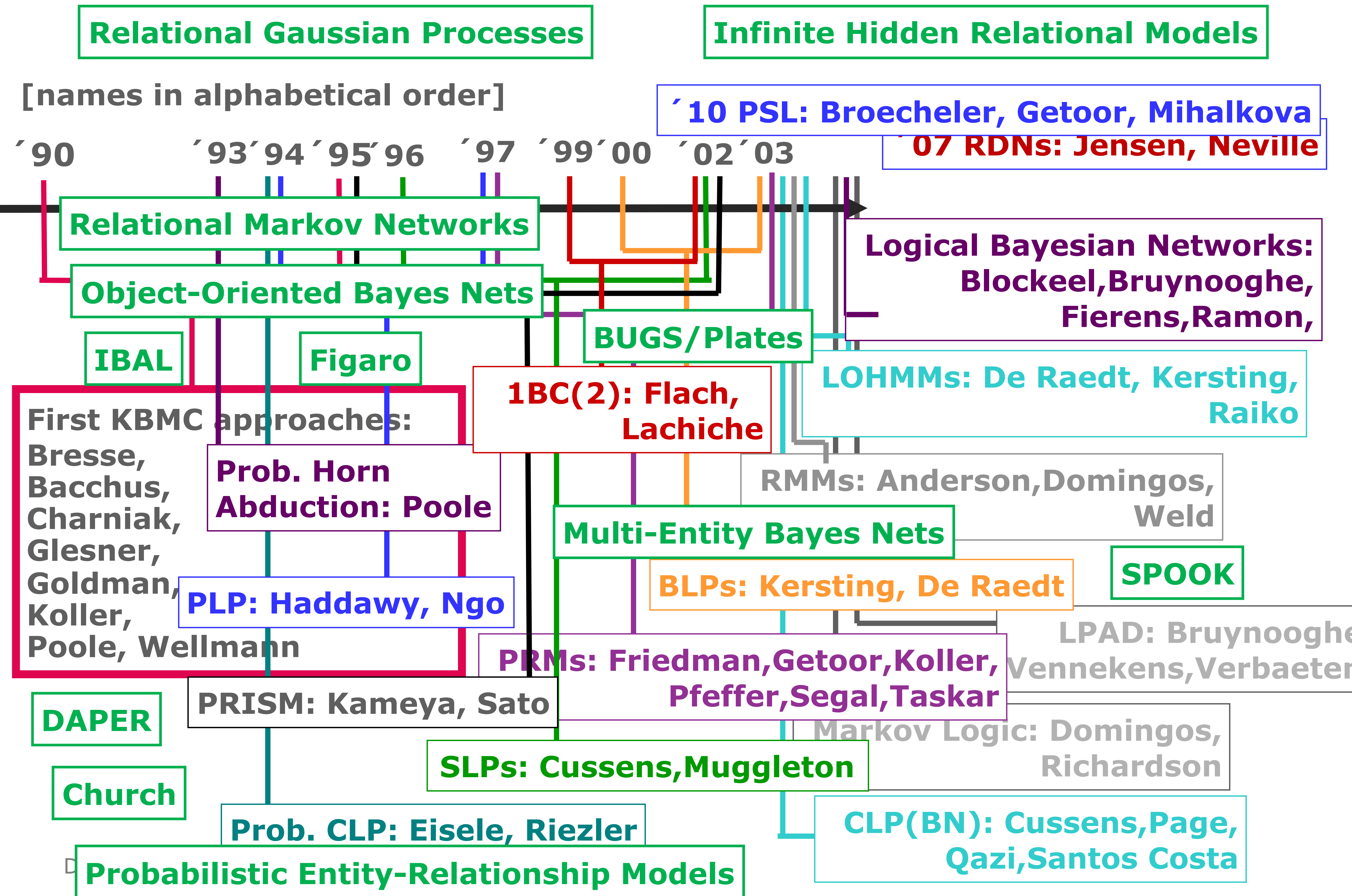
Common theme



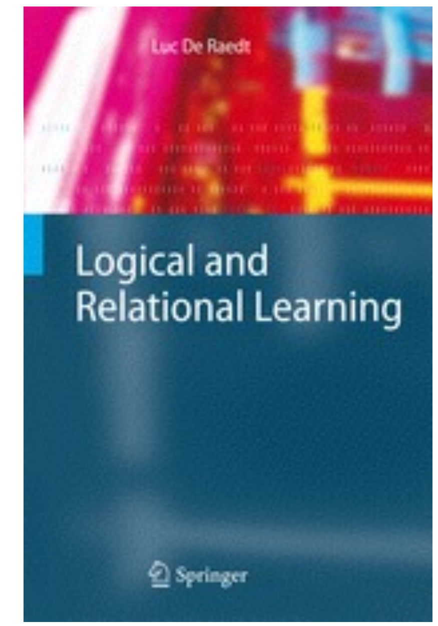
- many different formalisms
- our focus: probabilistic (logic) programming

Statistical relational learning, probabilistic logic learning, probabilistic programming, ...

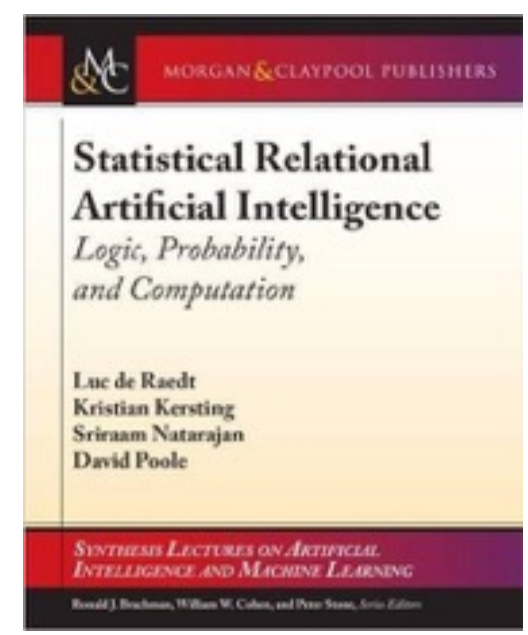
The (Incomplete) SRL Alphabet Soup



Many different angles



- Probabilistic programming
  - Logic programming and probabilistic databases
    - (ProbLog and DS as representatives)
  - Functional and imperative (Church as representatives)



- Statistical relational AI and learning
  - Markov Logic
  - Relational Bayesian Networks (and variants)

Probabilistic Logic Programs

- devised by Poole and Sato in the 90s.
- built on top of the *programming language* Prolog
- upgrade *directed* graphical models
  - combines the advantages / expressive power of programming languages (Turing equivalent) and graphical models
- Generalises probabilistic databases (Suciu et al.)
- Implementations include: PRISM, ICL, ProbLog, LPADs, CP-logic, Dyna, Pita, DC, …

Roadmap

- Modeling
- Reasoning
- Learning
- Dynamics
- Decisions

Part 1 : Modeling

ProbLog

probabilistic Prolog

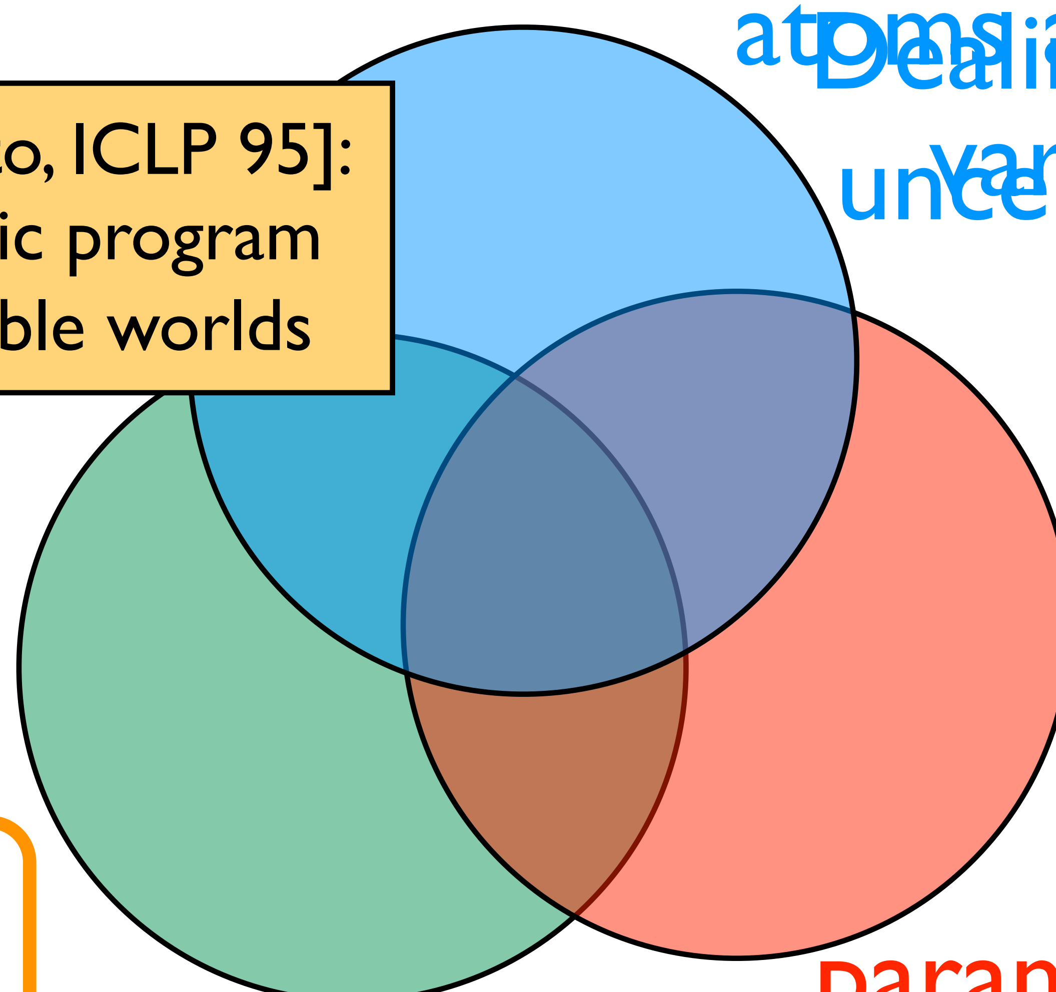
several possible worlds

```
0.8::stress(ann).
0.6::influences(ann,bob).
0.2::influences(bob,carl).
```

Dealing with uncertainty

atoms as random variables

Distribution Semantics [Sato, ICLP 95]:
probabilistic choices + logic program
→ distribution over possible worlds



Reasoning with relational data

Prolog / logic programming

```
stress(ann).
influences(ann,bob).
influences(bob,carl).
```

one world

Learning

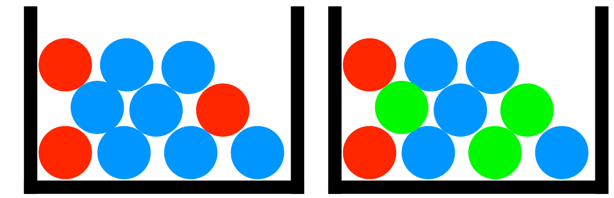
parameter learning, adapted relational learning techniques

```
smokes(X) :- stress(X).
smokes(X) :-
      influences(Y,X), smokes(Y).
```


http://dtai.cs.kuleuven.be/problog/

ProbLog by example:

A bit of gambling



- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

probabilistic fact: heads is true with probability 0.4 (and false with 0.6)

annotated disjunction: first ball is red with probability 0.3 and blue with 0.7

probabilistic choices

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue).
```

annotated disjunction: second ball is red with probability 0.2, green with 0.3, and blue with 0.5

logical rule encoding background knowledge

consequences

```
win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

Questions

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

marginal probability

- Probability of `win`?

query **conditional probability**

- Probability of `win` given `col(2,green)`?

evidence

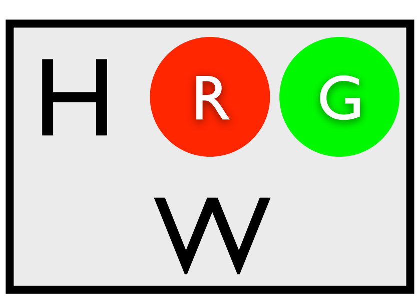
- Most probable world where `win` is true?

MPE inference

Possible Worlds

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue).
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue).

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

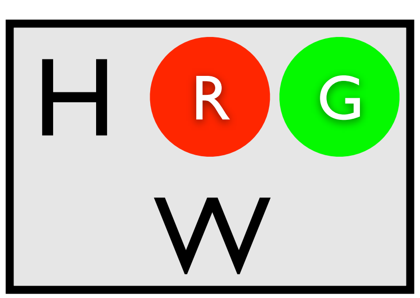
$0.4 \times 0.3 \times 0.3$



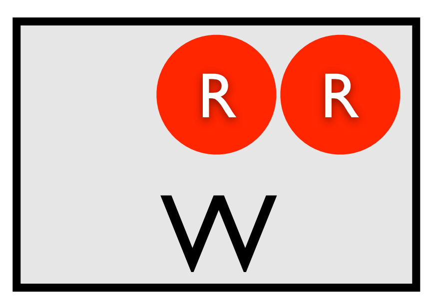
Possible Worlds

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

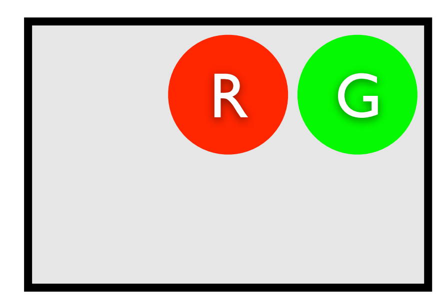
$0.4 \times 0.3 \times 0.3$



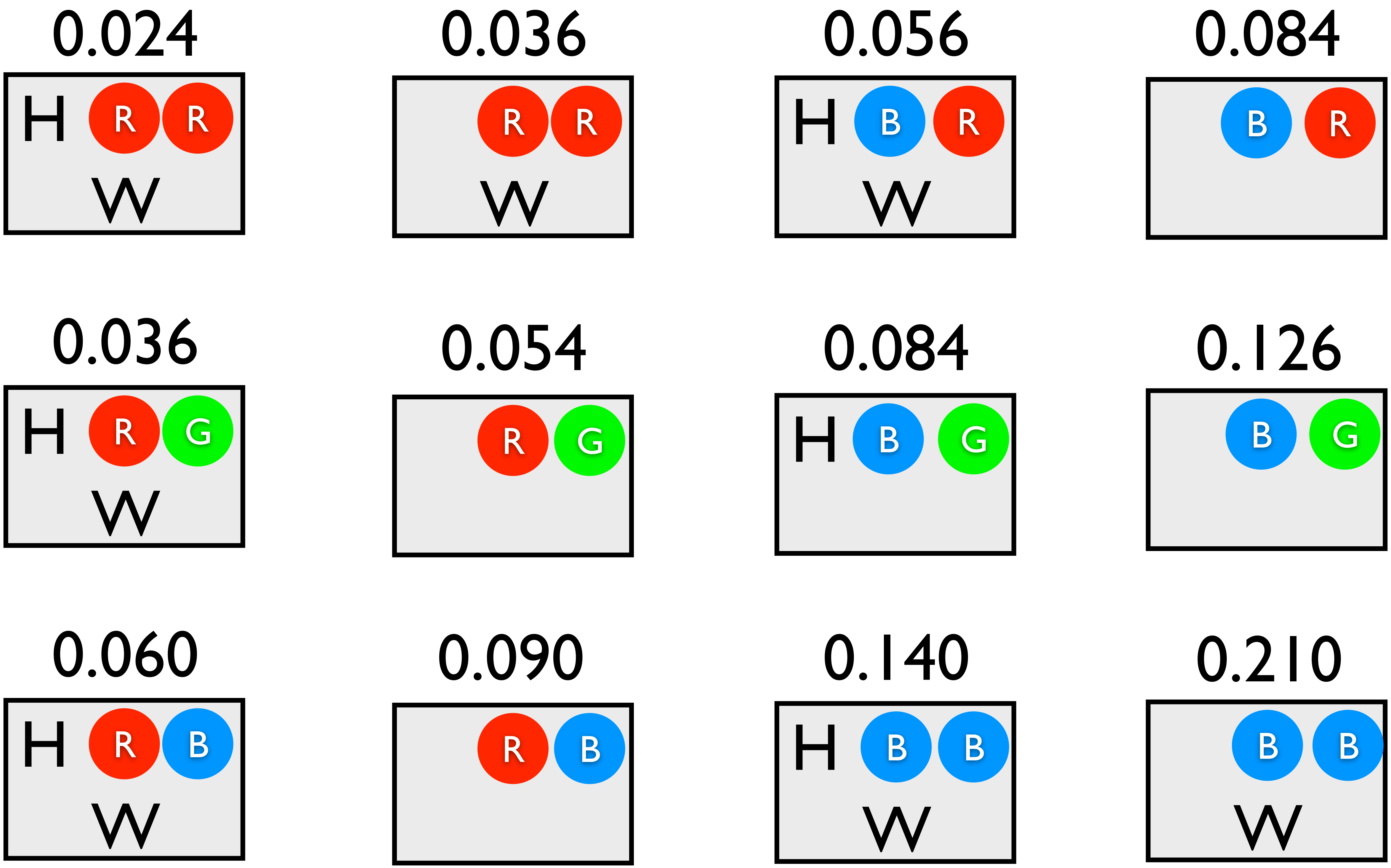
$(1-0.4) \times 0.3 \times 0.2$



$(1-0.4) \times 0.3 \times 0.3$

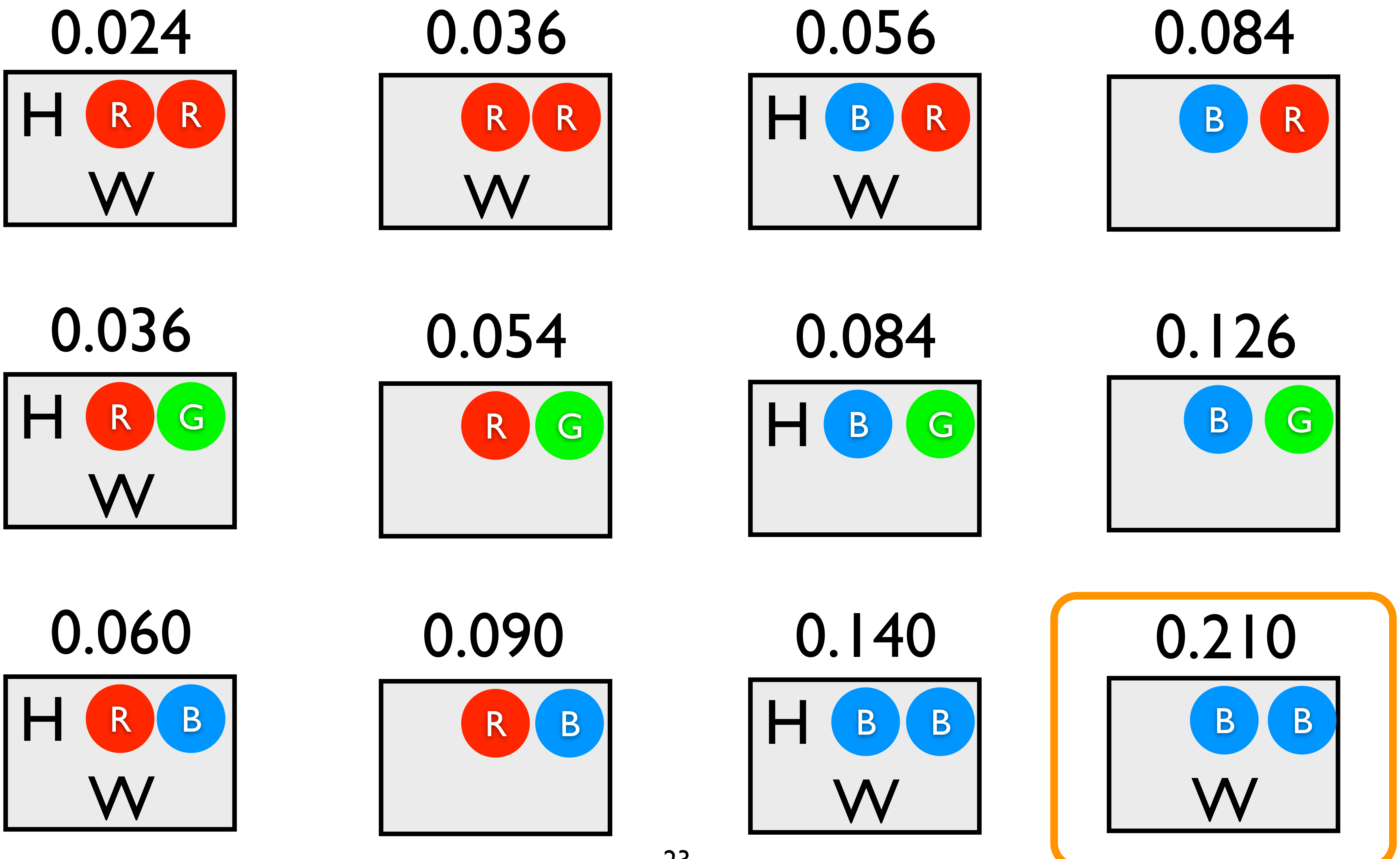


All Possible Worlds



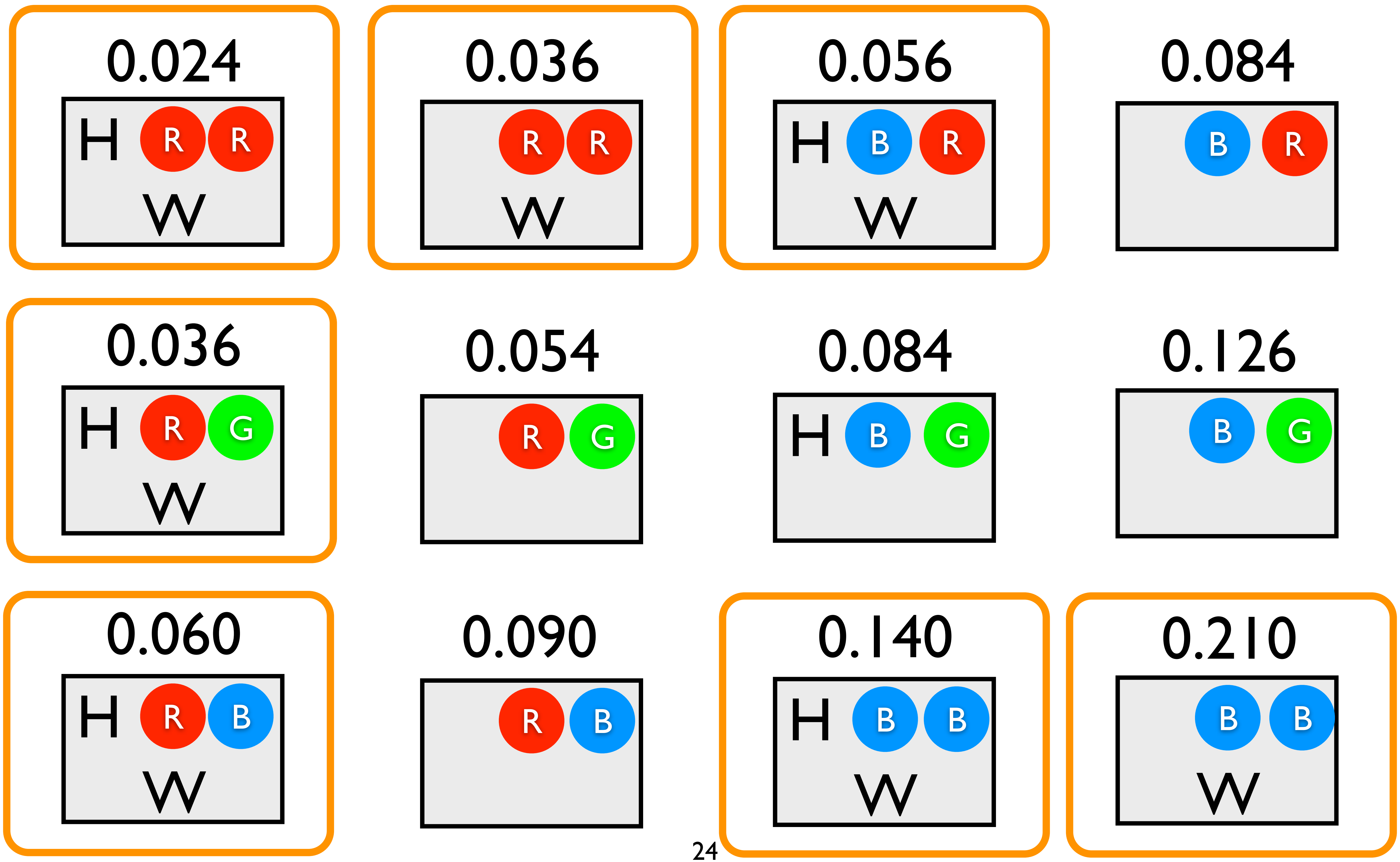
Most likely world where **`win`** is true?

