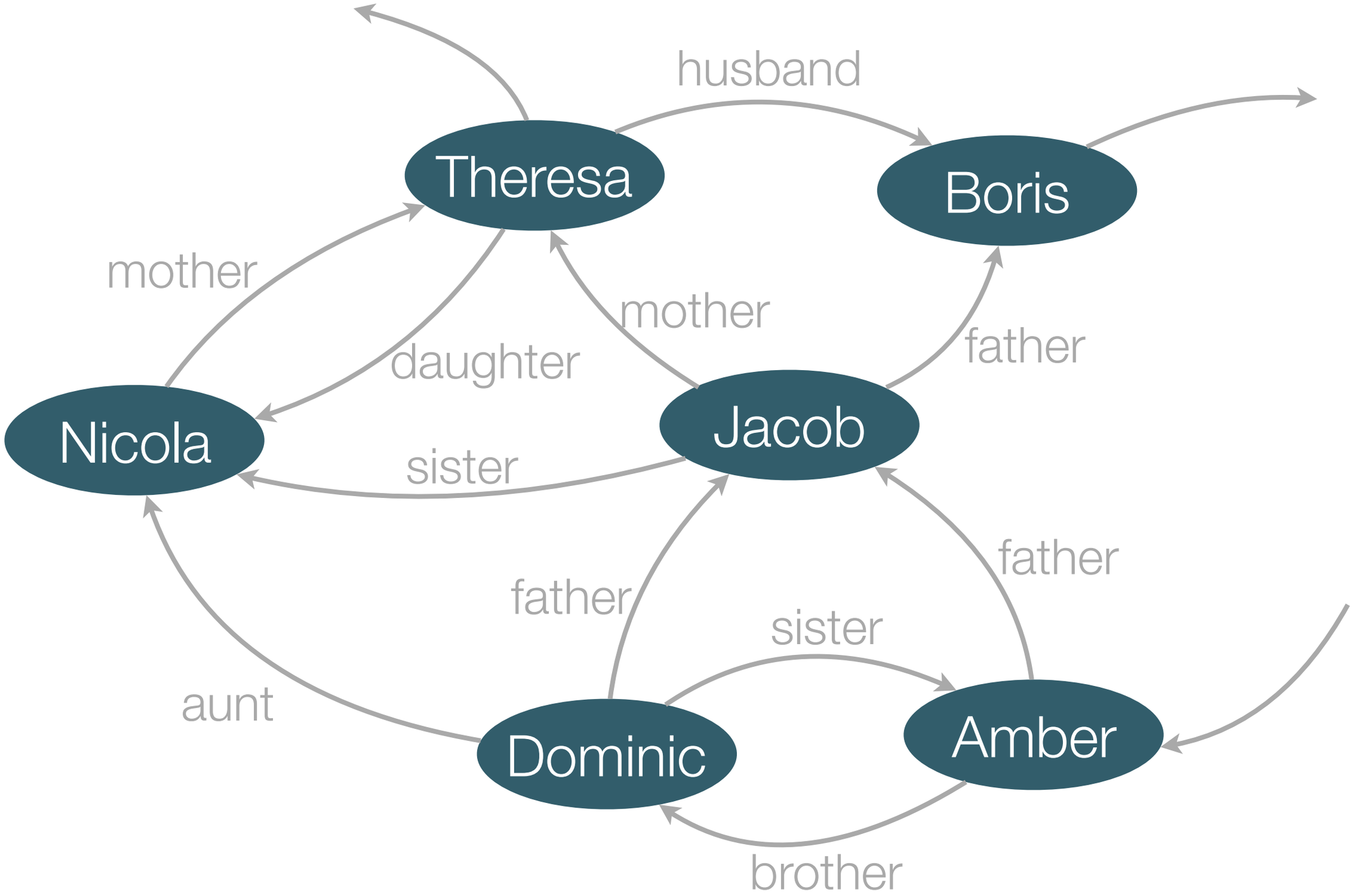
Use of a possibilistic logic-like framework for statistical relational learning

Steven Schockaert

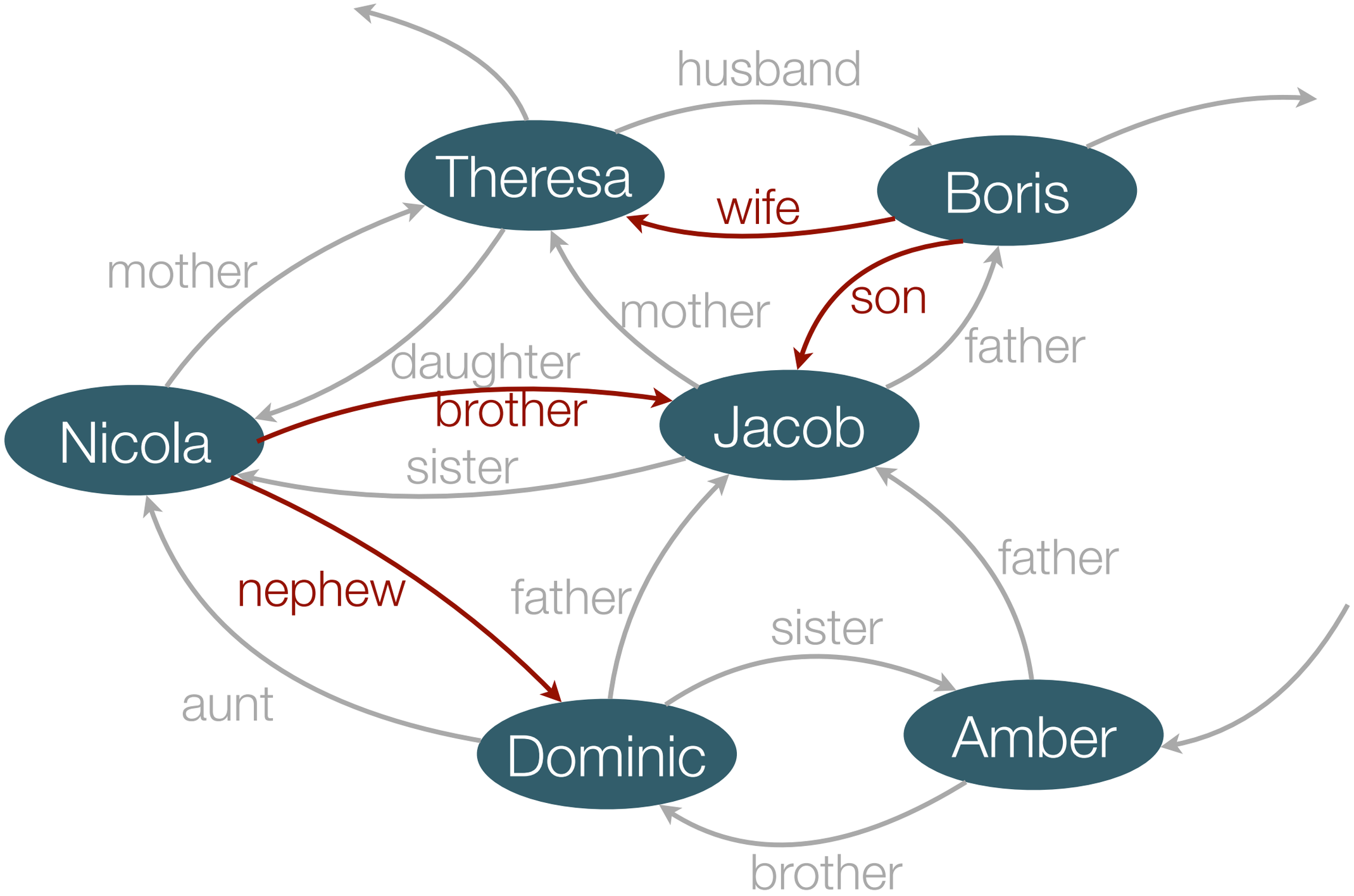
Joint work with: Ondrej Kuzelka, Jesse Davis, Martin Svatos

School of Computer Science & Informatics
Cardiff University, Cardiff, UK
SchockaertS1@cardiff.ac.uk
http://users.cs.cf.ac.uk/S.Schockaert