MPE Inference



$P(\underline{win}) = ?\sum = 0.562$

Marginal Probability

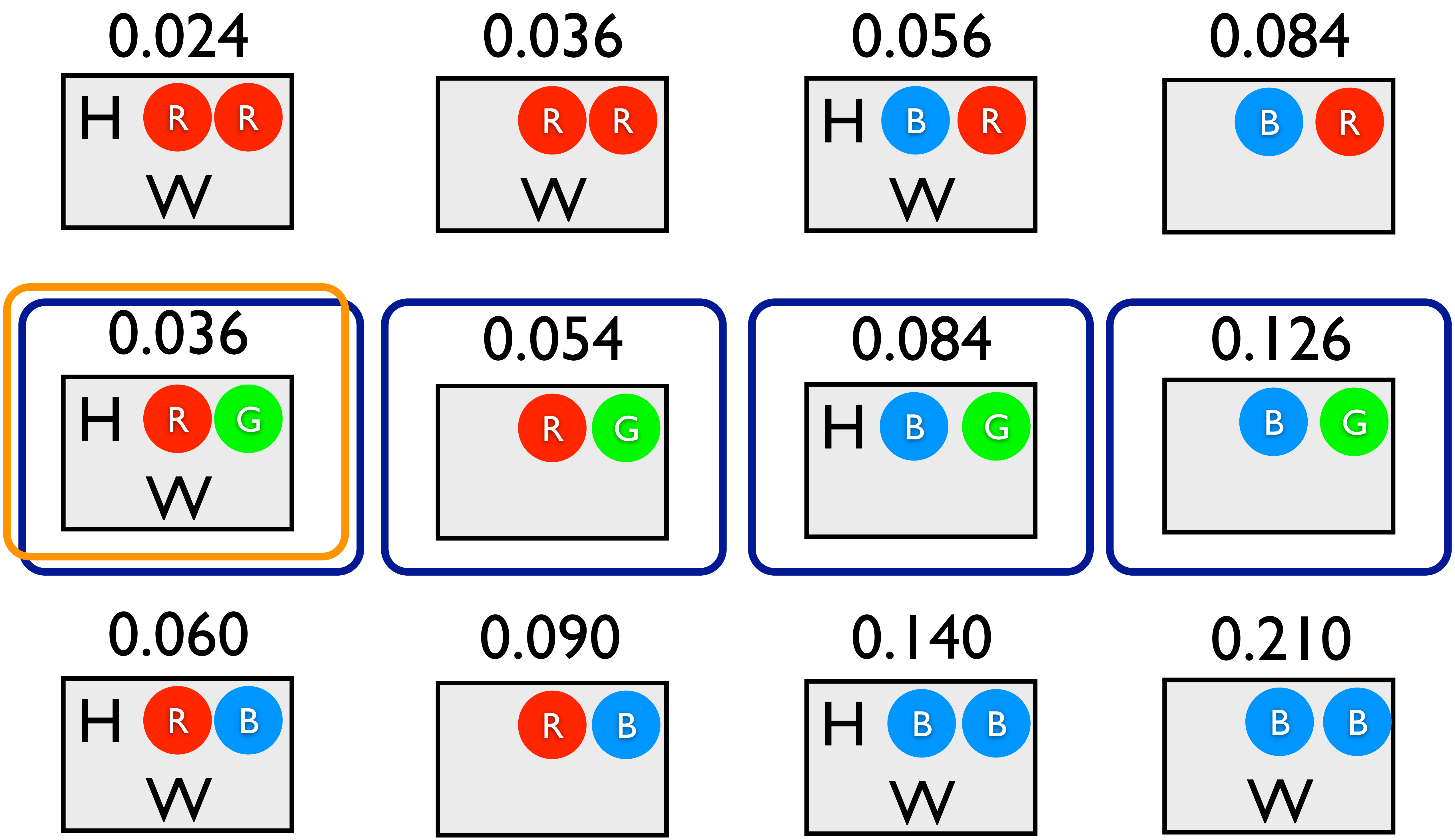


Conditional Probability

$P(win|col(2,green)) = ?$ Σ/Σ

$= P(win \wedge col(2,green))/P(col(2,green))$

$= 0.036/0.3 = 0.12$



Distribution Semantics

(with probabilistic facts)

[Sato, ICLP 95]

query

sum over possible worlds where Q is true

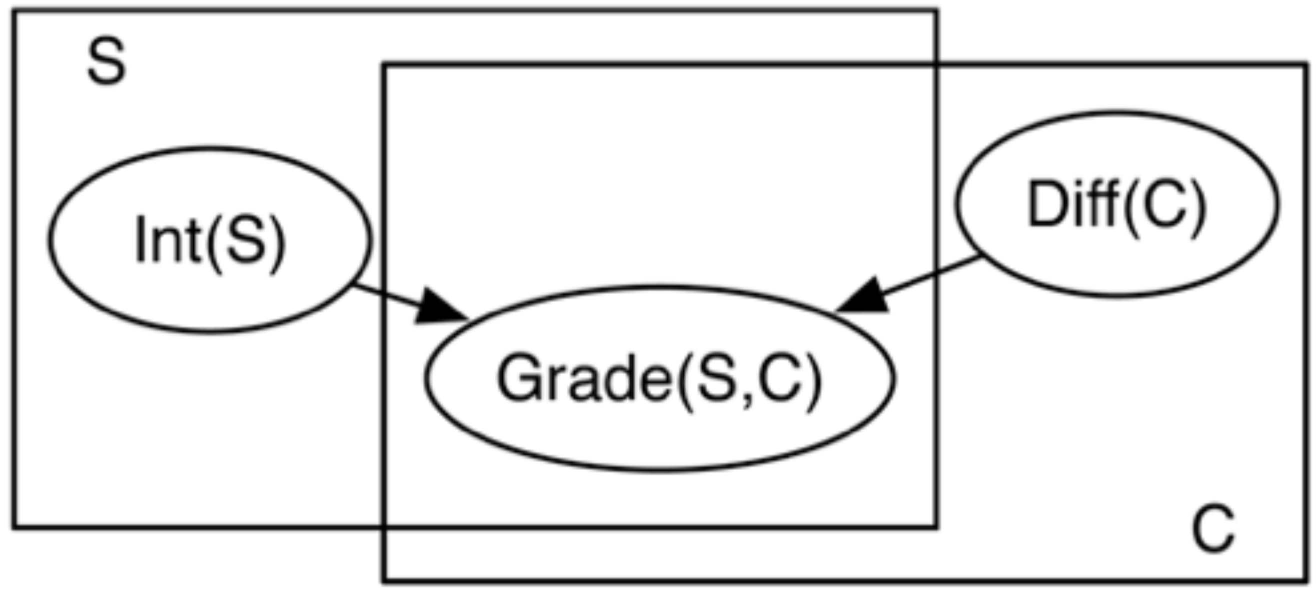
$$P(Q) = \sum_{F \cup R \models Q} \underbrace{\prod_{f \in F} p(f) \prod_{f \notin F} 1 - p(f)}_{\text{probability of possible world}}$$

subset of probabilistic facts

Prolog rules

probability of possible world

Flexible and Compact Relational Model for Predicting Grades



"Program" Abstraction:

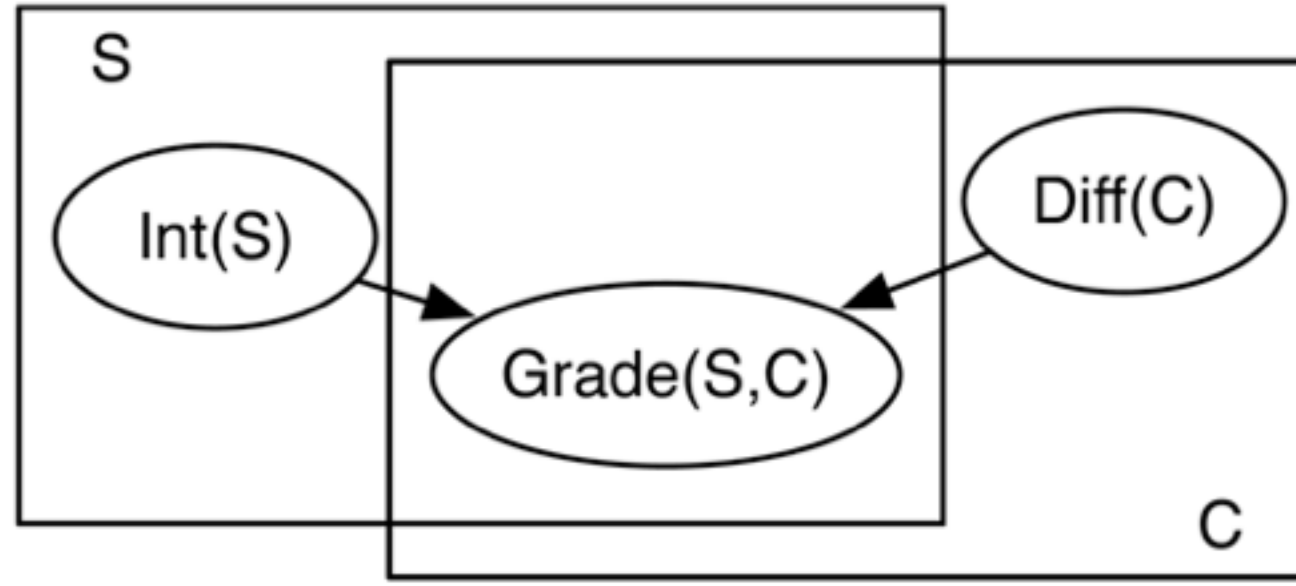
- S, C logical variable representing students, courses
- the set of individuals of a type is called a population
- Int(S), Grade(S, C), D(C) are parametrized random variables

Grounding:

- for every student s, there is a random variable Int(s)
- for every course c, there is a random variable Di(c)
- for every s, c pair there is a random variable Grade(s,c)
- all instances share the same structure and parameters

ProbLog by example:

Grading



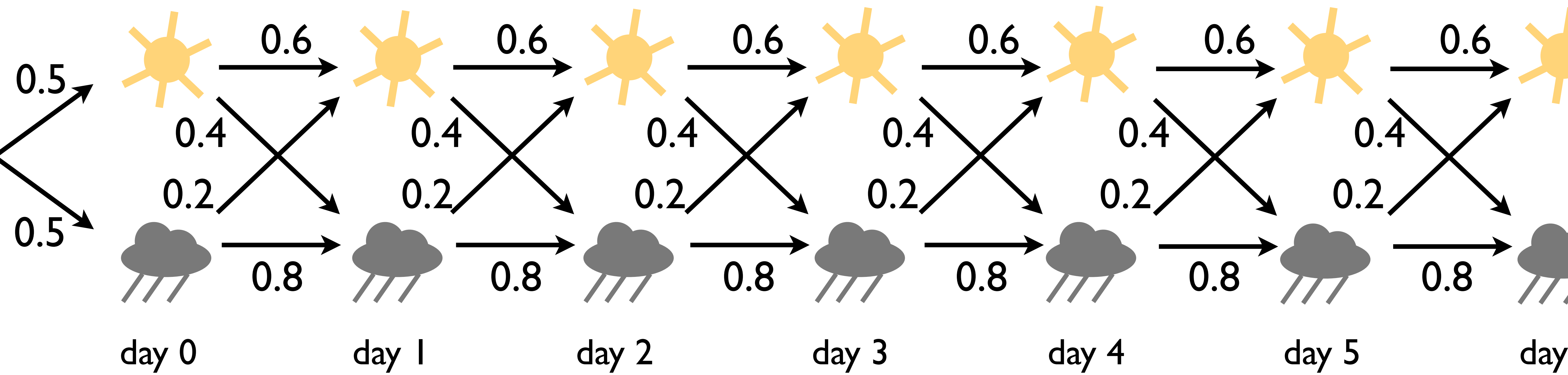

```
0.4 :: int(S) :- student(S).
0.5 :: diff(C):- course(C).

student(john). student(anna). student(bob).
course(ai).    course(ml).    course(cs).

gr(S,C,a) :- int(S), not diff(C).
0.3::gr(S,C,a); 0.5::gr(S,C,b);0.2::gr(S,C,c) :-
           int(S), diff(C).
0.1::gr(S,C,b); 0.2::gr(S,C,c); 0.2::gr(S,C,f) :-
           student(S), course(C),
           not int(S), not diff(C).
0.3::gr(S,C,c); 0.2::gr(S,C,f) :-
           not int(S), diff(C).
```

ProbLog by example: Grading

```
unsatisfactory(S) :- student(S), grade(S,C,f).

excellent(S) :- student(S), not grade(S,C,G), below(G,a).
excellent(S) :- student(S), grade(S,C,a).

0.4 :: int(S) :- student(S).
0.5 :: diff(C):- course(C).

student(john). student(anna). student(bob).
course(ai).    course(ml).    course(cs).

gr(S,C,a) :- int(S), not diff(C).
0.3::gr(S,C,a); 0.5::gr(S,C,b);0.2::gr(S,C,c) :-
           int(S), diff(C).
0.1::gr(S,C,b); 0.2::gr(S,C,c); 0.2::gr(S,C,f) :-
           student(S), course(C),
           not int(S), not diff(C).
0.3::gr(S,C,c); 0.2::gr(S,C,f) :-
           not int(S), diff(C).
```

ProbLog by example:

Rain or sun?

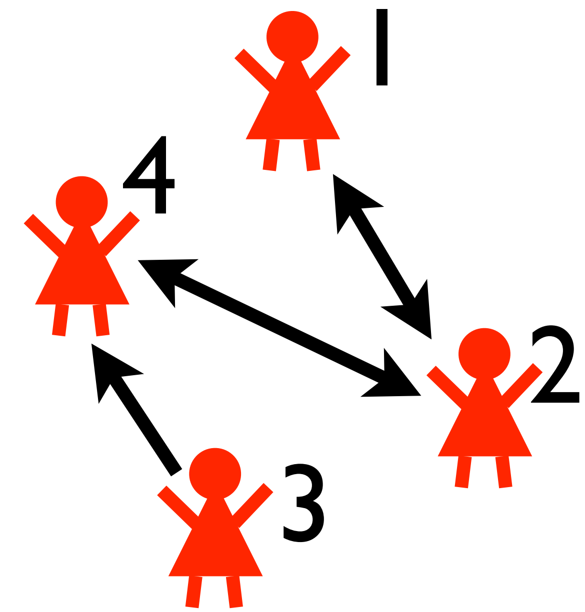



```
0.5::weather(sun,0) ; 0.5::weather(rain,0) <- true.

0.6::weather(sun,T) ; 0.4::weather(rain,T)
                <- T>0, Tprev is T-1, weather(sun,Tprev).
0.2::weather(sun,T) ; 0.8::weather(rain,T)
                <- T>0, Tprev is T-1, weather(rain,Tprev).
```

infinite possible worlds! BUT: finitely many partial worlds suffice to answer any given ground query

ProbLog by example:

Friends & smokers



typed probabilistic facts

= a probabilistic fact for each grounding

```
0.3::stress(X):- person(X).
0.2::influences(X,Y):-
                 person(X), person(Y).

smokes(X) :- stress(X).
smokes(X) :-
    friend(X,Y), influences(Y,X), smokes(Y).

0.4::asthma(X) <- smokes(X).
```

```
person(1).
person(2).
person(3).
person(4).

friend(1,2).
friend(2,1).
friend(2,4).
friend(3,4).
friend(4,2).
```

```
0.3::stress(1).
0.3::stress(2).
0.3::stress(3).
0.3::stress(4).
```

```
0.2::influences(1,1).
0.2::influences(1,2).
0.2::influences(1,3).
0.2::influences(1,4).
0.2::influences(2,1).
...
0.2::influences(4,2).
0.2::influences(4,3).
0.2::influences(4,4).
```

annotated disjunction with implicit head atom:

with probability 0.6, nothing happens

Concept: flexible probability

ProbLog by example:

Limited Luggage

```
weight(skis,6).
weight(boots,4).
weight(helmet,3).
weight(gloves,2).

1/6::pack(skis).
1/4::pack(boots).
1/3::pack(helmet).
1/2::pack(gloves).

P::pack(Item) :- weight(Item,Weight), P is 1.0/Weight.
excess(Limit) :- excess([skis,boots,helmet,gloves],Limit).

excess([],Limit) :- Limit<0.
excess([I|R],Limit) :-
   pack(I), weight(I,W), L is Limit-W, excess(R,L).
excess([I|R],Limit) :-
   \+pack(I), excess(R,Limit).
```

flexible probability: computed from the weight of the item

list of all items

no items left: did we add too much?

pack first item, decrease limit by its weight, and continue with rest of items

do **not** pack first item, continue with rest of items

[Suciu et al 2011]

Probabilistic Databases

several possible worlds

bornIn

person	city	P
ann	london	0,87
bob	york	0,95
		0,9
		0,56

cityIn

city	country	P
london	uk	0,99
york	uk	0,75
paris	usa	0,4

probabilistic tables + database queries
→ distribution over possible worlds

```
select
from bornIn x, cityIn y
where x.city=y.city
```

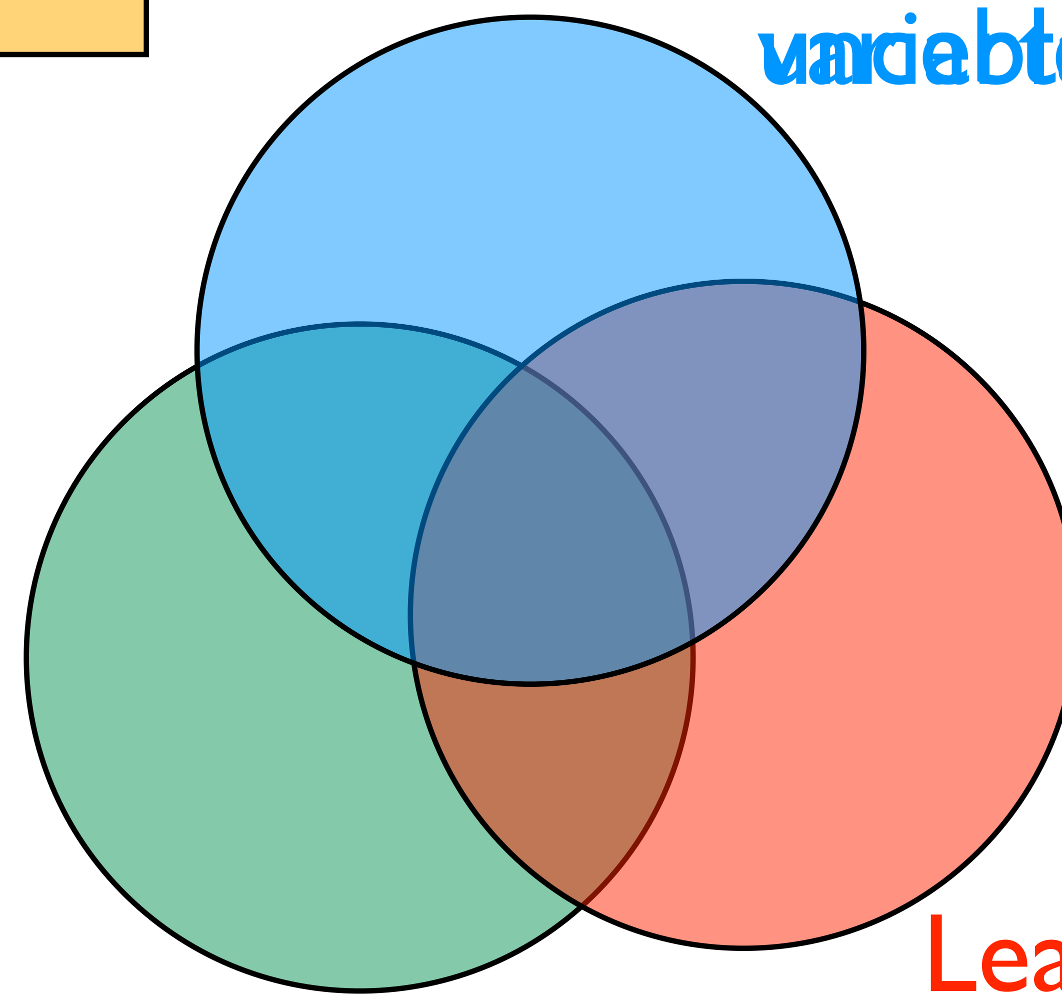
one world

bornIn

person	city
ann	london
bob	york
eve	new york
tom	paris

cityIn

city	country
london	uk
york	uk
paris	usa



Example: Information Extraction

Recently-Learned Facts twitter

Refresh

instance	iteration	date learned	confidence
kelly_andrews is a female	826	29-mar-2014	98.7
investment_next_year is an economic sector	829	10-apr-2014	95.3
shibenik is a geopolitical entity that is an organization	829	10-apr-2014	97.2
quality_web_design_work is a character trait	826	29-mar-2014	91.0
mercedes_benz_cls_by_carlsson is an automobile manufacturer	829	10-apr-2014	95.2
social_work is an academic program at the university rutgers_university	827	02-apr-2014	93.8
dante wrote the book the_divine_comedy	826	29-mar-2014	93.8
willie_aames was born in the city los_angeles	831	16-apr-2014	100.0
kitt_peak is a mountain in the state or province arizona	831	16-apr-2014	96.9
greenwich is a park in the city london	831	16-apr-2014	100.0

instances for many different relations

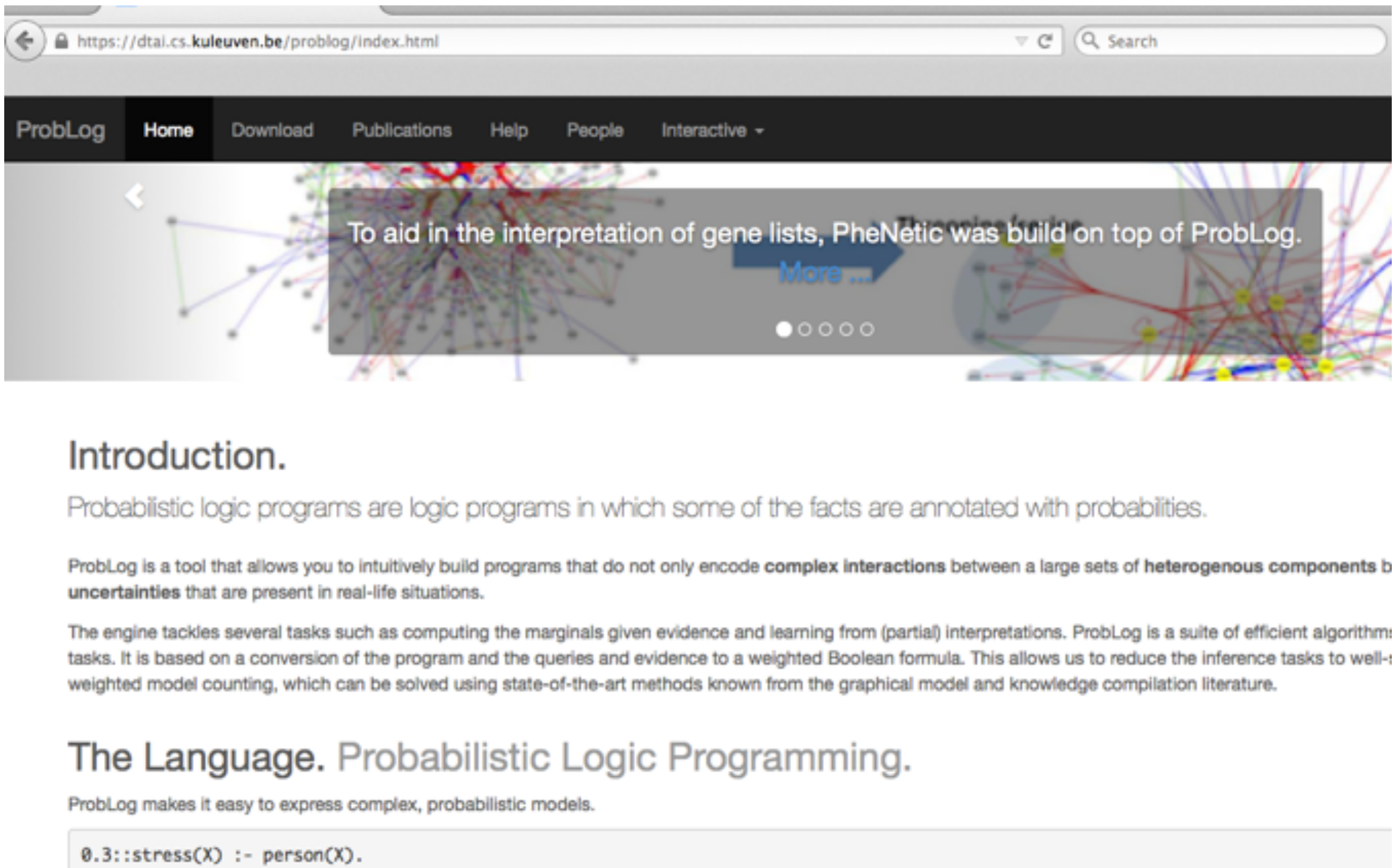
degree of certainty

NELL: http://rtw.ml.cmu.edu/rtw/

Distribution Semantics

- **probabilistic choices** + their **consequences**
- probability distribution over **possible worlds**
- how to efficiently answer **questions**?
  - most probable world (MPE inference)
  - probability of query (computing marginals)
  - probability of query given evidence

http://dtai.cs.kuleuven.be/problog



Church

probabilistic functional programming

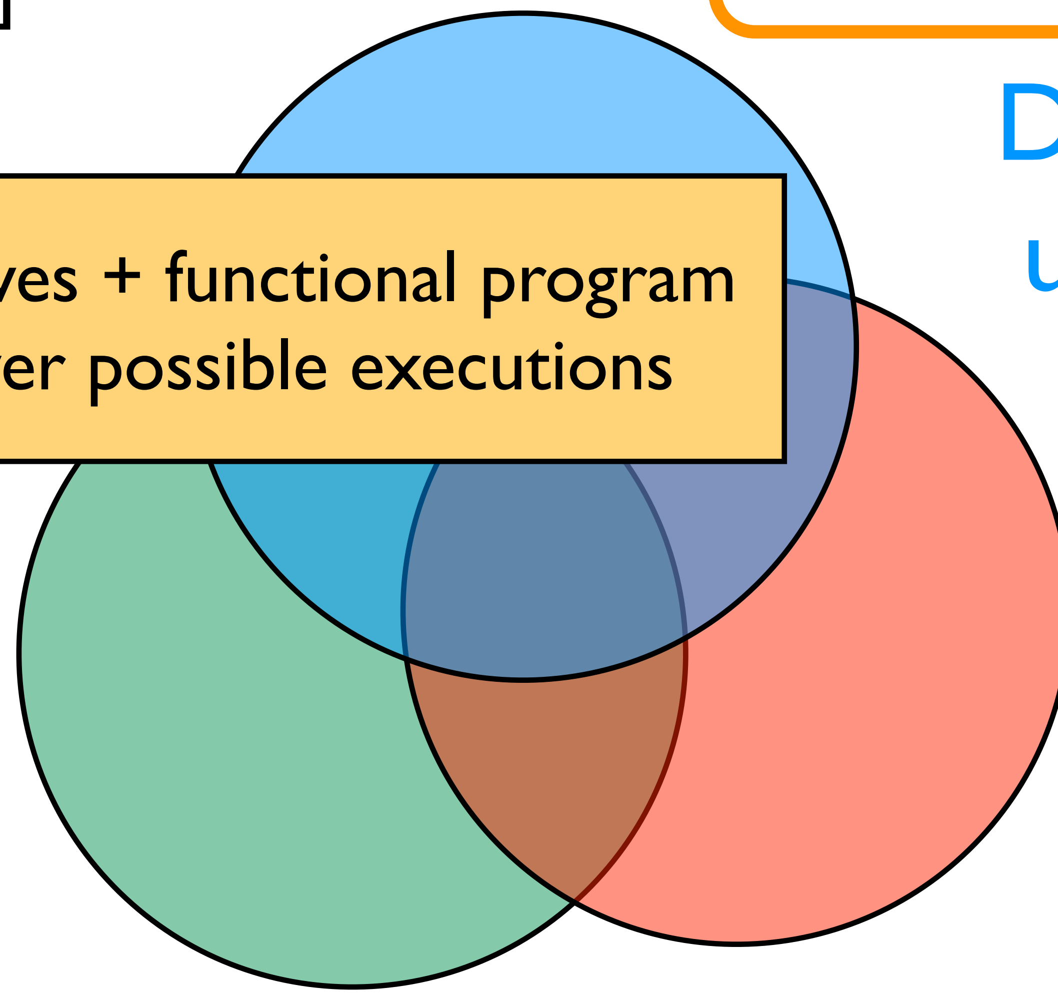
[Goodman et al, UAI 08]

several possible executions

```
(define randplus5
 (lambda (x) (if (flip 0.6)
                 (+ x 5)
                x)))

(map randplus5 '(1 2 3))
```

Dealing with uncertainty

random primitives



probabilistic primitives + functional program
→ distribution over possible executions

Reasoning with relational data

functional programming

Learning

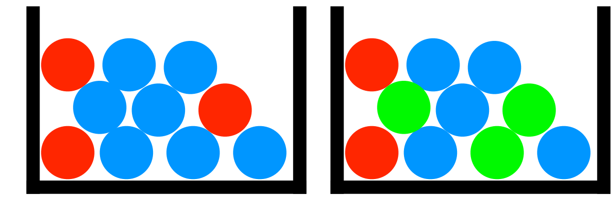
one execution

```
(define plus5 (lambda (x) (+ x 5)))

(map plus5 '(1 2 3))
```


http://probmods.org

Church by example:

A bit of gambling



- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
(define heads (mem (lambda () (flip 0.4))))

(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))

(define color2 (mem (lambda ()
                 (multinomial '(red green blue) '(0.2 0.3 0.5)))))

(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))

(define win1 (and (heads) redball))

(define win2 (equal? (color1) (color2)))

(define win (or win1 win2))
```

Probabilistic Programming Ingredients

inject random variables in programming language

- random var = simplest boolean concept in language
- extensions for continuous variables exist

define semantics :

- possible worlds for probabilistic logics / databases / StarAI
- trace-based semantics for functional / imperative languages
    - define probability of execution, whenever random variable encountered -> multiply by probabilty

inference to answer (conditional) queries

learning

- Bayesian inference : conditioning on observations and querying for distributions (typical for functional and imperative languages)
- Expected Maximisation (typical for Probabilistic Logics & StarAI)

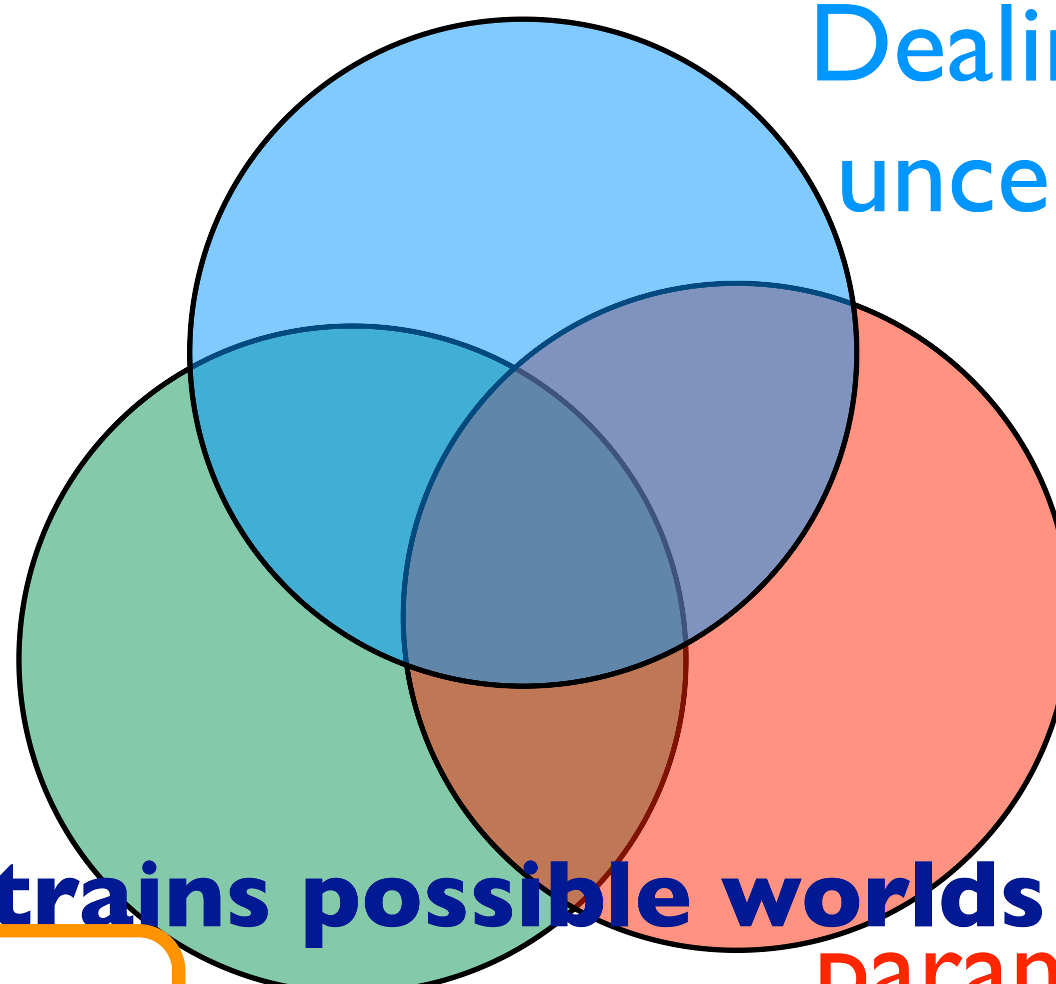
see also [Russell, CACM 15]

Markov Logic

soft constraints

```
12: for all X: stress(X) -> smokes(X).
3: for all X,Y: smokes(X), influences(X,Y) ->
        smokes(Y).
```

Dealing with
uncertainty



Reasoning with
First order logic
relational data

Constrains possible worlds

Learning
parameter learning,
adapted relational
learning techniques

```
{stress(ann), smokes(ann)}
{smokes(ann)}
```

```
for all X: stress(X) -> smokes(X).
for all X,Y: smokes(X), influences(X,Y) ->
        smokes(Y).
```


https://alchemy.cs.washington.edu/

Markov Logic: Intuition

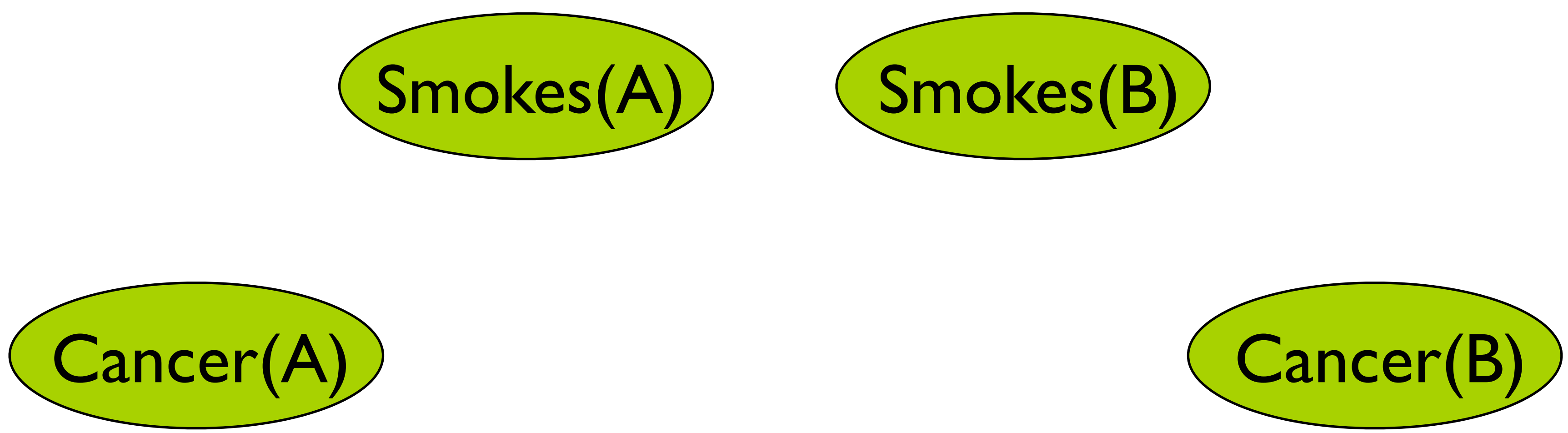
- *Undirected graphical model*
- A logical KB is a set of **hard constraints** on the set of possible worlds
- Let's make them **soft constraints**:
  When a world violates a formula,
  it becomes less probable, not impossible
- Give each formula a **weight**
  (Higher weight $\Rightarrow$ Stronger constraint)

$$P(\text{world}) \propto \exp\left(\sum \text{weights of formulas it satisfies}\right)$$

Markov Logic

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x, y) \Rightarrow \big(Smokes(x) \Leftrightarrow Smokes(y)\big)$

Suppose we have two constants: **Anna** (A) and **Bob** (B)

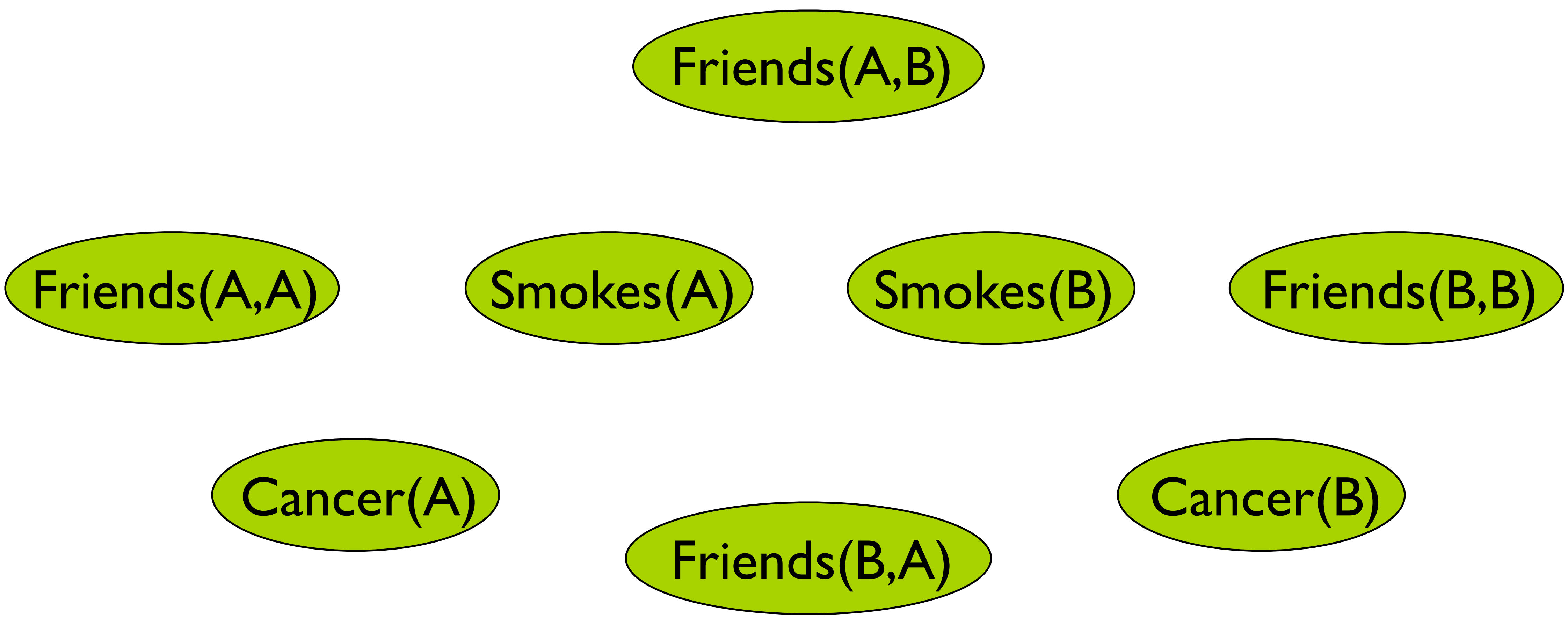


slides by Pedro Domingos

Markov Logic

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x, y) \Rightarrow \big(Smokes(x) \Leftrightarrow Smokes(y)\big)$

Suppose we have two constants: **Anna** (A) and **Bob** (B)

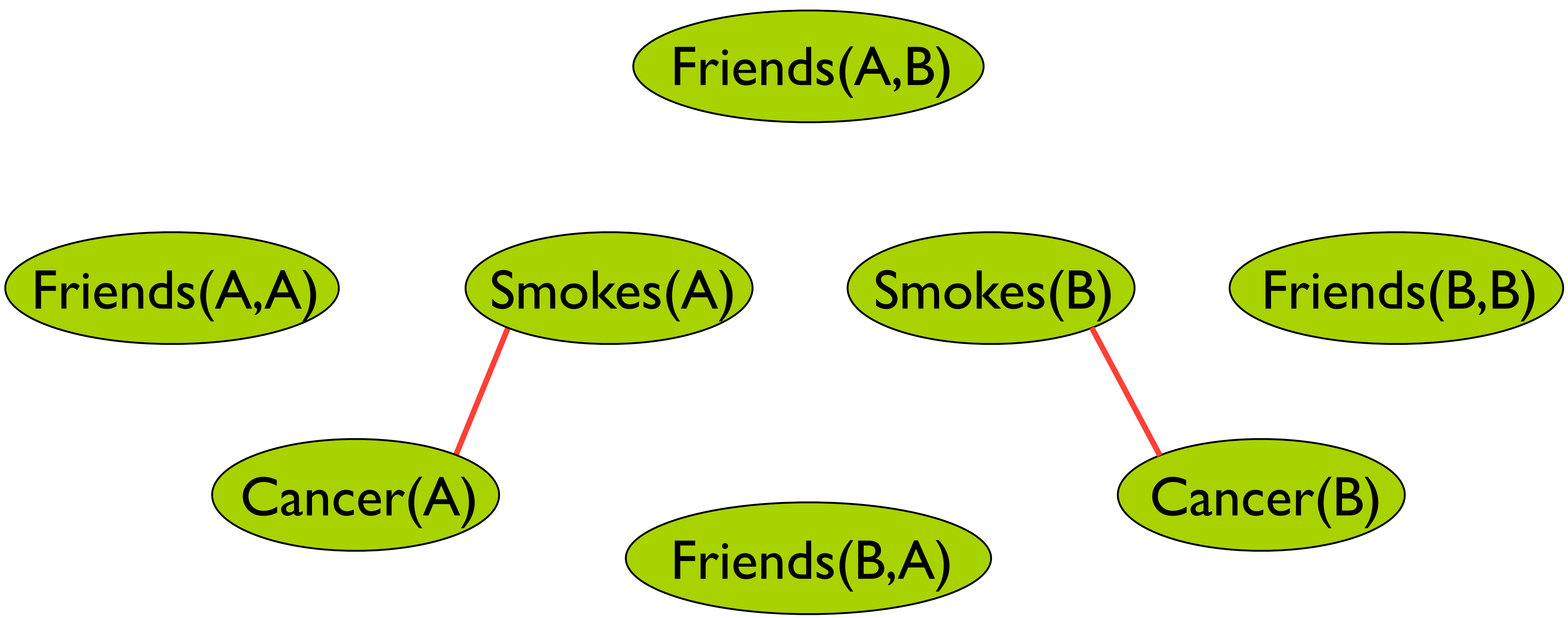


slides by Pedro Domingos

Markov Logic

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x, y) \Rightarrow \big(Smokes(x) \Leftrightarrow Smokes(y)\big)$

Suppose we have two constants: **Anna** (A) and **Bob** (B)

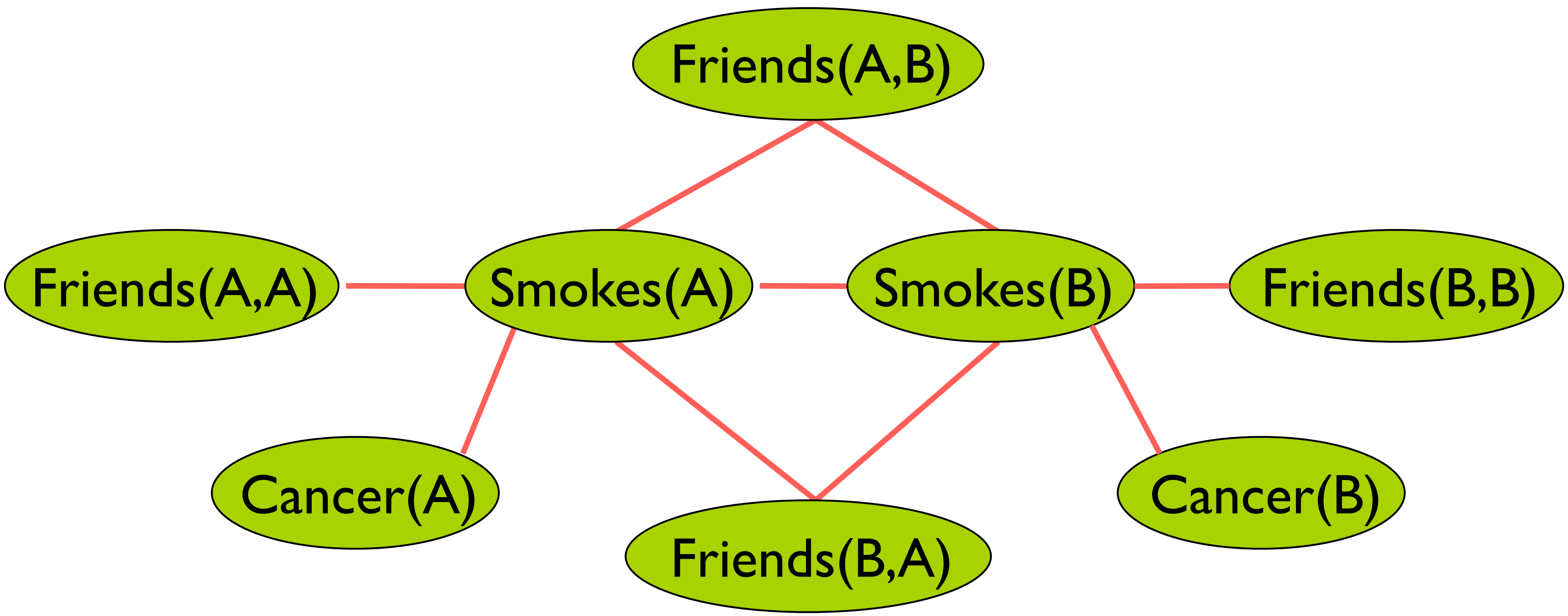


slides by Pedro Domingos

Markov Logic

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x, y) \Rightarrow \left(Smokes(x) \Leftrightarrow Smokes(y)\right)$

Suppose we have two constants: **Anna** (A) and **Bob** (B)



slides by Pedro Domingos

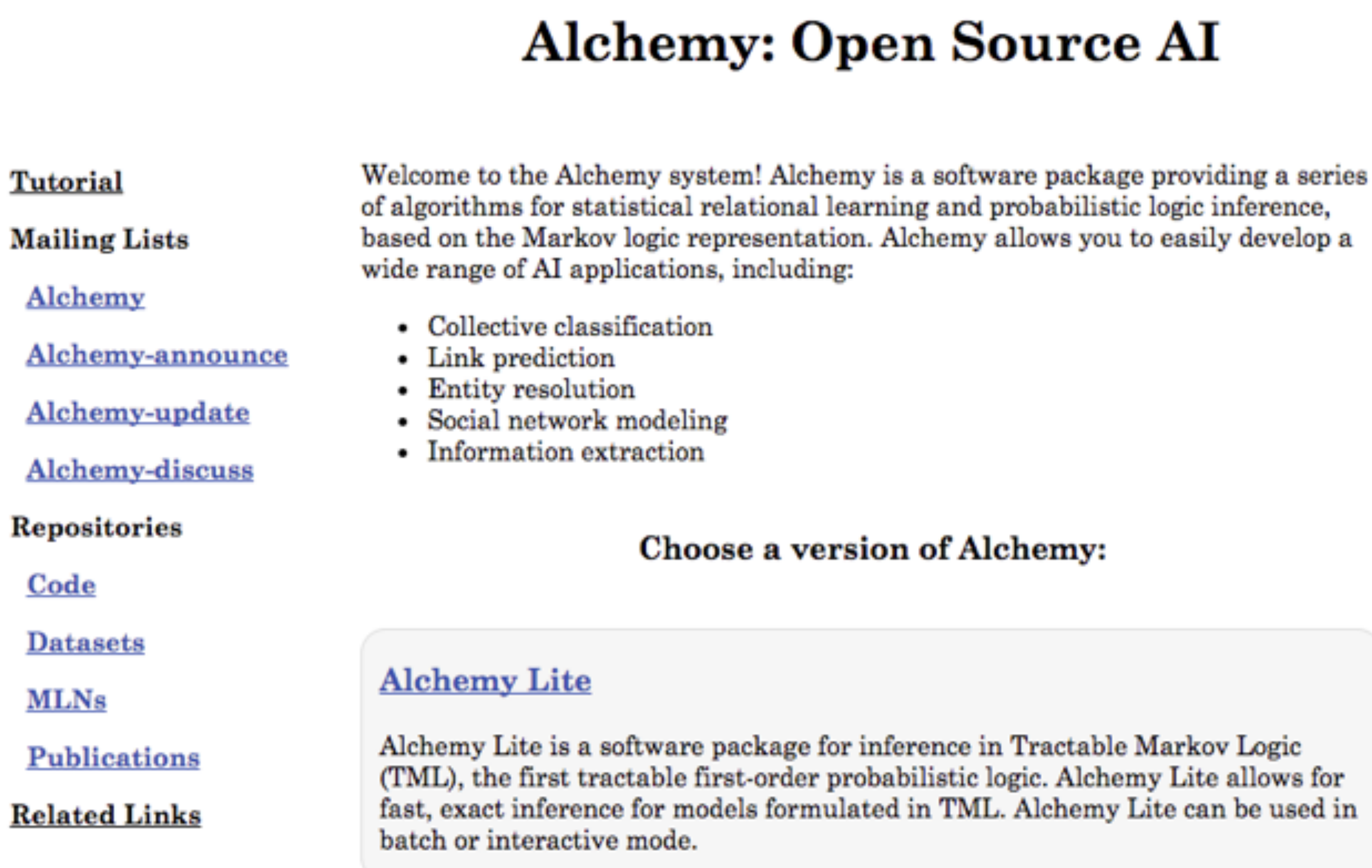
Markov Logic

- A Markov Logic Network (MLN) is a set of pairs (F, w) where
  - F is a formula in first-order logic
  - w is a real number
- An MLN defines a Markov network with
  - One node for each grounding of each predicate in the MLN
  - One feature for each grounding of each formula F in the MLN, with the corresponding weight w
- Probability of a world

$$P(x) = \frac{1}{Z} \exp\left(\sum_i \underbrace{w_i}_{\text{Weight of formula } i} \underbrace{n_i(x)}_{\text{No. of true groundings of formula } i \text{ in } x} \right)$$

Applications

- Natural language processing, Collective Classification, Social Networks, Activity Recognition, …



Alchemy: Open Source AI

Tutorial

Mailing Lists

- Alchemy
- Alchemy-announce
- Alchemy-update
- Alchemy-discuss

Repositories

- Code
- Datasets
- MLNs
- Publications

Related Links

Welcome to the Alchemy system! Alchemy is a software package providing a series of algorithms for statistical relational learning and probabilistic logic inference, based on the Markov logic representation. Alchemy allows you to easily develop a wide range of AI applications, including:

- Collective classification
- Link prediction
- Entity resolution
- Social network modeling
- Information extraction

Choose a version of Alchemy:

Alchemy Lite

Alchemy Lite is a software package for inference in Tractable Markov Logic (TML), the first tractable first-order probabilistic logic. Alchemy Lite allows for fast, exact inference for models formulated in TML. Alchemy Lite can be used in batch or interactive mode.

Part II : Inference

Inference

The challenge : disjoint sum problem