Predicting missing relational facts



Predicting missing relational facts



Knowledge graph embedding

Learn vector representation for each entity and scoring function for each relation such that:

$$P(e, r, f) = \text{score}_r(\mathbf{e}, \mathbf{f})$$

Drawbacks:

- Poor performance if “test graph” is sparse
- Only able to capture relatively simple dependencies
- Black box

Advantages:

- Scalable (millions of entities)
- Easy to incorporate text descriptions, images, etc.
- Only able to capture relatively simple dependencies

Markov logic networks

0	$\mathit{FacultyEmeritus}(V1)$
-5.64478	$\mathit{AdvisedBy}(V1, V2)$
-5.10868	$\mathit{Student}(V1)$
-2.06003	$\mathit{FacultyAffiliate}(V1)$
-3.18786	$\mathit{PostQuals}(V1)$
-3.64133	$\mathit{PostGenerals}(V1)$
-0.80596	$\mathit{FacultyAdjunct}(V1)$
-0.0391718	$\mathit{TempAdvisedBy}(V1, V1)$
-4.12274	$\mathit{Publication}(V1, V2)$
-0.0551981	$\mathit{AdvisedBy}(V1, V1)$
-2.95067	$\mathit{Faculty}(V1)$
-6.56008	$\mathit{ta}(V1, V2, V3)$
-4.90068	$\mathit{ProjectMember}(V1, V2)$
-4.44769	$\mathit{TempAdvisedBy}(V1, V2)$
5.10868	$\mathit{Professor}(V1)$
-6.31879	$\mathit{TaughtBy}(V1, V2, V3)$
-4.69704	$\mathit{PreQuals}(V1)$
1.67999e-06	$\neg\mathit{TaughtBy}(V1, V2, V3) \vee \neg\mathit{FacultyAdjunct}(V2) \vee \neg\mathit{TempAdvisedBy}(V2, V4)$
1.08275	$\neg\mathit{TaughtBy}(V1, V2, V3) \vee \neg\mathit{FacultyAdjunct}(V2) \vee \neg\mathit{Publication}(V4, V2)$
6.07986	$\mathit{Faculty}(V1) \vee \neg\mathit{TempAdvisedBy}(V2, V1) \vee \mathit{FacultyAffiliate}(V1)$
2.48337	$\mathit{PreQuals}(V1) \vee \mathit{PostQuals}(V1) \vee \mathit{ProjectMember}(V2, V1) \vee \neg\mathit{TempAdvisedBy}(V1, V3)$
9.33506	$\mathit{PreQuals}(V1) \vee \mathit{PostQuals}(V1) \vee \neg\mathit{AdvisedBy}(V1, V2) \vee \mathit{PostGenerals}(V1)$
-12.0071	$\neg\mathit{Student}(V1) \vee \mathit{FacultyAdjunct}(V1) \vee \mathit{Professor}(V1) \vee \mathit{Faculty}(V1) \vee \mathit{FacultyAffiliate}(V1)$
-2.05975	$\mathit{PostQuals}(V1) \vee \mathit{FacultyAdjunct}(V1) \vee \neg\mathit{Faculty}(V1) \vee \neg\mathit{AdvisedBy}(V2, V1) \vee \mathit{FacultyAffiliate}(V1)$

UWCSE

Markov logic networks

Learn domain knowledge in the form of a weighted set of rules w:F

$$p_{\mathcal{M}}(\omega) = \frac{1}{Z}\exp\left(\sum_{w:F\in\mathcal{M}} w n_F(\omega)\right)$$

number of groundings of F satisfied in ω

normalisation constant

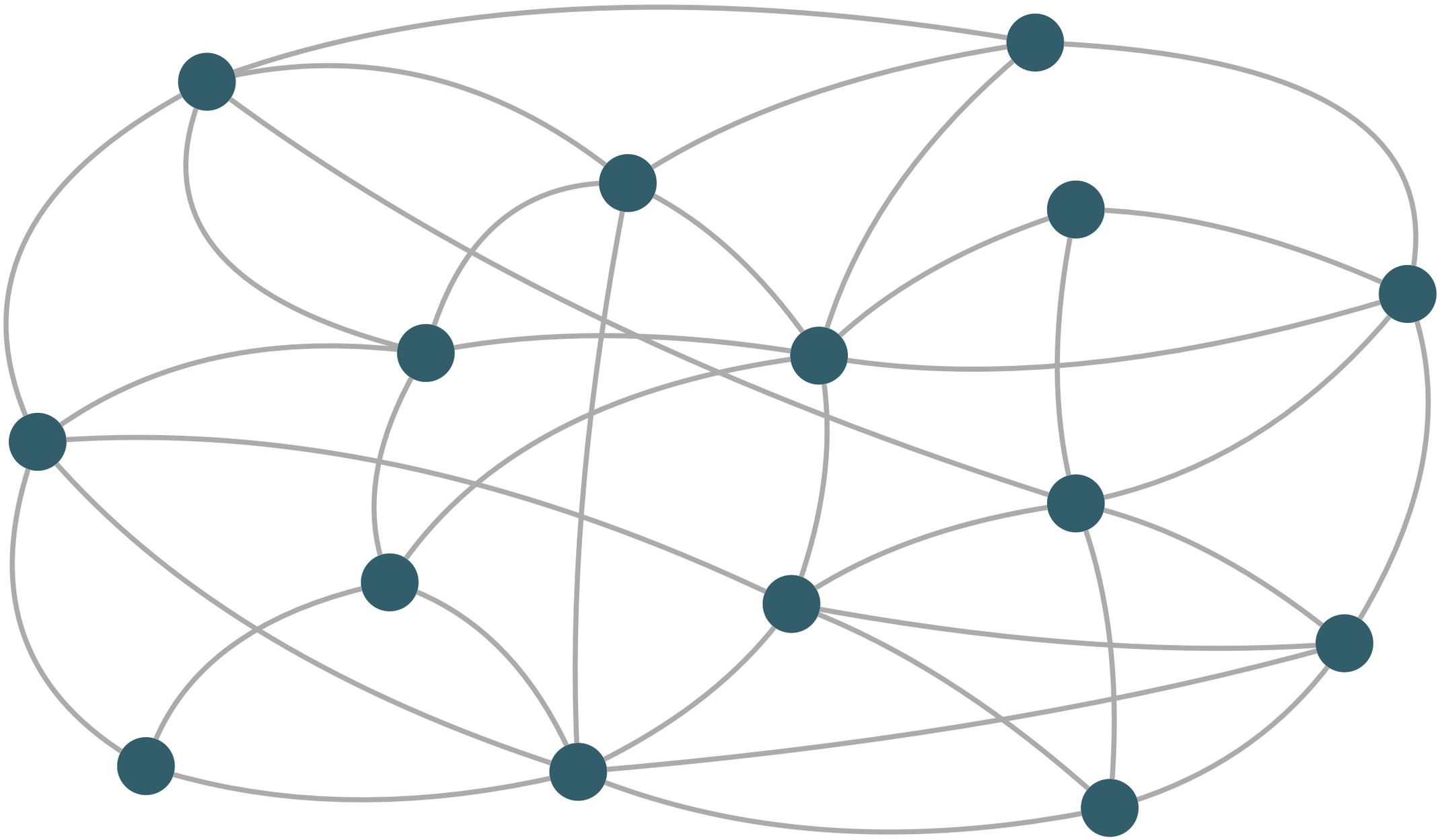
Given some set of evidence literals E, find the most probable world ω satisfying all literals in E; predict all literals satisfied in ω as true

Drawbacks:

- Weights dependent on domain size
- Interaction of the rules hard to interpret
- Computationally rather expensive