```
0.4::heads(1).
0.7::heads(2).
0.5::heads(3).
win :- heads(1).
win :- heads(2), heads(3).
```

$win \leftrightarrow h(1) \vee (h(2) \wedge h(3))$

$$P(win) = P(h(1) \vee (h(2) \wedge h(3))$$

$$=/= P(h(1)) + P(h(2) \wedge h(3))$$

should be

$$= P(h(1)) + P(h(2) \wedge h(3)) - P(h(1) \wedge h(2) \wedge h(3))$$

Inference

Map to Weighted Model Counting Problem and Solver

```
0.4::heads(1).
0.7::heads(2).
0.5::heads(3).
win :- heads(1).
win :- heads(2), heads(3).
```

$$win \leftrightarrow h(1) \vee (h(2) \wedge h(3))$$

Ground out

+ Put formula in CNF format

$$(\neg win \vee h(1) \vee h(2)) \wedge (\neg win \vee h(1) \vee h(3)) \wedge (win \vee \neg h(1)) \wedge (win \vee \neg h(2) \vee \neg h(3))$$

+ weights

+ call WMC

$h(1) \rightarrow 0.4$ $\quad h(2) \rightarrow 0.7$ $\quad h(3) \rightarrow 0.5$

$\neg h(1) \rightarrow 0.6$ $\quad \neg h(2) \rightarrow 0.3$ $\quad \neg h(3) \rightarrow 0.5$

Weighte

$$P(Q) = \sum_{F \cup R \models Q} \prod_{f \in F} p(f) \prod_{f \notin F} 1 - p(f)$$

propositional formula in conjunctive normal form (CNF)

given by SRL model & query

$$WMC(\phi) = \sum_{I_V \models \phi} \prod_{l \in I_V} w(l)$$

weight of literal

for p::f,
w(f) = p
w(not f) = 1−p

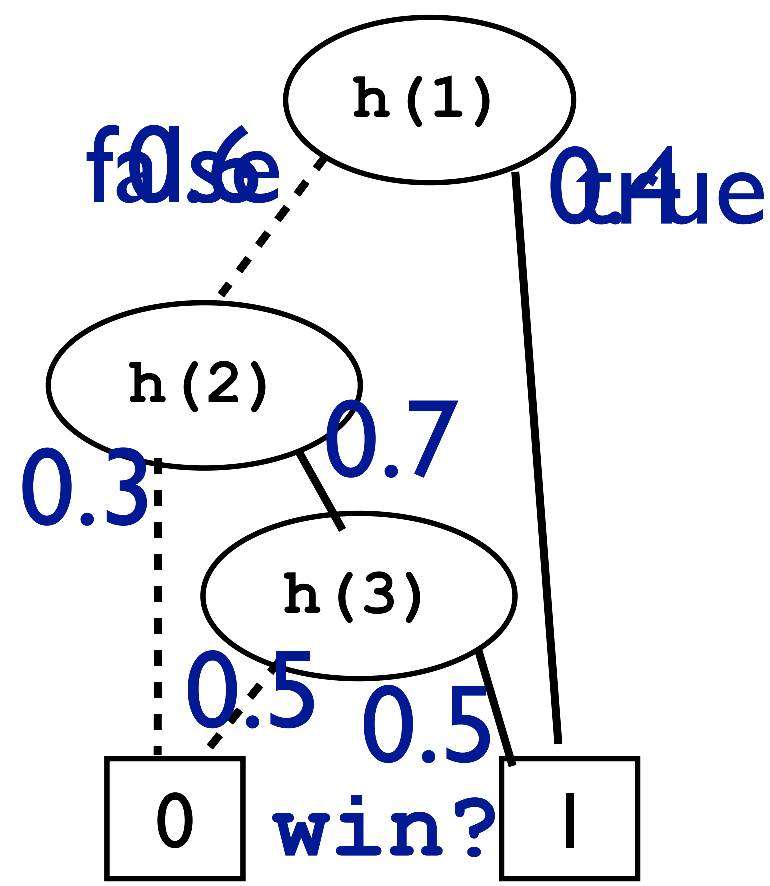
interpretations (truth value assignments) of propositional variables

possible worlds

Weighted Model Counting

- Simple WMC solvers based on a generalisation of DPLL algorithm for SAT (Davis Putnam Logeman Loveland algorithm)
- Current solvers often use knowledge compilation (is also state of the art for inference in graphical models) — here an OBDD, many variations s-dDNNF, SDDs, …

P(**win**) = probability of reaching 1-leaf



win ↔ h(1) ∨ (h(2) ∧ h(3))

More inference

- Many variations / extensions
- Approximate inference
- Lifted inference
  - infected(X) :- contact(X,Y), sick(Y).

Part III : Learning
a. Parameters

Parameter Learning

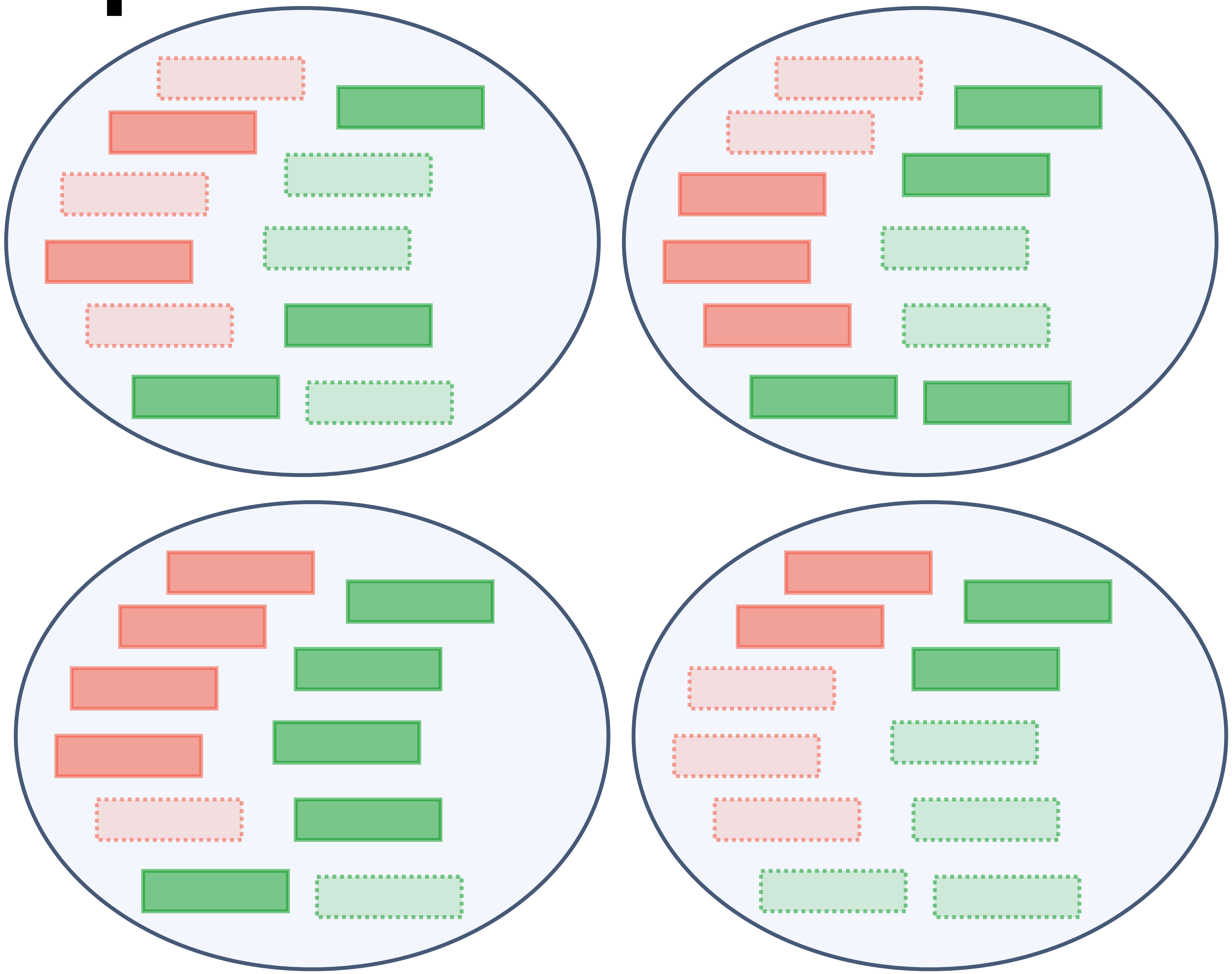
e.g., webpage classification model

for each *CLASS1*, *CLASS2* and each *WORD*

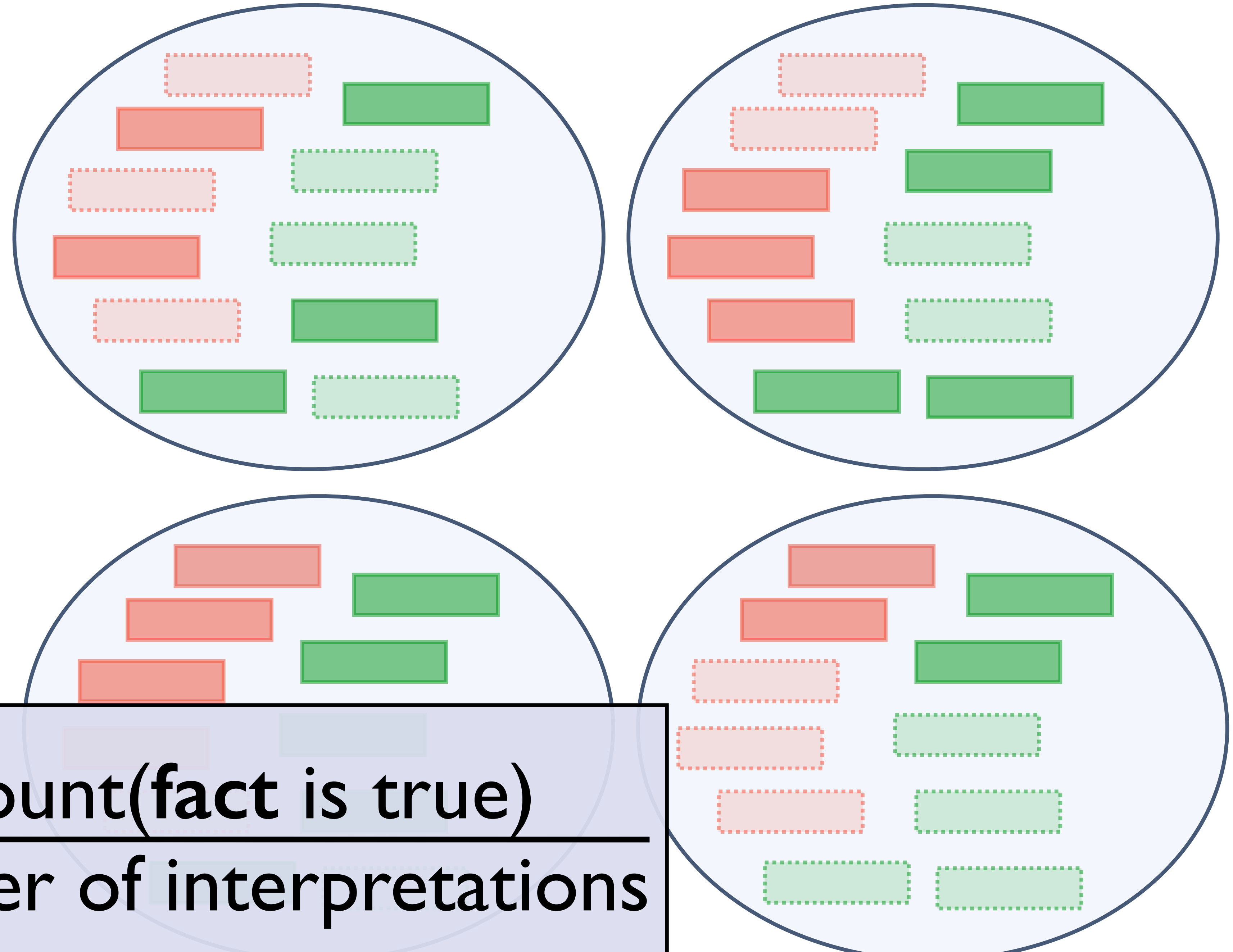
?? :: link_class(Source,Target,*CLASS1*,*CLASS2*).
?? :: word_class(*WORD*,*CLASS*).

```
class(Page,C)    :-   has_word(Page,W), word_class(W,C).