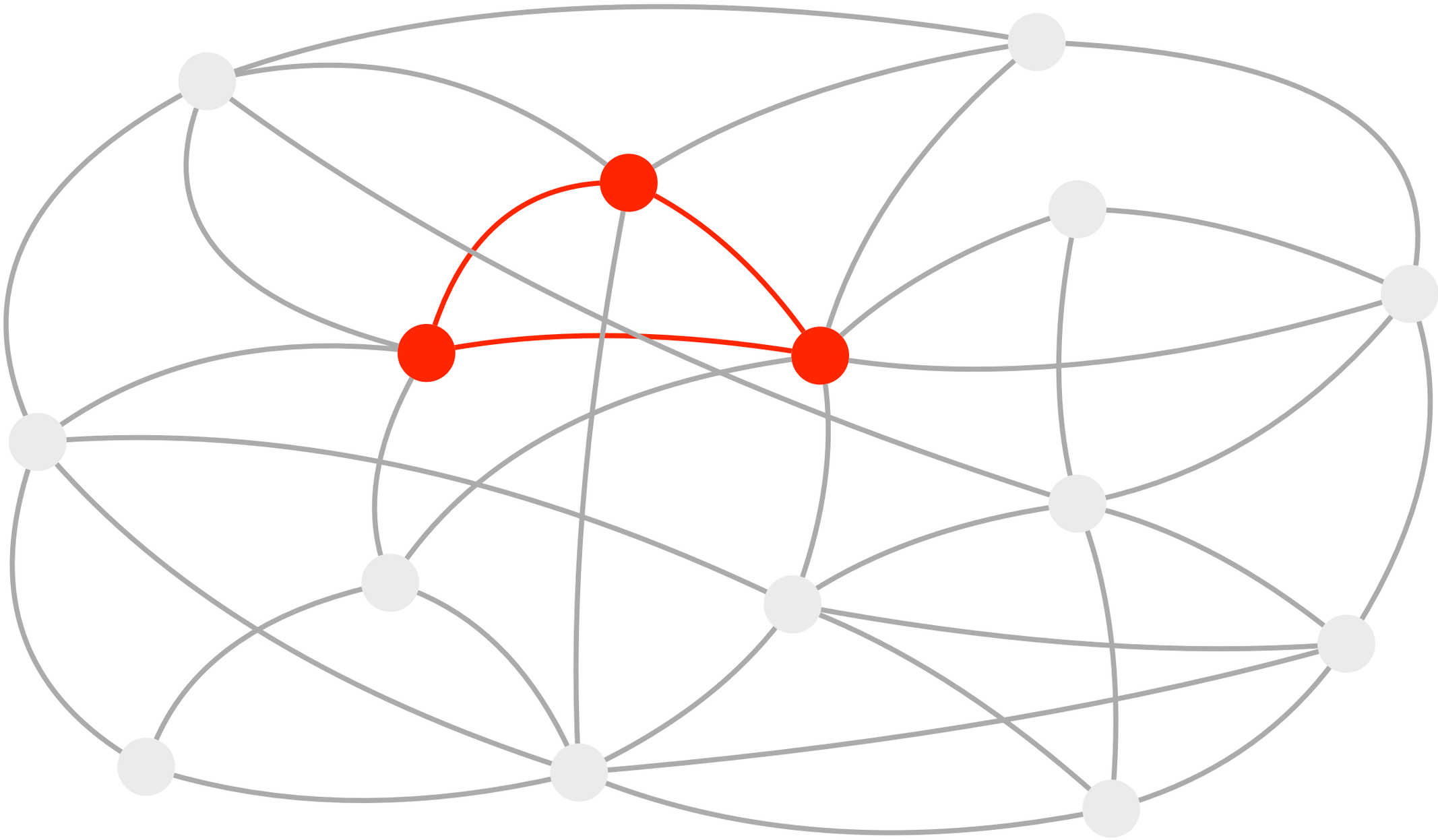
Relational marginals

We only have one relational graph, i.e. one training example. How can we estimate probabilities from one example? What does the probability of a (universally quantified) clause represent?