class(Page,C)          :-   links_to(OtherPage,Page),
    class(OtherPage,OtherClass),
                              link_class(OtherPage,Page,OtherClass,C).
```

Sampling Interpretations

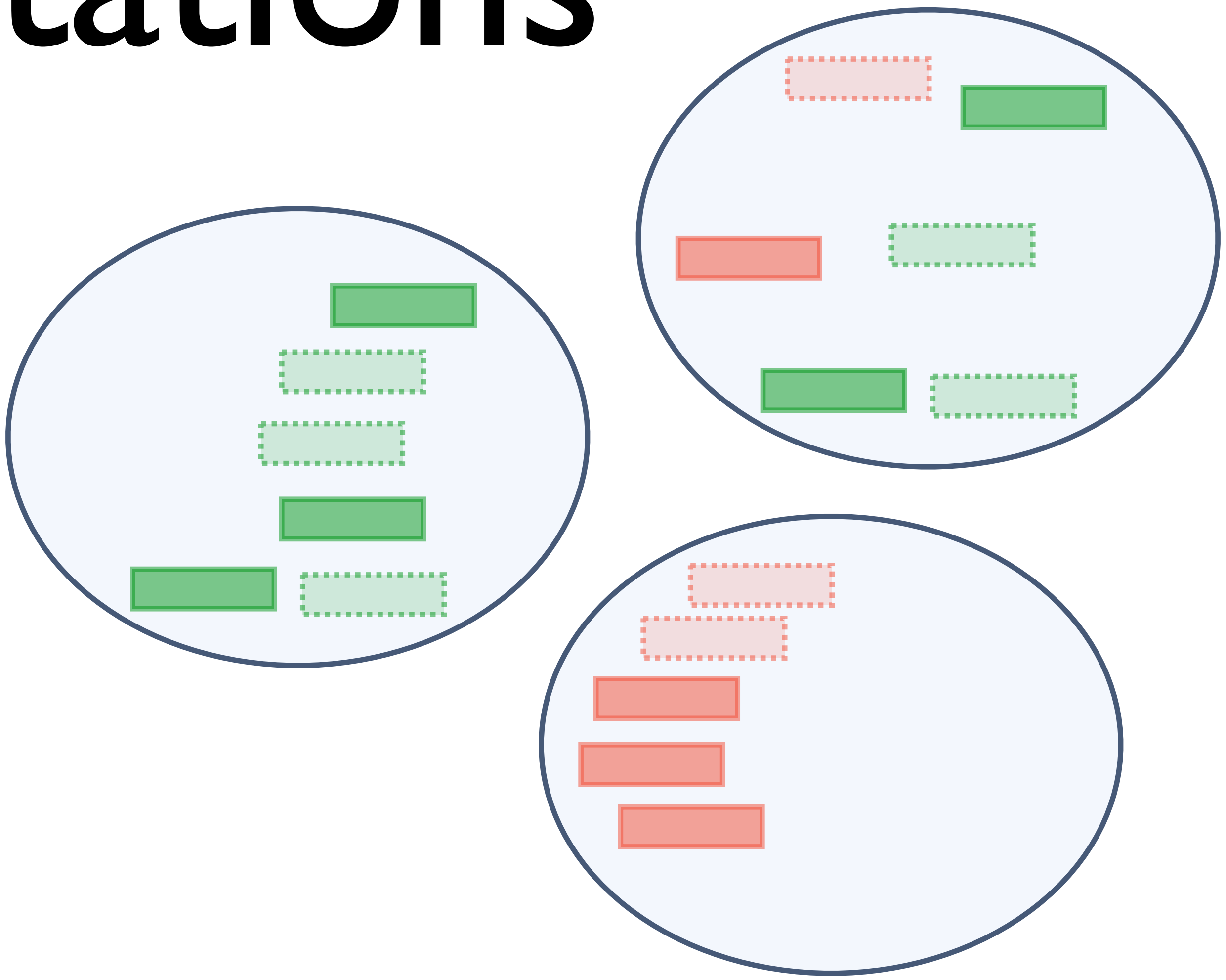


Parameter Estimation



$$p(\textbf{fact}) = \frac{\text{count}(\textbf{fact} \text{ is true})}{\text{Number of interpretations}}$$

Learning from partial interpretations



- Not all facts observed
- Soft-EM
- use **expected count** instead of **count**
- **P(Q |E) -- conditional queries !**

 [Gutmann et al, ECML 11; Fierens et al, TPLP 14]

Part III : Learning
b. Rules / Structure

Information Extraction in NELL

Recently-Learned Facts

twitter

Refresh

instance	iteration	date learned	confidence
kelly_andrews is a female	826	29-mar-2014	98.7
investment_next_year is an economic sector	829	10-apr-2014	95.3
shibenik is a geopolitical entity that is an organization	829	10-apr-2014	97.2
quality_web_design_work is a character trait	826	29-mar-2014	91.0
mercedes_benz_cls_by_carlsson is an automobile manufacturer	829	10-apr-2014	95.2
social_work is an academic program at the university rutgers_university	827	02-apr-2014	93.8
dante wrote the book the_divine_comedy	826	29-mar-2014	93.8
willie_aames was born in the city los_angeles	831	16-apr-2014	100.0
kitt_peak is a mountain in the state or province arizona	831	16-apr-2014	96.9
greenwich is a park in the city london	831	16-apr-2014	100.0

instances for many different relations

degree of certainty

NELL: http://rtw.ml.cmu.edu/rtw/

ProbFOIL

- Upgrade rule-learning to a probabilistic setting within a relational learning / inductive logic programming setting
  - Works with a probabilistic logic program instead of a deterministic one.
- Introduce ProbFOIL, an adaption of Quinlan's FOIL to this setting.
- Apply to probabilistic databases like NELL

Pro Log

surfing(X) :- not pop(X) and windok(X).

H

surfing(X) :- not pop(X) and sunshine(X).

pop(e1).   windok(e1).   sunshine(e1).   **B**

?-surfing(e1). **e**
no

B U H |=\= e   (H does not cover e)

An ILP example

ProbLog

a probabilistic Prolog

p1:: surfing(X) :- not pop(X) and windok(X).

H

p2:: surfing(X) :- not pop(X) and sunshine(X).

0.2::pop(e1).   0.7::windok(e1).   0.6::sunshine(e1).

B

?-P(surfing(e1)). **e**

gives (1-0.2) x 0.7 x p1 + (1-0.2) x 0.6 x (1-0.7) x p2 = P(B U H |= e)

not pop   x windok x p1 + not pop x sunshine x (not windok) x p1

probability that the example is covered

Inductive Probabilistic Logic Programming

Given

a set of example facts $e \in E$ together with the probability p that they hold

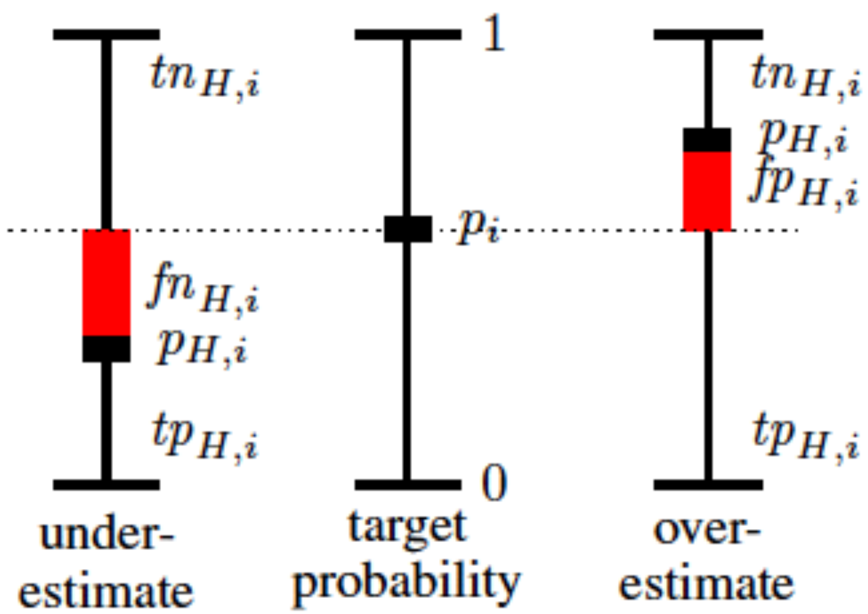
a background theory B in ProbLog

a hypothesis space L (a set of clauses)

Find

$$\arg\min_H loss(H, B, E) = \arg\min_H \sum_{e_i \in E} |P_s(B \cup H \models e) - p_i|$$

Adapt Rule-learner



Contingency table: not only 1 / 0 values

Covering: use multiple rules to cover an example

Algorithm 1 The ProbFOIL$^+$ learning algorithm.

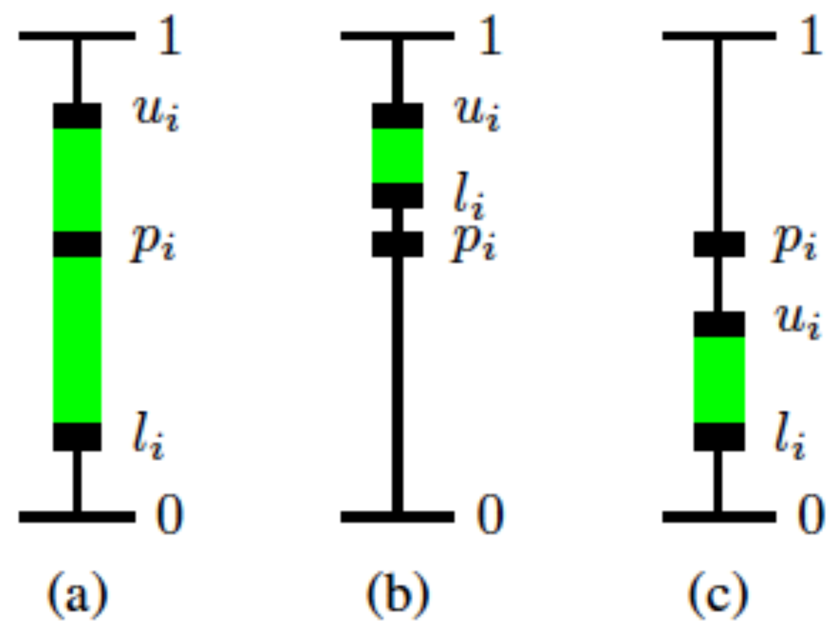
```
function PROBFOIL+(target)
  H := ∅
  while true do
    clause := LEARNRULE(H, target)
    if GSCORE(H) < GSCORE(H ∪ {clause}) then
      H := H ∪ {clause}
    else return H
function LEARNRULE(H, target)
  candidates := {x :: target ← true}
  best := (x :: target ← true)
  while candidates ≠ ∅ do
    next_cand := ∅
    for all x :: target ← body ∈ candidates do
      for all refinement ∈ ρ(target ← body) do
        if not REJECT(H, best, x :: target ← body) then
          next_cand := next_cand ∪ {x :: target ← body∧
                                        refinement}
          if LSCORE (H, x :: target ← body ∧ refinement) >
                                        LSCORE(H, best) then
            best := (x :: target ← body ∧ refinement)
    candidates := next_cand
  return best
```

Technical Novelty

p:: surfing(X) :- not pop(X) and windok(X).

$u_i = (p{=}1)$

$l_i = (p{=}0)$



ProbFOIL includes

a method to determine “optimal” p for a given rule

Experiments