Relational marginals

Relational marginal distribution ≈ distribution over size-k subgraphs



Relational marginals

Relational marginal distribution ≈ distribution over size-k subgraphs

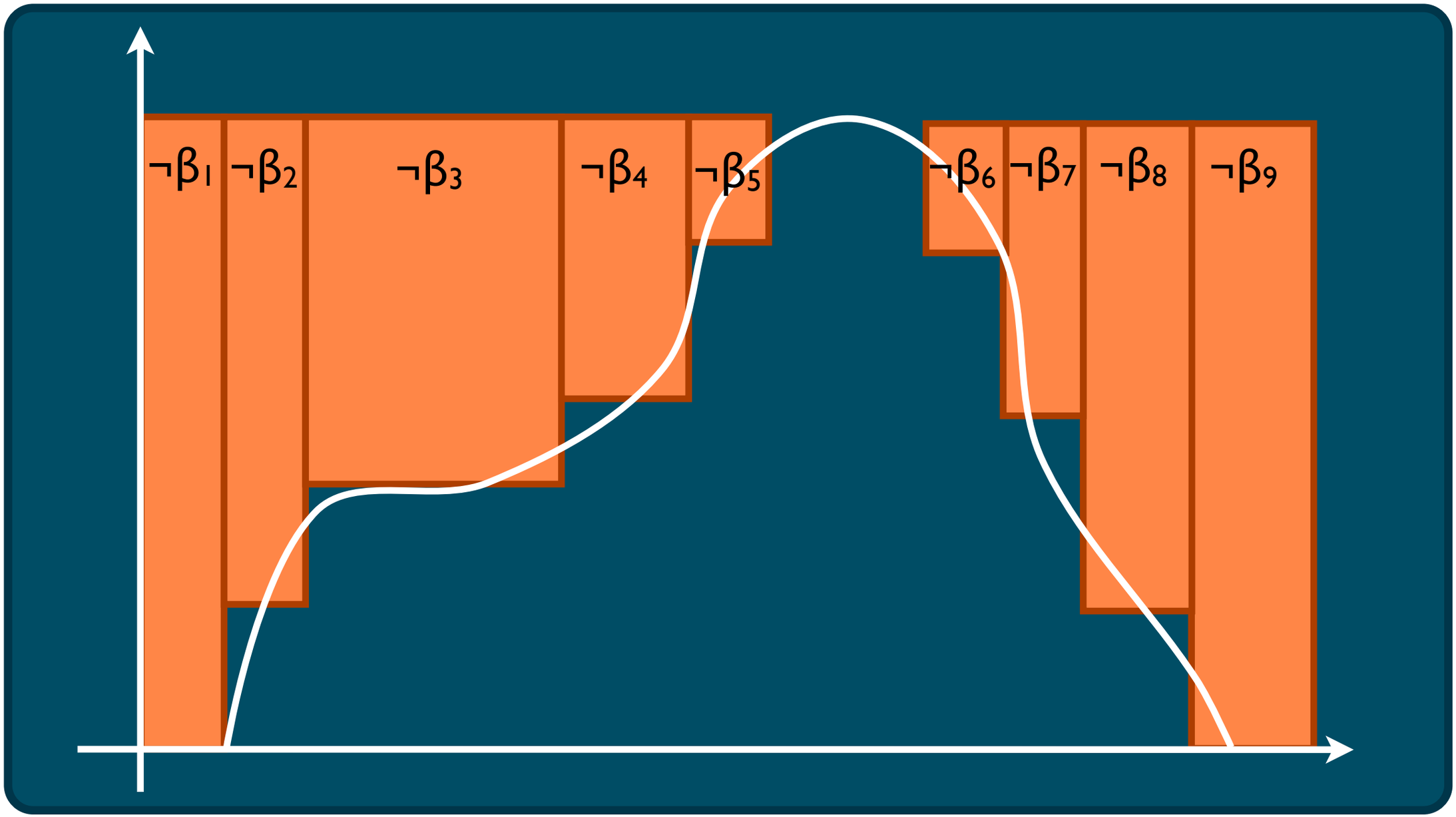
Probability of a clause given by the relational marginal distribution:

$$P_{\Upsilon,k}(\alpha) = \sum_{\omega:\omega \models \alpha} P_{\Upsilon,k}(\omega)$$

Main idea: encode the relational marginal distribution using possibilistic logic

- Needs first-order formulas, but these are merely templates for sets of ground formulas
- Exact encoding exponentially large, so need heuristic method for selecting set of rules

Possibilistic encoding of probability distributions



Possibilistic logic: $\pi(\omega) = \min\{1 - \lambda \,:\, \omega \not\models \beta, (\beta, \lambda) \in \Theta\}$