Table 4: Precision for different experimental setups and parameters (A: $m = 1, p = 0.99$, B: $m = 1000, p = 0.90$).

Setting	**athleteplaysforteam**		**athleteplayssport**		**teamplaysinleague**		**athleteplaysinleague**		**teamplaysagainstteam**	
train/test/rule	**A**	**B**	**A**	**B**	**A**	**B**	**A**	**B**	**A**	**B**
1: det/det/det	74.00	69.36	94.14	93.47	96.29	82.15	80.95	74.14	73.40	73.86
2: det/prob/det	73.51	69.57	97.53	94.85	96.70	87.83	90.83	77.73	73.70	73.35
3: det/prob/prob	74.67	69.82	95.86	94.74	96.35	82.57	82.26	75.29	73.84	74.34
4: det/prob/prob	77.25	73.87	96.53	96.04	98.00	90.59	84.91	79.36	77.26	77.83
5: det/prob/prob	74.76	69.97	95.85	94.69	96.44	82.51	81.99	75.07	73.90	74.16
6: prob/prob/det	75.83	73.11	93.40	93.76	94.44	93.67	79.41	79.42	80.87	80.60
7: prob/prob/prob	78.31	73.72	95.62	95.10	98.84	91.86	96.94	79.49	85.78	81.81

Table 3: Learned relational rules for the different predicates (fold 1).

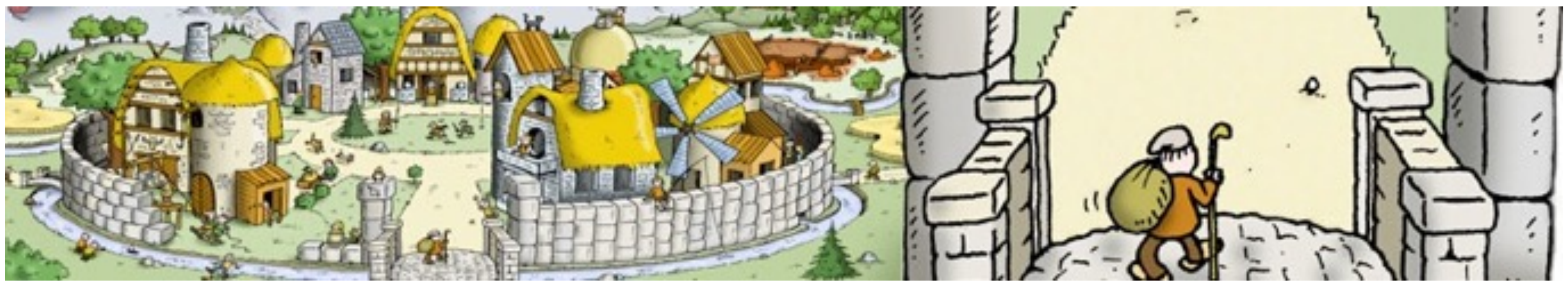
0.9375::athleteplaysforteam(A,B)	←	athleteledsportsteam(A,B).
0.9675::athleteplaysforteam(A,B)	←	athleteledsportsteam(A,V1), teamplaysagainstteam(B,V1).
0.9375::athleteplaysforteam(A,B)	←	athleteplayssport(A,V1), teamplayssport(B,V1).
0.5109::athleteplaysforteam(A,B)	←	athleteplaysinleague(A,V1), teamplaysinleague(B,V1).
0.9070::athleteplayssport(A,B)	←	athleteledsportsteam(A,V2), teamalsoknownas(V2,V1), teamplayssport(V1,B), teamplayssport(V2,B).
0.9070::athleteplayssport(A,B)	←	athleteplaysforteam(A,V2), teamalsoknownas(V2,V1), teamplayssport(V1,B), teamplayssport(V2,B),teamalsoknownas(V1,V2).
0.9070::athleteplayssport(A,B)	←	athleteplaysforteam(A,V1), teamplayssport(V1,B).
0.9286::athleteplaysinleague(A,B)	←	athleteledsportsteam(A,V1), teamplaysinleague(V1,B).
0.7868::athleteplaysinleague(A,B)	←	athleteplaysforteam(A,V2), teamalsoknownas(V2,V1), teamplaysinleague(V1,B).
0.9384::athleteplaysinleague(A,B)	←	athleteplayssport(A,V2), athleteplayssport(V1,V2), teamplaysinleague(V1,B).
0.9024::athleteplaysinleague(A,B)	←	athleteplaysforteam(A,V1), teamplaysinleague(V1,B).

ProbFOIL

- Upgrade rule-learning to a probabilistic setting within a relational learning / inductive logic programming setting
  - Works with a probabilistic logic program instead of a deterministic one.
- Introduce ProbFOIL, an adaption of Quinlan's FOIL to this setting.
- Apply to probabilistic databases like NELL

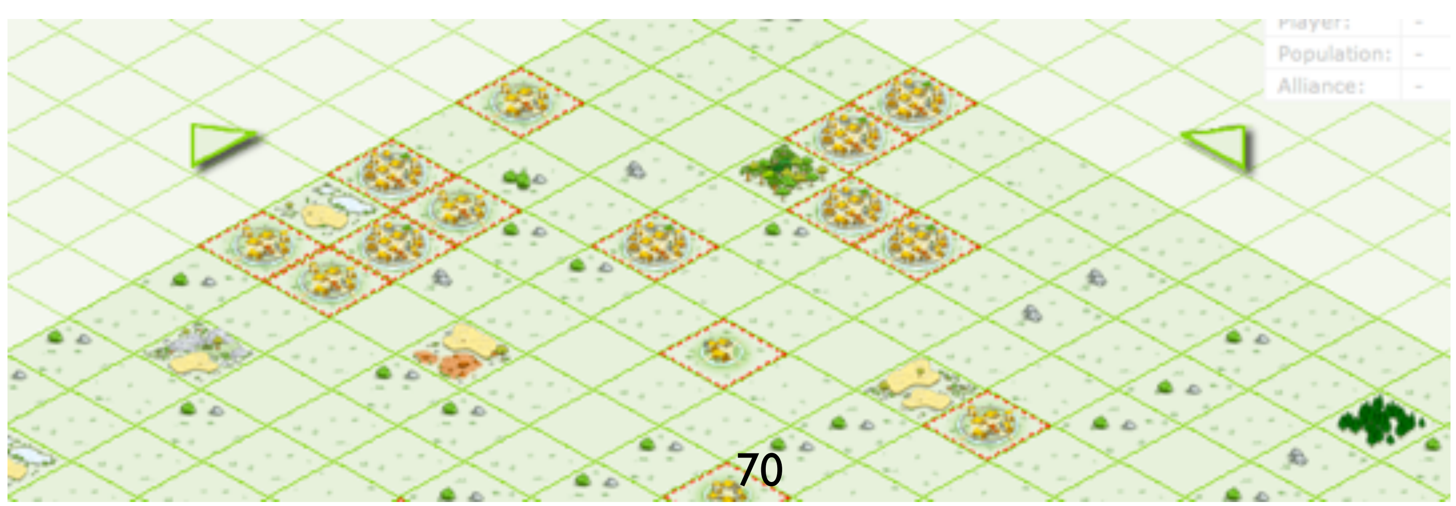
Part IV : Dynamics

Dynamics: Evolving Networks



- *Travian*: A massively multiplayer real-time strategy game
  - Commercial game run by TravianGames GmbH
  - ~3.000.000 players spread over different “worlds”
  - ~25.000 players in one world

[Thon et al. ECML 08]



World Dynamics

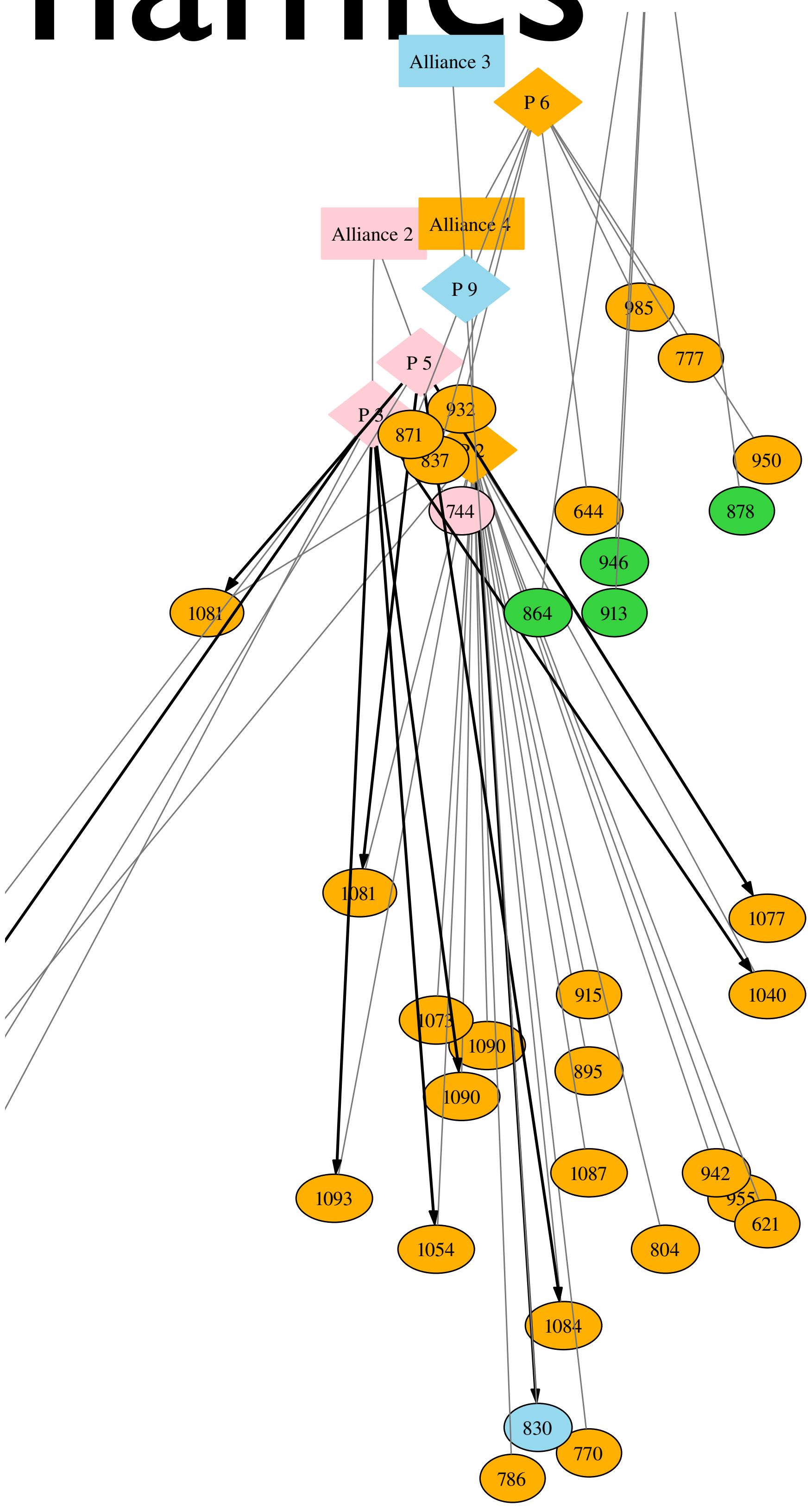
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world ?
Can we use it for playing better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

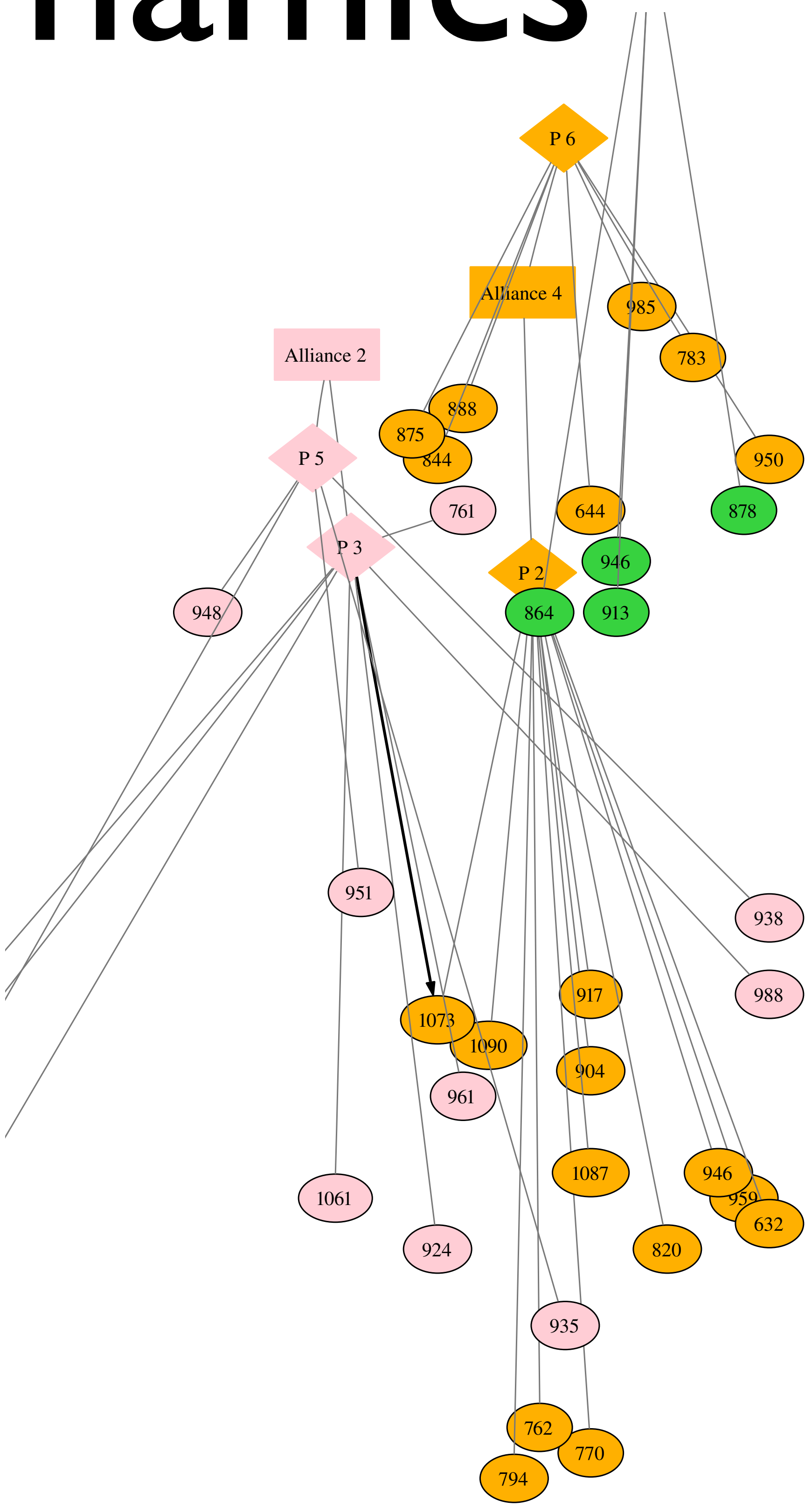
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

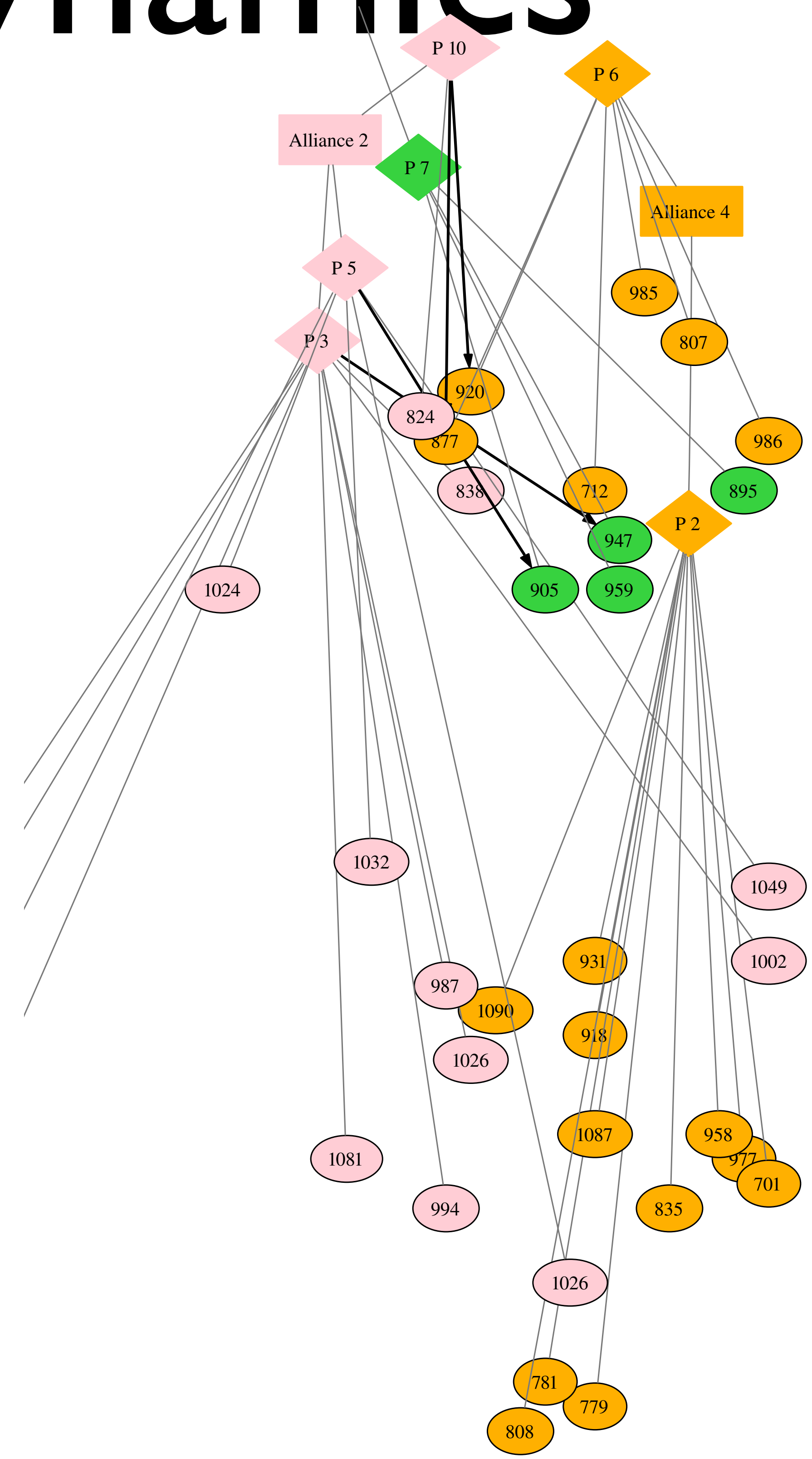
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

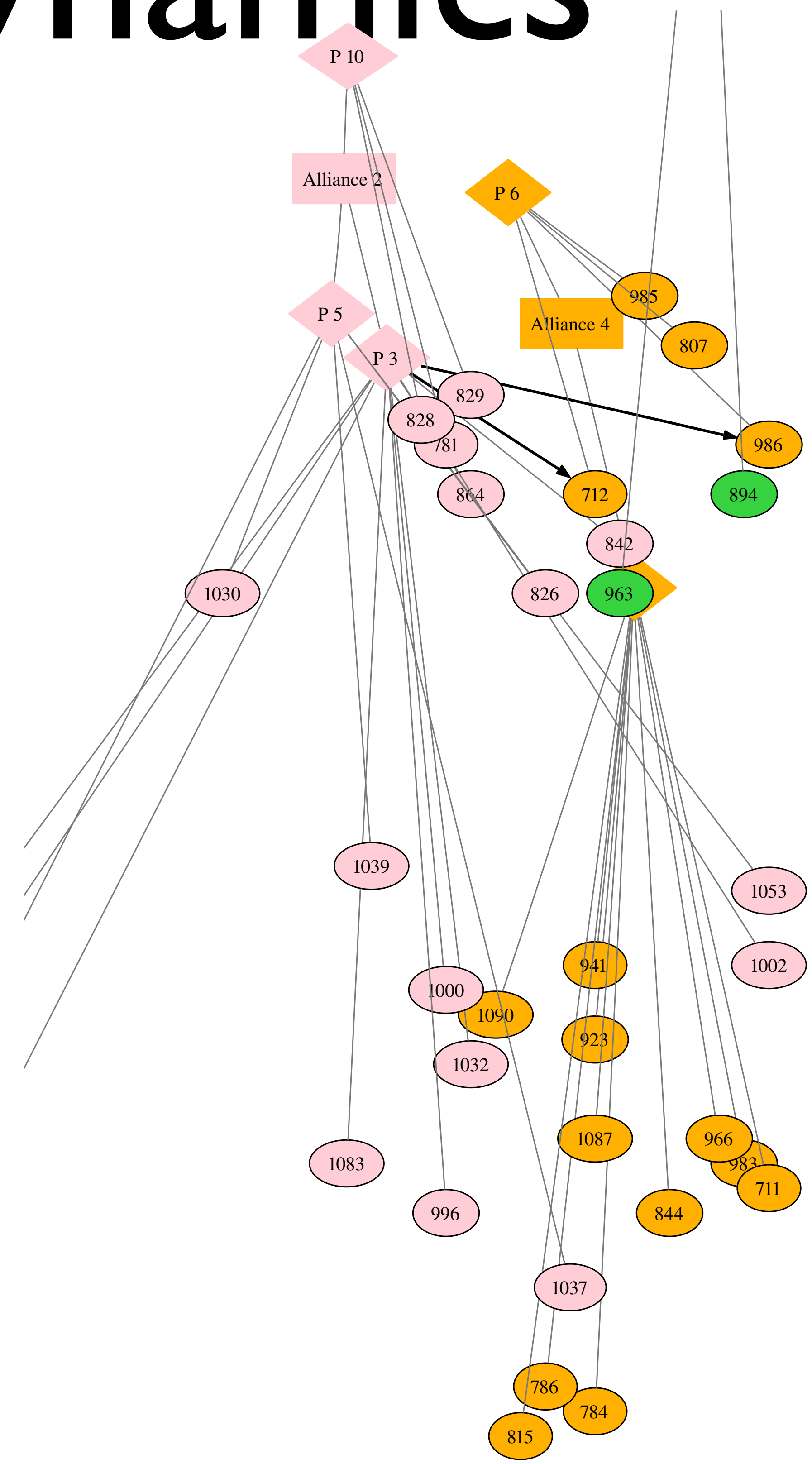
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

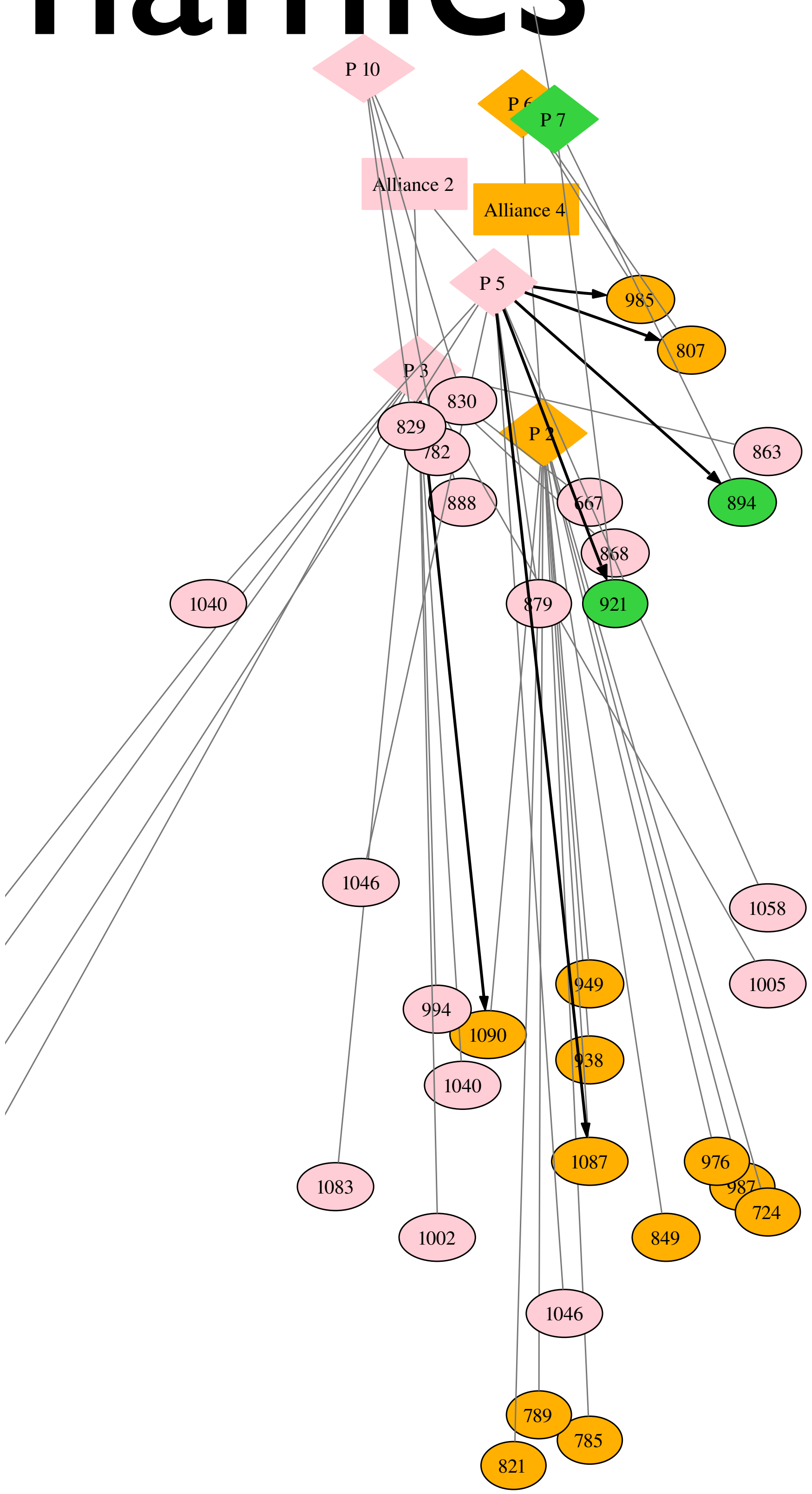
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

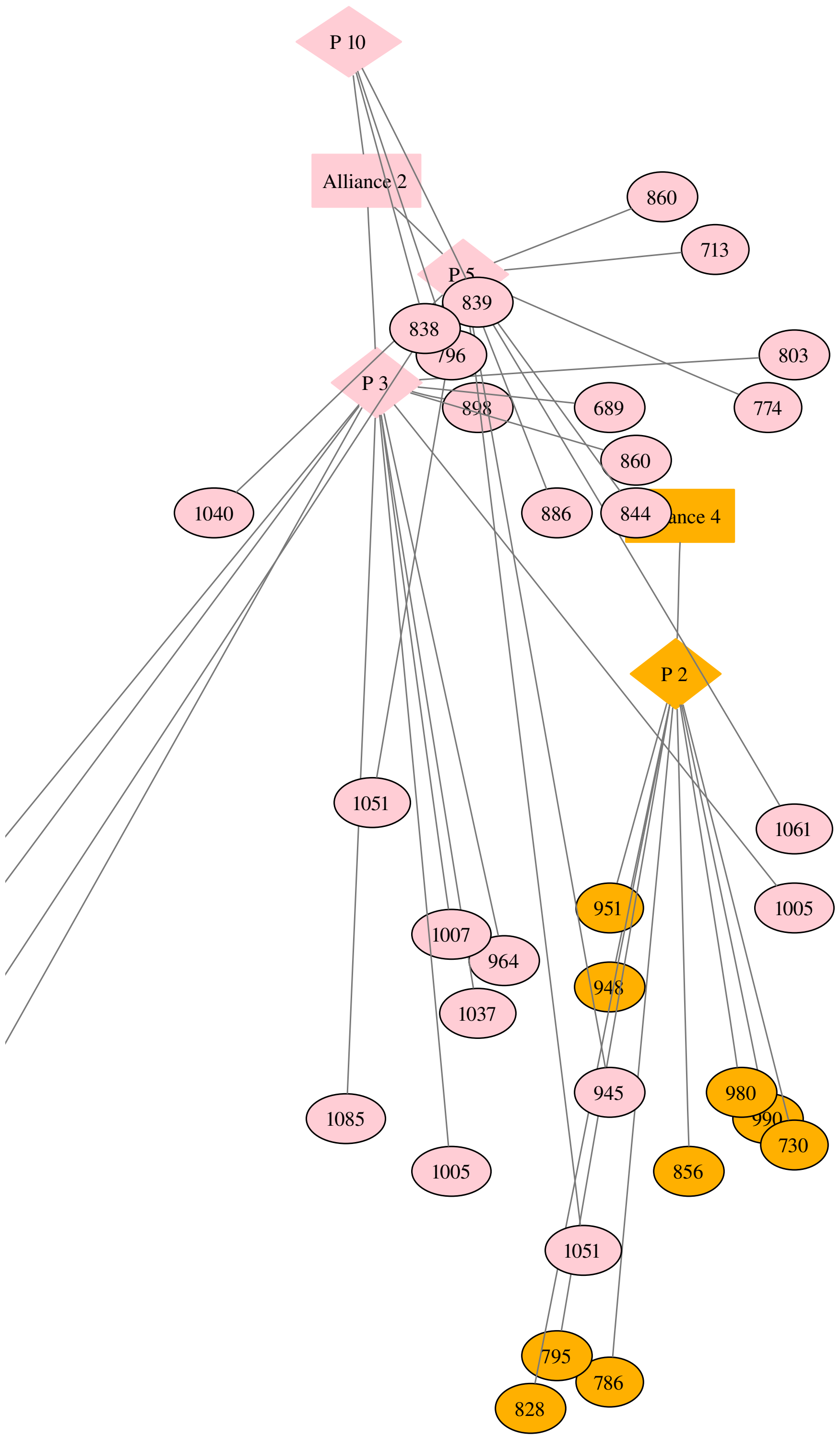
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world ?
Can we use it for playing better ?

[Thon, Landwehr, De Raedt, ECML08]

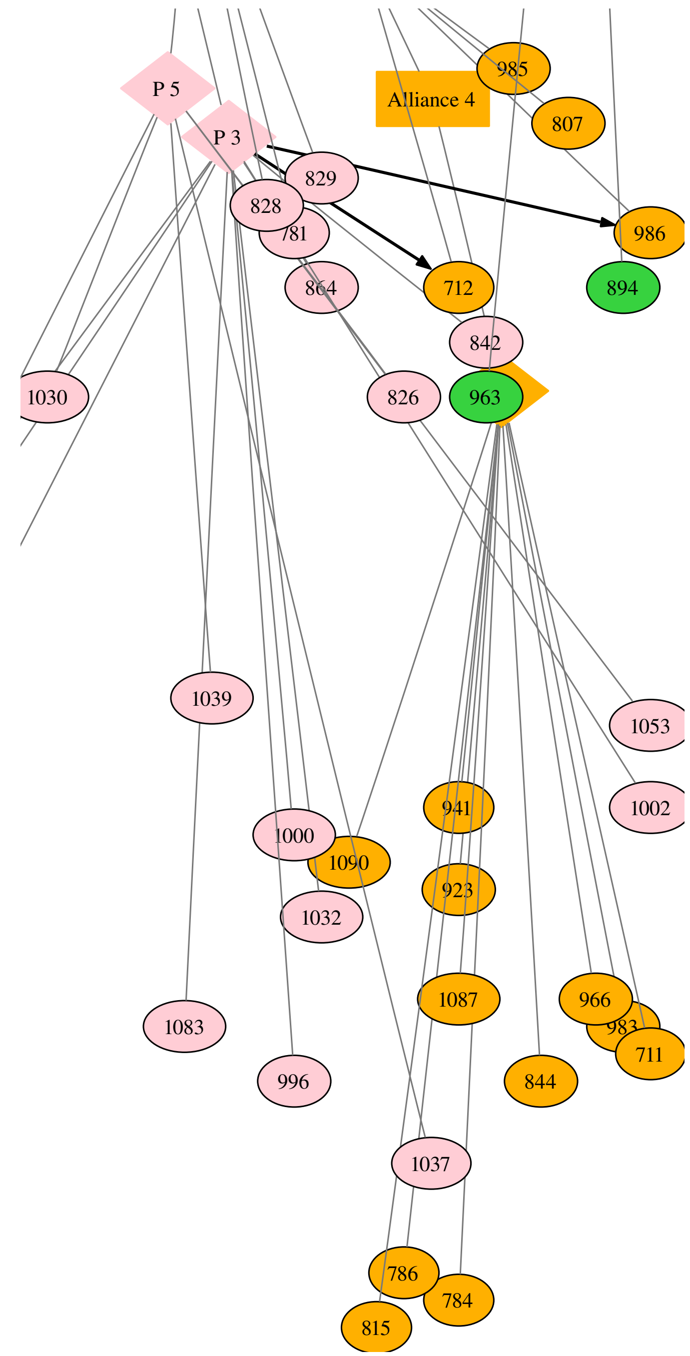


CPT-Rules

$$\underbrace{b_1, \ldots b_n}_{\text{cause}} \rightarrow \underbrace{h_1 : p_1 \vee \ldots \vee h_m : p_m}_{\text{effect}}$$

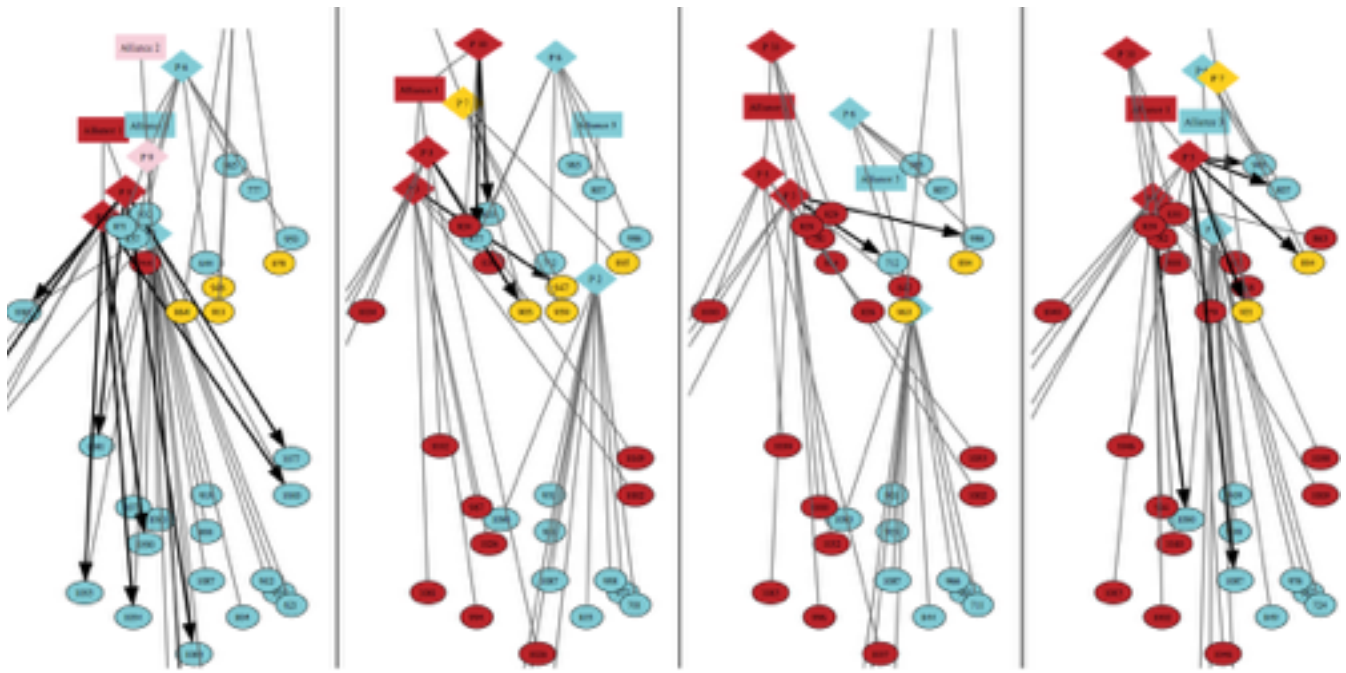
$$city(C, Owner), city(C2, Attacker), close(C, C2) \rightarrow$$
$$conquest(Attacker, C2) \ : \ p \ \vee \ nil \ : \ (1-p)$$

conquer a city which is close
P(conquest(),Time+5) ?
learn parameters



Thon et al. MLJ 11

Causal Probabilistic Time-Logic (CPT-L)



how does the world change over time?

one of the **effects** holds at time T+1

```
0.4::conquest(Attacker,C); 0.6::nil :-
        city(C,Owner),city(C2,Attacker),close(C,C2).
```

if **cause** holds at time T

[Thon et al, MLJ 11]

Distributional Clauses (DC)

- Discrete- and continuous-valued random variables

random variable with Gaussian distribution

```
length(Obj) ~ gaussian(6.0,0.45) :- type(Obj,glass).
stackable(OBot,OTop) :-
      ≃length(OBot) ≥ ≃length(OTop),
      ≃width(OBot) ≥ ≃width(OTop).
ontype(Obj,plate) ~ finite([0 : glass, 0.0024 : cup,
                           0 : pitcher, 0.8676 : plate,
                           0.0284 : bowl, 0 : serving,
                           0.1016 : none])
                       :- obj(Obj), on(Obj,O2), type(O2,plate).
```

comparing values of random variables

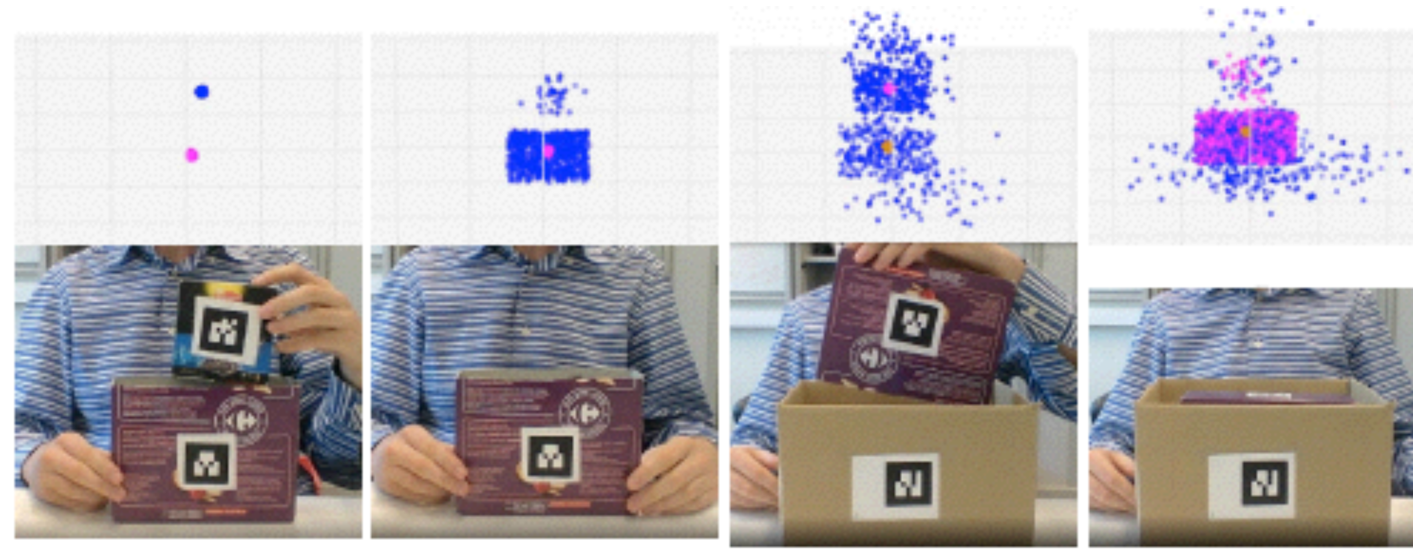
random variable with discrete distribution

[Gutmann et al, TPLP 11; Nitti et al, IROS 13]

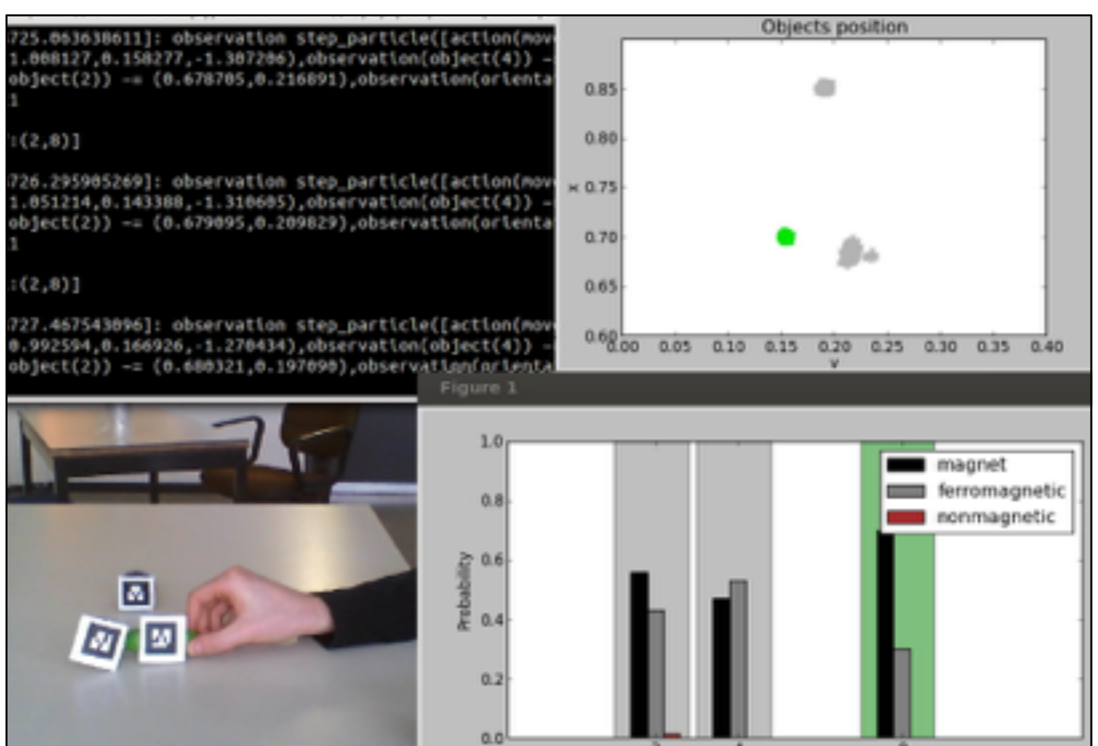
Relational State Estimation over Time

Magnetism scenario

- object tracking
- category estimation from interactions



(a) cube on the box (b) cube inside the box (c) rotated box on a beige box (d) cube and box inside the beige box



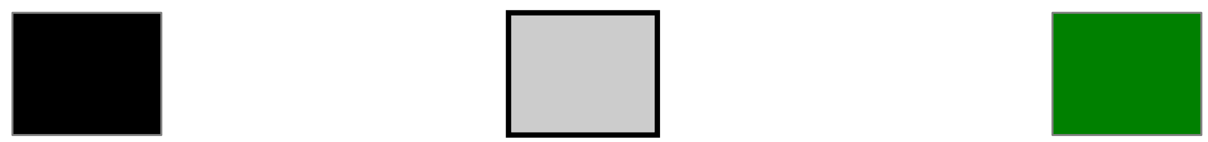
Box scenario

- object tracking even when invisible
- estimate spatial relations

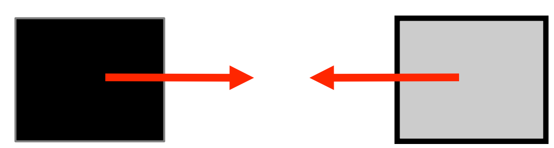
[Nitti et al, IROS 13]

Magnetic scenario

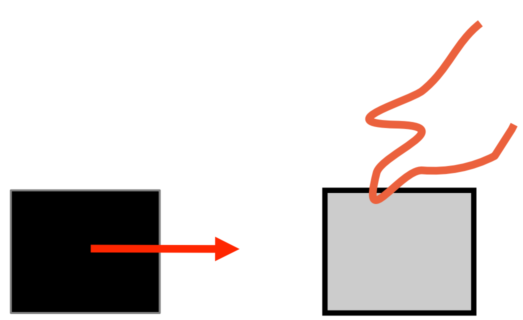
- 3 object types: magnetic, ferromagnetic, nonmagnetic

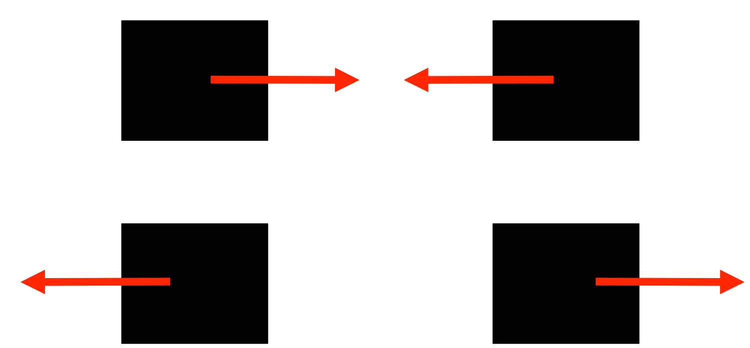


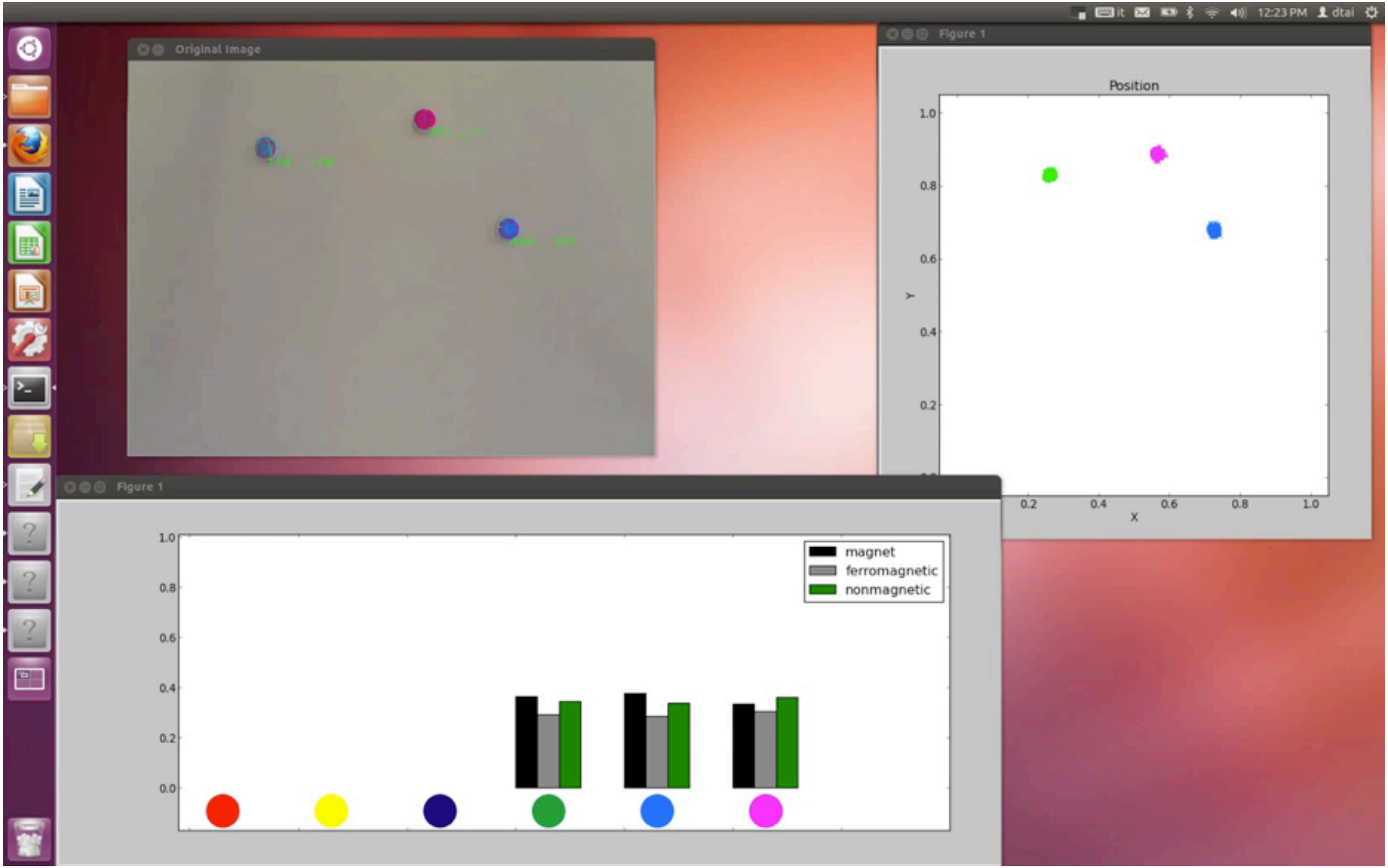
- Nonmagnetic objects do not interact
- A magnet and a ferromagnetic object attract each other



- Magnetic force that depends on the distance
- If an object is held magnetic force is compensated.






Original Image
Figure 1
Position
1.0
0.8
0.6
0.4
0.2
Y
0.2
0.4
0.6
0.8
1.0
X
Figure 1
magnet
ferromagnetic
nonmagnetic
1.0
0.8
0.6
0.4
0.2
0.0
12:23 PM
dtai

Magnetic scenario

- 3 object types: magnetic, ferromagnetic, nonmagnetic

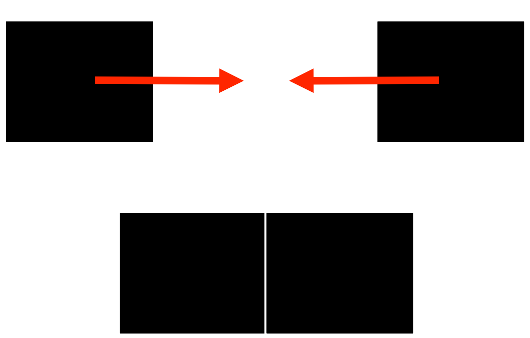
$\text{type}(X)_t \sim \text{finite}([1/3{:}\text{magnet}, 1/3{:}\text{ferromagnetic}, 1/3{:}\text{nonmagnetic}]) \leftarrow \text{object}(X).$

- 2 magnets attract or repulse

$\text{interaction}(A,B)_t \sim \text{finite}([0.5{:}\text{attraction}, 0.5{:}\text{repulsion}]) \leftarrow \text{object}(A), \text{object}(B), A<B, \text{type}(A)_t = \text{magnet}, \text{type}(B)_t = \text{magnet}.$

- Next position after attraction

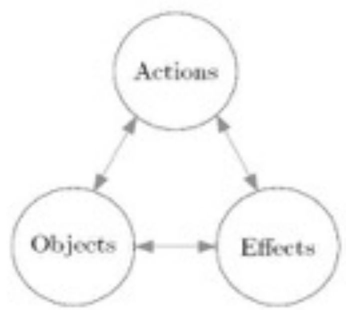
$\text{pos}(A)_{t+1} \sim \text{gaussian}(\text{middlepoint}(A,B)_t, \text{Cov}) \leftarrow \text{near}(A,B)_t, \text{not}(\text{held}(A)), \text{not}(\text{held}(B)), \text{interaction}(A,B)_t = \text{attr}, c/\text{dist}(A,B)_t^2 > \text{friction}(A)_t.$



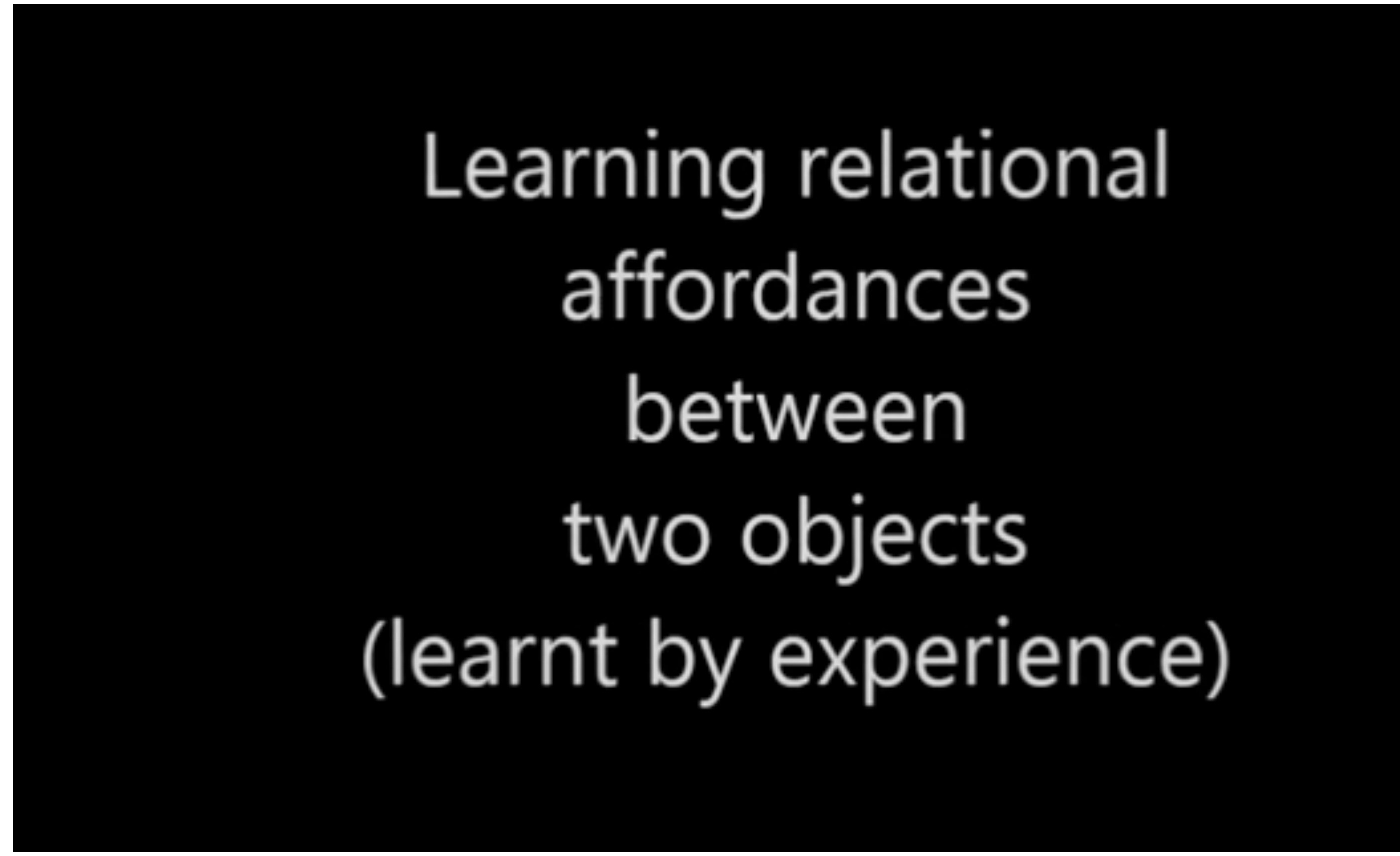
$\text{pos}(A)_{t+1} \sim \text{gaussian}(\text{pos}(A)_t, \text{Cov}) \leftarrow \text{not}(\ \text{attraction}(A,B)\ ).$

Learning relational affordances

Learn probabilistic model



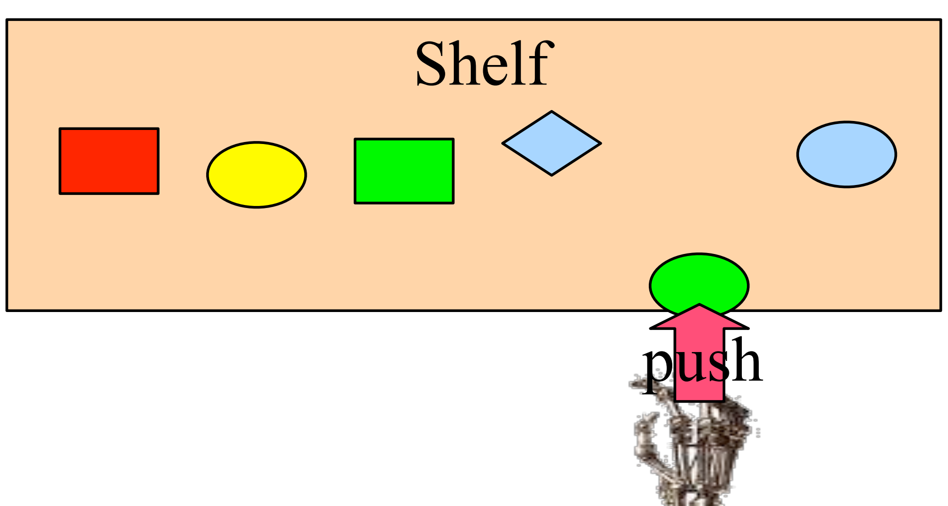
Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

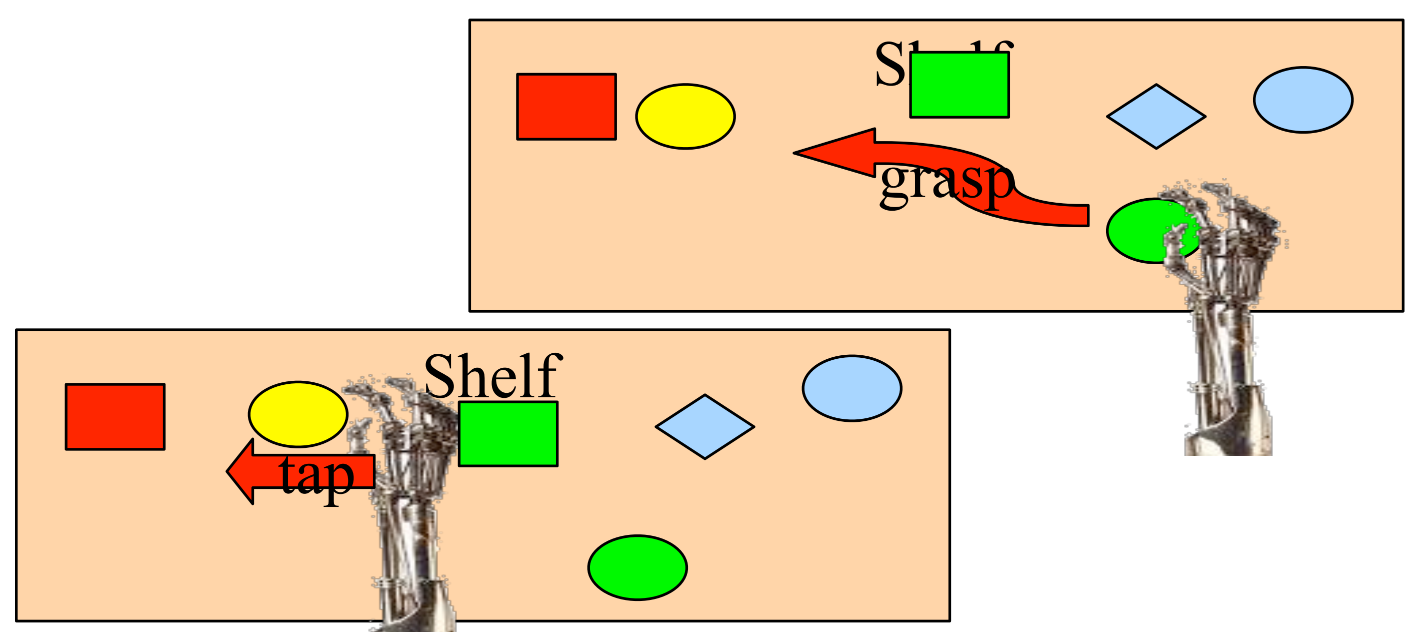


From two object interactions

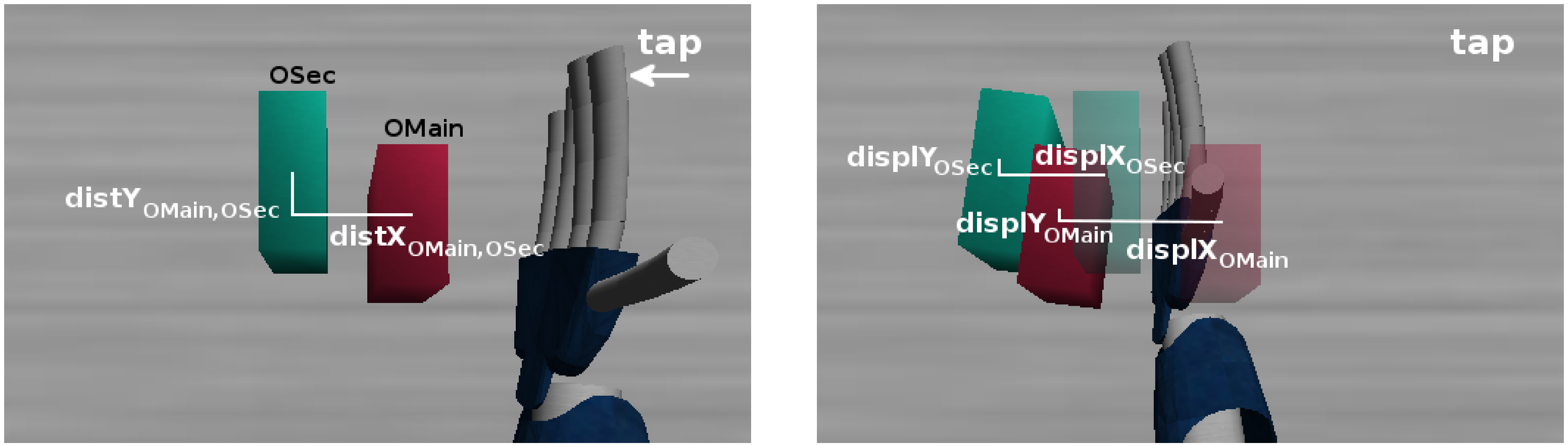
Generalize to N

Moldovan et al. ICRA 12, 13, 14
Nitti et al, MLJ 16, 17; ECAI 16





What is an affordance ?



Clip 8: Relational O before (l), and E after the action execution (r).

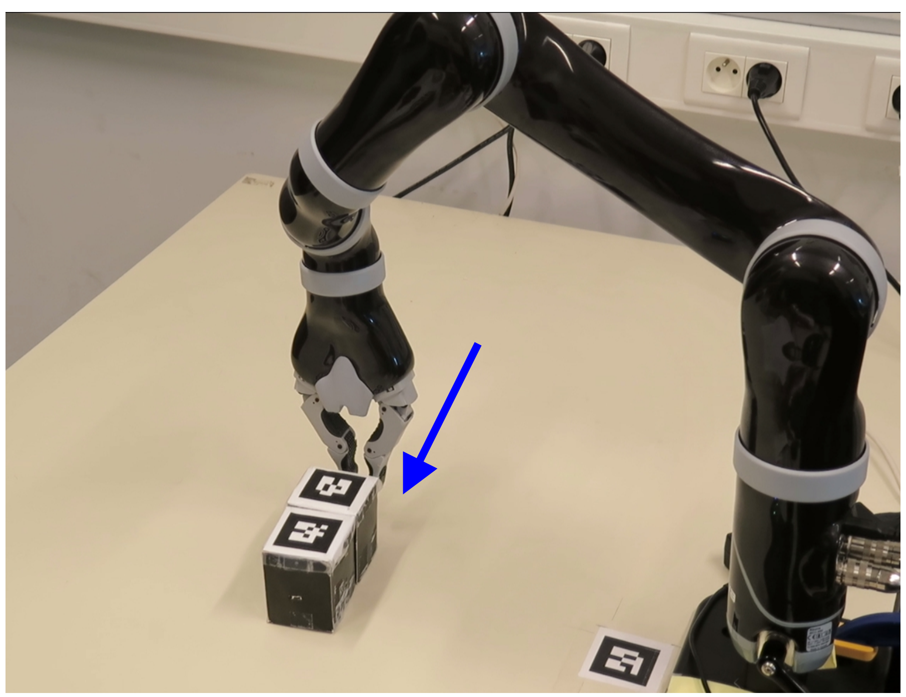
Table 1: Example collected O, A, E data for action in Figure 8

Object Properties	Action	Effects
$shape_{O_{Main}}$: $sprism$ $shape_{O_{Sec}}$: $sprism$ $distX_{O_{Main},O_{Sec}}$: $6.94cm$ $distY_{O_{Main},O_{Sec}}$: $1.90cm$	$tap(10)$	$displX_{O_{Main}}$: $10.33cm$ $displY_{O_{Main}}$: $-0.68cm$ $displX_{O_{Sec}}$: $7.43cm$ $displY_{O_{Sec}}$: $-1.31cm$

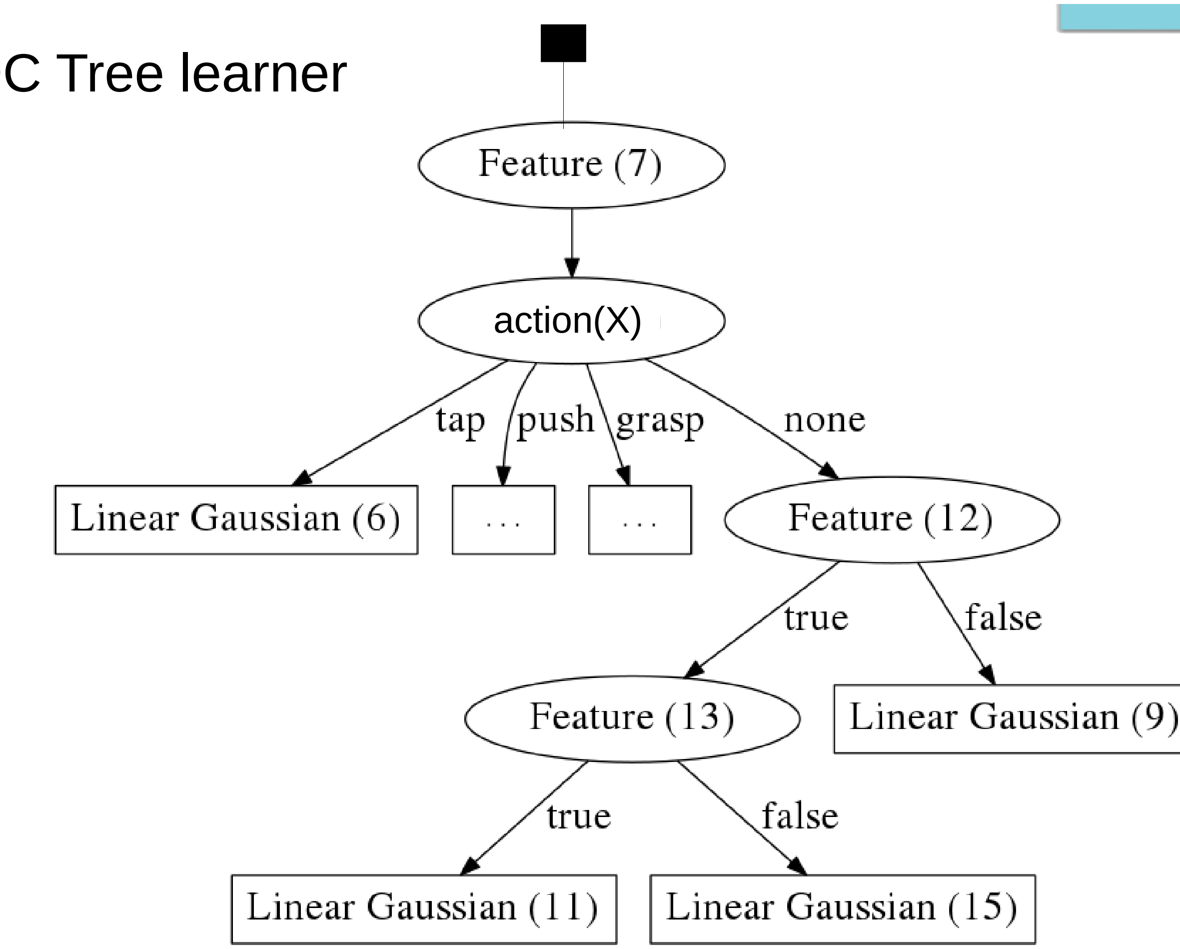
- Formalism — related to STRIPS but models delta
  - but also joint probability model over A, E, O

Relational Affordance Learning

- **Learning the Structure of Dynamic Hybrid Relational Models**
  Nitti, Ravkic, et al. ECAI 2016
  - Captures relations/affordances
  - Suited to learn affordances in robotics set-up, continuous and discrete variables
  - Planning in hybrid robotics domain

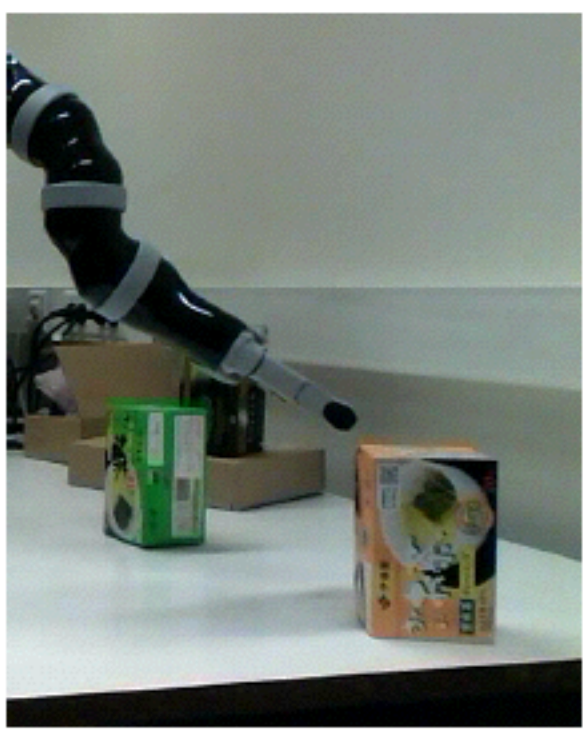


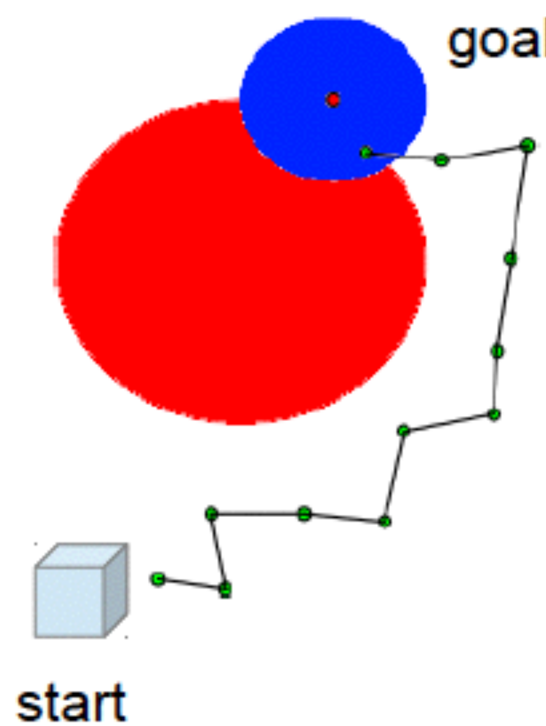
DDC Tree learner

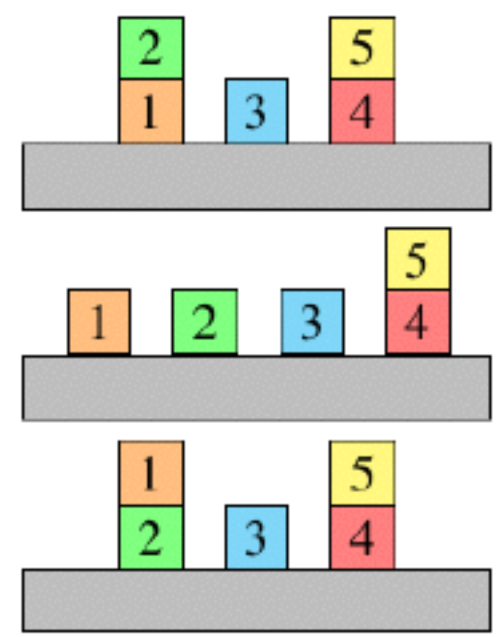


Planning

- Main task: probabilistic planning
  Find the best action to achieve the goal
- Discrete + continuous + relational representation



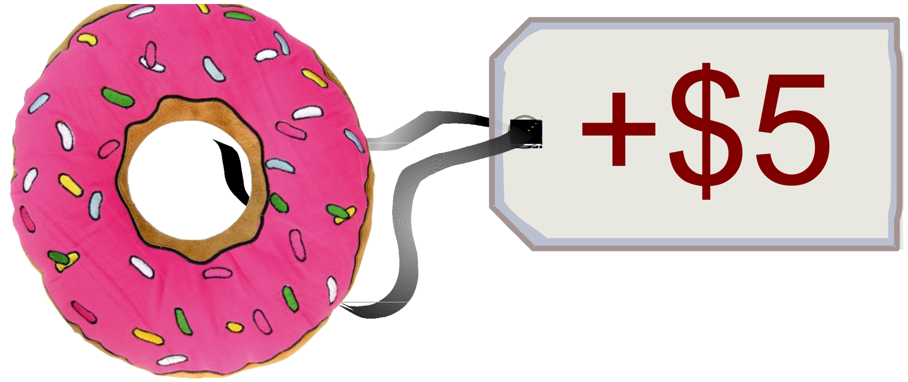




[Nitti et al ECML 15, MLJ 17]

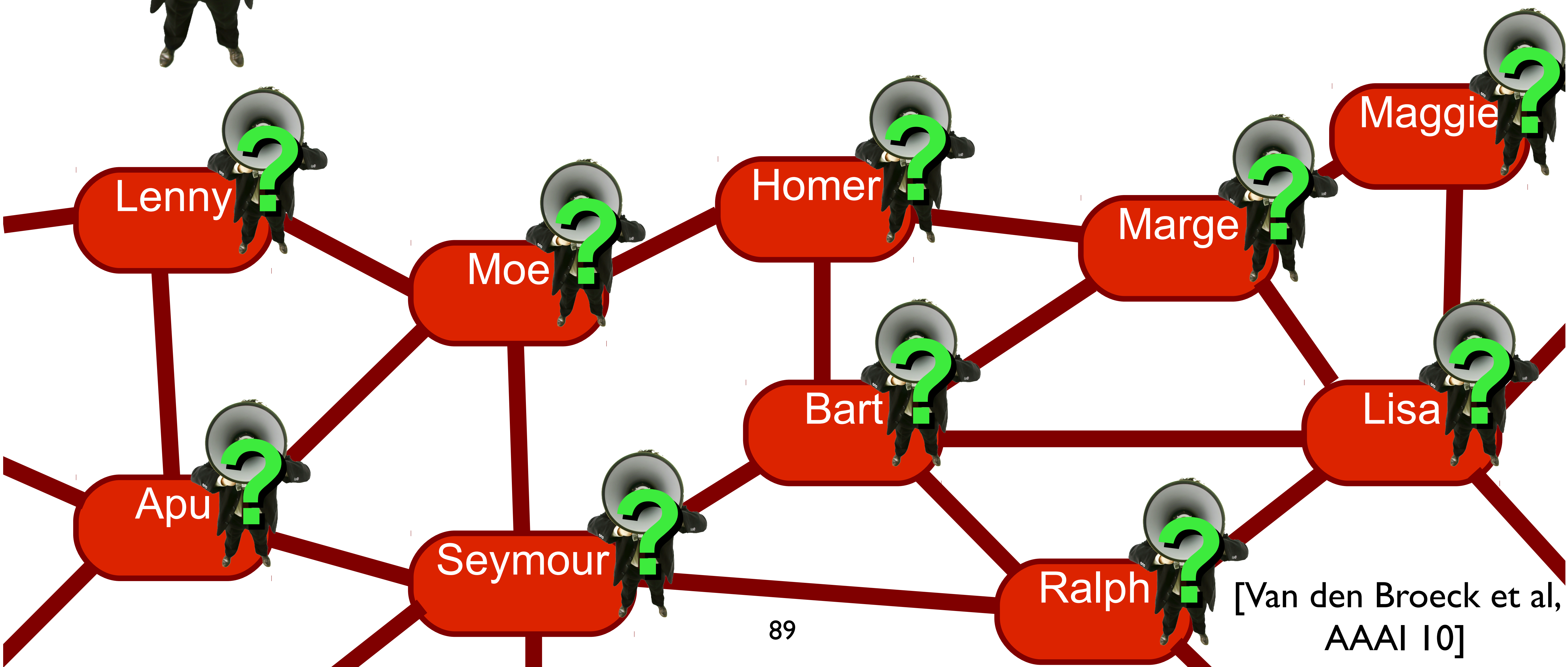
Part V : Decisions

Viral Marketing



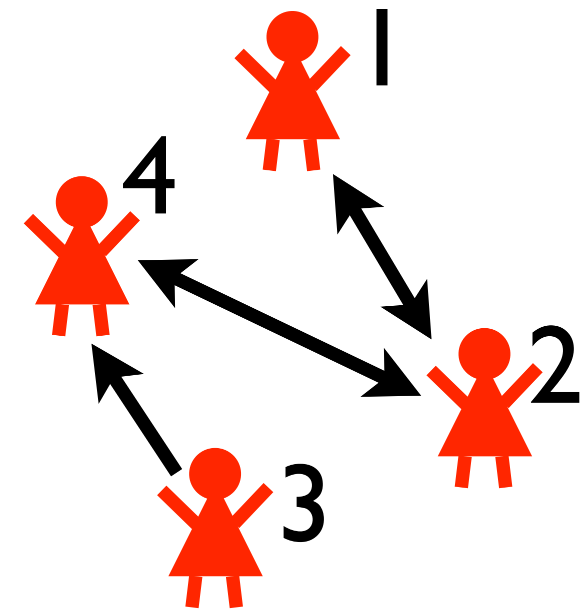
Which advertising strategy maximizes expected profit?

decide truth values of some atoms



[Van den Broeck et al, AAAI 10]

DTProbLog




```
? :: marketed(P) :- person(P).

0.3 :: buy_trust(X,Y) :- friend(X,Y).
0.2 :: buy_marketing(P) :- person(P).

buys(X) :- friend(X,Y), buys(Y), buy_trust(X,Y).
buys(X) :- marketed(X), buy_marketing(X).

buys(P) => 5 :- person(P).
marketed(P) => -3 :- person(P).
```

```
person(1).
person(2).
person(3).
person(4).

friend(1,2).
friend(2,1).
friend(2,4).
friend(3,4).
friend(4,2).
```

decision fact: true or false!

probabilistic facts + logical rules

utility facts: cost/reward if true

utility = -3 -3 + 5 + 5 = 4

probability = 0.0032

```
marketed(1)          marketed(3)
    bt(2,1)     bt(2,4)          bm(1)
buys(1)       buys(2)
```

world contributes 0.0032×4 to expected utility of strategy

task: find strategy that maximizes expected utility

solution: using ProbLog technology

Phenetic

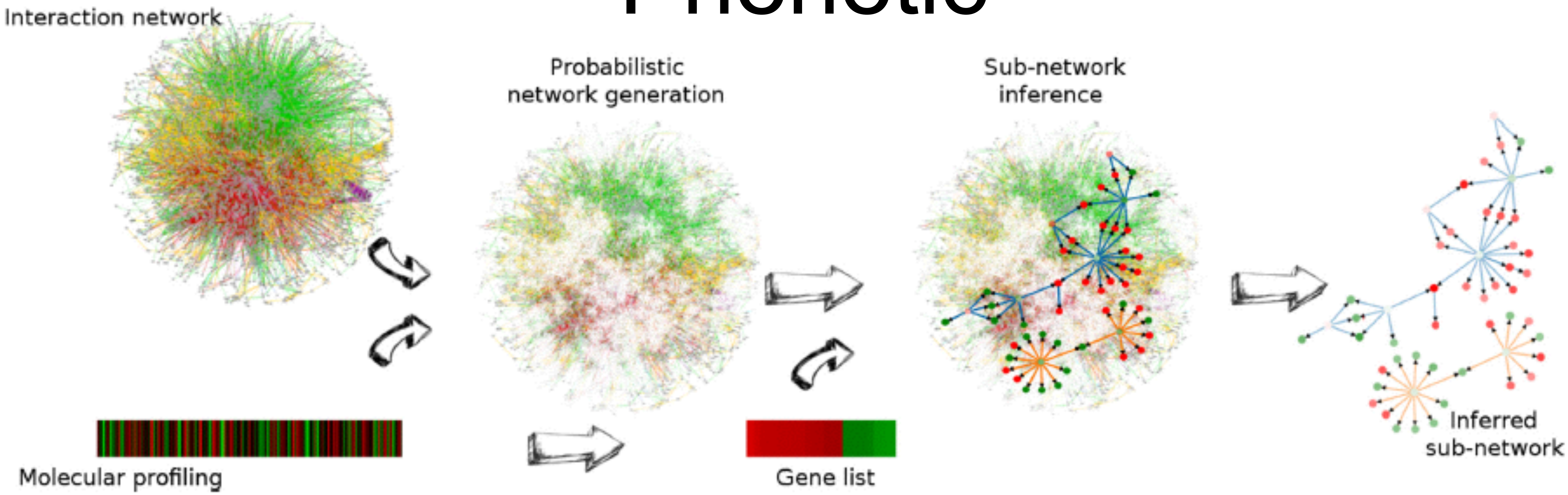


Figure 1. Overview of PheNetic, a web service for network-based interpretation of 'omics' data. The web service uses as input a genome wide interaction network for the organism of interest, a user generated molecular profiling data set and a gene list derived from these data. Interaction networks for a wide variety of organisms are readily available from the web server. Using the uploaded user-generated molecular data the interaction network is converted into a probabilistic network: edges receive a probability proportional to the levels measured for the terminal nodes in the molecular profiling data set. This probabilistic interaction network is used to infer the sub-network that best links the genes from the gene list. The inferred sub-network provides a trade-off between linking as many genes as possible from the gene list and selecting the least number of edges.

- Causes: Mutations
  - All related to similar phenotype
- Effects: Differentially expressed genes
- 27 000 cause effect pairs

- Interaction network:
  - 3063 nodes
    - Genes
    - Proteins
  - 16794 edges
    - Molecular interactions
    - Uncertain

- Goal: connect causes to effects through common subnetwork
  - = Find mechanism
- Techniques:
  - DTProbLog
  - Approximate inference

[De Maeyer et al., Molecular Biosystems 13, NAR 15]

Applications

- Medical reasoning (Peter Lucas et al)
- Knowledge base construction and Nell (De Raedt et al)
- Biology/Phenetic (De Maeyer et al, NAR 15)
- Robotics (Nitti et al., MLJ 16, MLJ 17, Moldovan et al. RA 17)
- Activity Recognition (Skarlatidis et al, TPLP 14)
- …

A key question in AI:

Dealing with uncertainty

- probability theory

dels

Reas

relat

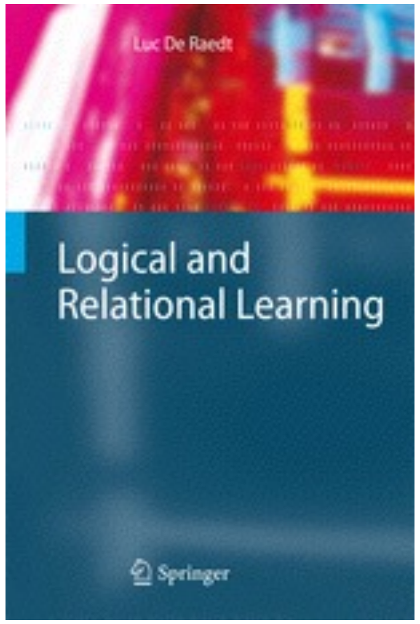
- lo
- da
- pr
- ...

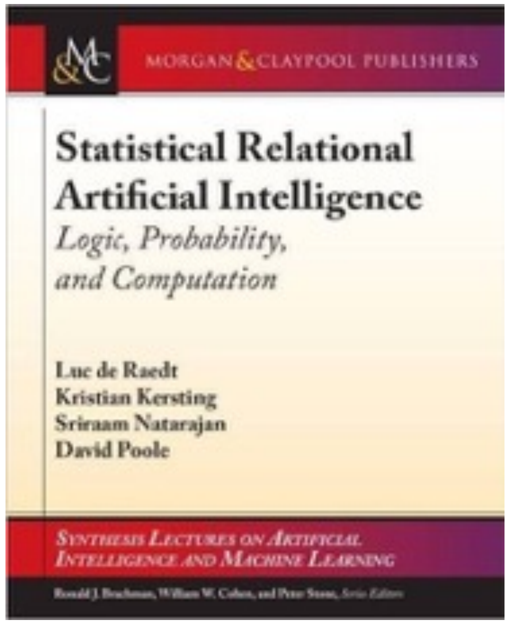
- Our answer: probabilistic (logic) programming
  = probabilistic choices + (logic) program
- Many languages, systems, applications, ...
- ... and much more to do!

Statistical relational learning, probabilistic logic learning, probabilistic programming, ...

Further Reading

- Logic and Learning
- Probabilistic programming
    - Logic programming and probabilistic databases
        - (ProbLog and DS as representatives)
    - http://dtai.cs.kuleuven.be/problog/ —
        - check also [DR & Kimmig, MLJ 15]
- Statistical relational AI and learning
    - Markov Logic

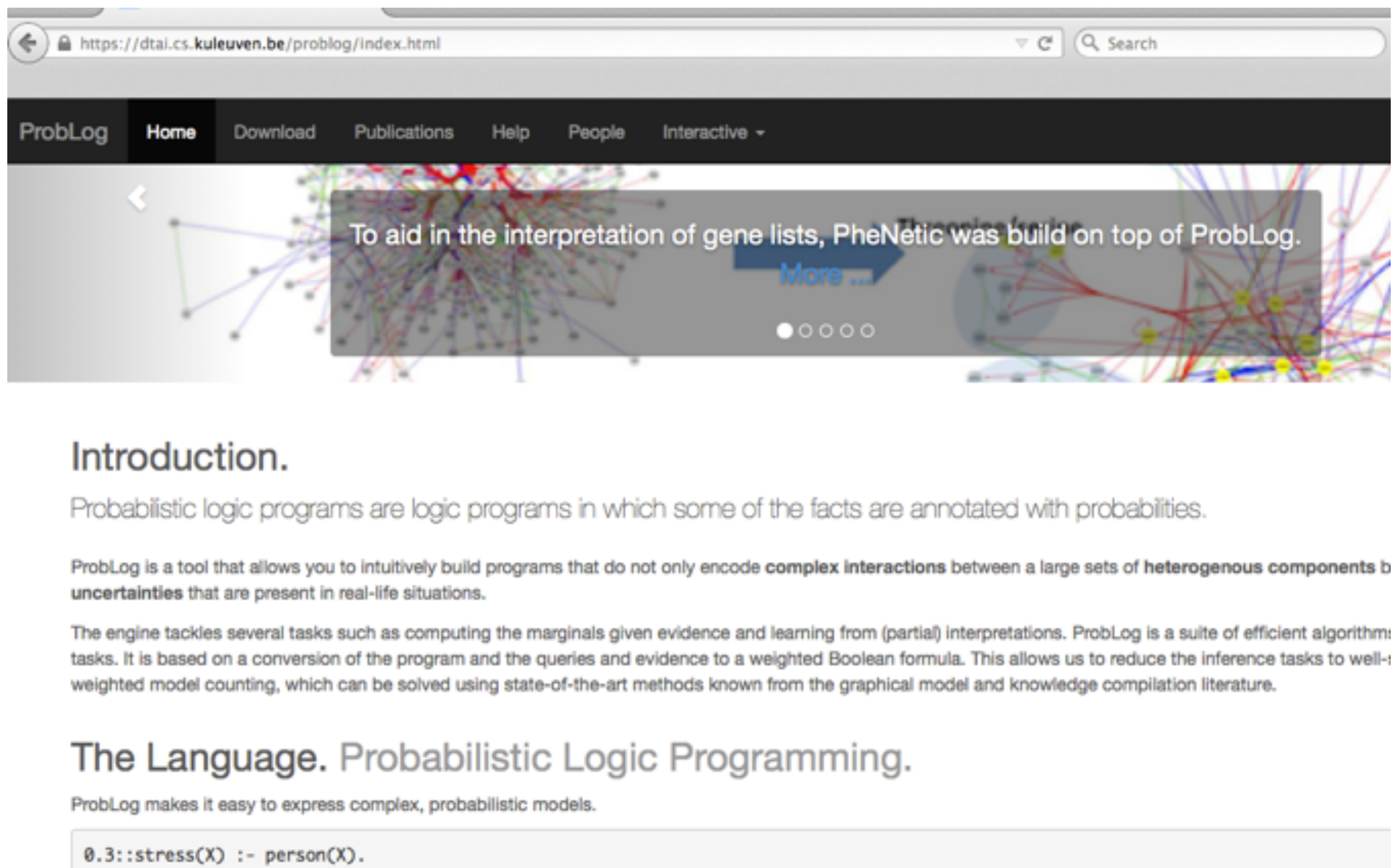




Maurice Bruynooghe
Bart Demoen
Anton Dries
Daan Fierens
Jason Filippou
Bernd Gutmann
Manfred Jaeger
Gerda Janssens
Kristian Kersting
Angelika Kimmig
Theofrastos Mantadelis
Wannes Meert
Bogdan Moldovan
Siegfried Nijssen
Davide Nitti
Joris Renkens
Kate Revoredo
Ricardo Rocha
Vitor Santos Costa
Dimitar Shterionov
Ingo Thon
Hannu Toivonen
Guy Van den Broeck
Mathias Verbeke
Jonas Vlasselaer

Thanks !

http://dtai.cs.kuleuven.be/problog



PLP Systems

- **PRISM** http://sato-www.cs.titech.ac.jp/prism/
- **ProbLog2** http://dtai.cs.kuleuven.be/problog/
- Yap Prolog http://www.dcc.fc.up.pt/~vsc/Yap/ includes
  - **ProbLog1**
  - **cplint** https://sites.google.com/a/unife.it/ml/cplint
  - **CLP(BN)**
  - **LP2**
- **PITA** in XSB Prolog http://xsb.sourceforge.net/
- **AILog2** http://artint.info/code/ailog/ailog2.html
- **SLPs** http://stoics.org.uk/~nicos/sware/pepl
- **contdist** http://www.cs.sunysb.edu/~cram/contdist/
- **DC** https://code.google.com/p/distributional-clauses
- **WFOMC** http://dtai.cs.kuleuven.be/ml/systems/wfomc

References

Bach SH, Broecheler M, Getoor L, O'Leary DP (2012) Scaling MPE inference for constrained continuous Markov random fields with consensus optimization. In: Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS-12)

Broecheler M, Mihalkova L, Getoor L (2010) Probabilistic similarity logic. In: Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI-10)

Bryant RE (1986) Graph-based algorithms for Boolean function manipulation. IEEE Transactions on Computers 35(8):677–691

Cohen SB, Simmons RJ, Smith NA (2008) Dynamic programming algorithms as products of weighted logic programs. In: Proceedings of the 24th International Conference on Logic Programming (ICLP-08)

Cussens J (2001) Parameter estimation in stochastic logic programs. Machine Learning 44(3):245–271

De Maeyer D, Renkens J, Cloots L, De Raedt L, Marchal K (2013) Phenetic: network-based interpretation of unstructured gene lists in e. coli. Molecular BioSystems 9(7):1594–1603

De Raedt L, Kimmig A (2013) Probabilistic programming concepts. CoRR abs/1312.4328

De Raedt L, Kimmig A, Toivonen H (2007) ProbLog: A probabilistic Prolog and its application in link discovery. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)

De Raedt L, Frasconi P, Kersting K, Muggleton S (eds) (2008) Probabilistic Inductive Logic Programming — Theory and Applications, Lecture Notes in Artificial Intelligence, vol 4911. Springer

Eisner J, Goldlust E, Smith N (2005) Compiling Comp Ling: Weighted dynamic programming and the Dyna language. In: Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP-05)

Fierens D, Blockeel H, Bruynooghe M, Ramon J (2005) Logical Bayesian networks and their relation to other probabilistic logical models. In: Proceedings of the 15th International Conference on Inductive Logic Programming (ILP-05)

Fierens D, Van den Broeck G, Bruynooghe M, De Raedt L (2012) Constraints for probabilistic logic programming. In: Proceedings of the NIPS Probabilistic Programming Workshop

Fierens D, Van den Broeck G, Renkens J, Shterionov D, Gutmann B, Thon I, Janssens G, De Raedt L (2014) Inference and learning in probabilistic logic programs using weighted Boolean formulas. Theory and Practice of Logic Programming (TPLP) FirstView

Getoor L, Friedman N, Koller D, Pfeffer A, Taskar B (2007) Probabilistic relational models. In: Getoor L, Taskar B (eds) An Introduction to Statistical Relational Learning, MIT Press, pp 129–174

Goodman N, Mansinghka VK, Roy DM, Bonawitz K, Tenenbaum JB (2008) Church: a language for generative models. In: Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)

Gutmann B, Thon I, De Raedt L (2011a) Learning the parameters of probabilistic logic programs from interpretations. In: Proceedings of the 22nd European Conference on Machine Learning (ECML-11)

Gutmann B, Thon I, Kimmig A, Bruynooghe M, De Raedt L (2011b) The magic of logical inference in probabilistic programming. Theory and Practice of Logic Programming (TPLP) 11((4–5)):663–680

Huang B, Kimmig A, Getoor L, Golbeck J (2013) A flexible framework for probabilistic models of social trust. In: Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction (SBP-13)

Jaeger M (2002) Relational Bayesian networks: A survey. Linköping Electronic Articles in Computer and Information Science 7(015)

Kersting K, Raedt LD (2001) Bayesian logic programs. CoRR cs.AI/0111058

Kimmig A, Van den Broeck G, De Raedt L (2011a) An algebraic Prolog for reasoning about possible worlds. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-11)

Kimmig A, Demoen B, De Raedt L, Santos Costa V, Rocha R (2011b) On the implementation of the probabilistic logic programming language ProbLog. Theory and Practice of Logic Programming (TPLP) 11:235–262

Koller D, Pfeffer A (1998) Probabilistic frame-based systems. In: Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)

McCallum A, Schultz K, Singh S (2009) FACTORIE: Probabilistic programming via imperatively defined factor graphs. In: Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS-09)

Milch B, Marthi B, Russell SJ, Sontag D, Ong DL, Kolobov A (2005) Blog: Probabilistic models with unknown objects. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)

Moldovan B, De Raedt L (2014) Occluded object search by relational affordances. In: IEEE International Conference on Robotics and Automation (ICRA-14)

Moldovan B, Moreno P, van Otterlo M, Santos-Victor J, De Raedt L (2012) Learning relational affordance models for robots in multi-object manipulation tasks. In: IEEE International Conference on Robotics and Automation (ICRA-12)

Muggleton S (1996) Stochastic logic programs. In: De Raedt L (ed) Advances in Inductive Logic Programming, IOS Press, pp 254–264

Nitti D, De Laet T, De Raedt L (2013) A particle filter for hybrid relational domains. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-13)

Nitti D, De Laet T, De Raedt L (2014) Relational object tracking and learning. In: IEEE International Conference on Robotics and Automation (ICRA), June 2014

Pfeffer A (2001) IBAL: A probabilistic rational programming language. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)

Pfeffer A (2009) Figaro: An object-oriented probabilistic programming language. Tech. rep., Charles River Analytics

Poole D (2003) First-order probabilistic inference. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)

Richardson M, Domingos P (2006) Markov logic networks. Machine Learning 62(1-2):107–136

Santos Costa V, Page D, Cussens J (2008) CLP(BN): Constraint logic programming for probabilistic knowledge. In: De Raedt et al (2008), pp 156–188

Sato T (1995) A statistical learning method for logic programs with distribution semantics. In: Proceedings of the 12th International Conference on Logic Programming (ICLP-95)

Sato T, Kameya Y (2001) Parameter learning of logic programs for symbolic-statistical modeling. J Artif Intell Res (JAIR) 15:391–454

Sato T, Kameya Y (2008) New advances in logic-based probabilistic modeling by prism. In: Probabilistic Inductive Logic Programming, pp 118–155

Skarlatidis A, Artikis A, Filiopou J, Paliouras G (2014) A probabilistic logic programming event calculus. Theory and Practice of Logic Programming (TPLP) FirstView

Suciu D, Olteanu D, Ré C, Koch C (2011) Probabilistic Databases. Synthesis Lectures on Data Management, Morgan & Claypool Publishers

Taskar B, Abbeel P, Koller D (2002) Discriminative probabilistic models for relational data. In: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI-02)

Thon I, Landwehr N, De Raedt L (2008) A simple model for sequences of relational state descriptions. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD-08)

Thon I, Landwehr N, De Raedt L (2011) Stochastic relational processes: Efficient inference and applications. Machine Learning 82(2):239–272

Van den Broeck G, Thon I, van Otterlo M, De Raedt L (2010) DTProbLog: A decision-theoretic probabilistic Prolog. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10)

Van den Broeck G, Taghipour N, Meert W, Davis J, De Raedt L (2011) Lifted probabilistic inference by first-order knowledge compilation. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)

Vennekens J, Verbaeten S, Bruynooghe M (2004) Logic programs with annotated disjunctions. In: Proceedings of the 20th International Conference on Logic Programming (ICLP-04)

Vennekens J, Denecker M, Bruynooghe M (2009) CP-logic: A language of causal probabilistic events and its relation to logic programming. Theory and Practice of Logic Programming (TPLP) 9(3):245–308

Wang WY, Mazaitis K, Cohen WW (2013) Programming with personalized pagerank: a locally groundable first-order probabilistic logic. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM-13)