Relational possibilistic logic

$\text{son}(X, Y) \wedge \text{male}(Y) \rightarrow \text{father}(Y, X)$ 1

$\text{son}(X, Y) \wedge \text{son}(Z, Y) \rightarrow \text{sibling}(X, Z)$ 0.5

$\text{son}(X, Y) \wedge \text{married}(Y, Z) \rightarrow \text{son}(X, Z)$ 0.25

Relational possibilistic logic

$E = \{\text{son}(a, b), \text{married}(b, c)\}$

$\text{son}(a, b) \wedge \text{male}(b) \rightarrow \text{father}(b, a)$	1
$\cdots$	
$\text{son}(c, b) \wedge \text{male}(b) \rightarrow \text{father}(b, c)$	
$\text{son}(a, b) \wedge \text{son}(c, b) \rightarrow \text{sibling}(a, c)$	0.5
$\cdots$	
$\text{son}(c, b) \wedge \text{son}(a, b) \rightarrow \text{sibling}(c, a)$	
$\text{son}(a, b) \wedge \text{married}(b, c) \rightarrow \text{son}(a, c)$	0.25
$\cdots$	
$\text{son}(c, b) \wedge \text{married}(b, a) \rightarrow \text{son}(c, a)$	

Rule learning

Learn a set of hard constraints

- Exhaustive search over all small clauses
- Keep all clauses which are never violated in the given knowledge graph

Construct training examples for rule learner

- **Positive examples**: literals from the knowledge graph
- **Negative examples**: literals which are missing from the knowledge graph but which are consistent with the learned constraints

Rule learner

- Use beam search to find a rule which maximises accuracy
- Repeat process several times for each predicate

Weight learning

Main principle for weight learning is maximising the likelihood of a given set of sampled sub-graphs

Assume ordering $\alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n$ is given

Number of worlds whose probability is 1-λ_i'

$$\textbf{Maximize: } \prod_{\omega \in E} P(\omega) = \prod_{i=1}^{n} (1 - \lambda_i')^{|\mathcal{E}_{i+1}| - |\mathcal{E}_i|}$$
$$\text{where } \mathcal{E}_i = \{\omega \in \mathcal{E} | \omega \models \alpha_i \wedge \cdots \wedge \alpha_n\}.$$

Subject to:

Weights to be learned have to satisfy the given ordering

$$\lambda_1' \leq \lambda_2' \leq \cdots \leq \lambda_n'$$

Ensure probabilities sum to 1

$$\sum_{i=1}^{k} (1 - \lambda_i') \cdot (|M_{i+1}| - |M_i|) = 1$$

Number of models of $\alpha_i \wedge \ldots \wedge \alpha_n$

Can be solved using geometric programming

Ordering of the rules is done in a greedy fashion

Learned theories

$\ldots$ (116 hard constraints not shown here)

$(\mathit{Publication}(V0, V1) \leftarrow \mathit{TempAdvisedBy}(V3, V1) \wedge \mathit{Publication}(V0, V3), \lambda_{16})$

$(\mathit{PostQuals}(V0) \leftarrow \mathit{TempAdvisedBy}(V0, V3) \wedge \mathit{TempAdvisedBy}(V4, V3) \wedge \mathit{PostQuals}(V4), \lambda_{15})$

$(\mathit{AdvisedBy}(V0, V1) \leftarrow \mathit{FacultyAffiliate}(V1), \mathit{Publication}(V3, V1) \wedge \mathit{Publication}(V3, V0), \lambda_{15})$

$(\mathit{PostGenerals}(V0) \leftarrow \mathit{Publication}(V2, V0) \wedge \mathit{AdvisedBy}(V0, V4), \lambda_{14})$

$(\mathit{FacultyAdjunct}(V0) \leftarrow \mathit{AdvisedBy}(V2, V0), \mathit{FacultyAdjunct}(V4) \wedge \mathit{AdvisedBy}(V2, V4), \lambda_{13})$

$(\mathit{PostGenerals}(V0) \leftarrow \mathit{TaughtBy}(V2, V0, V4) \wedge \mathit{Student}(V0), \lambda_{13})$

$(\mathit{PostQuals}(V0) \leftarrow \mathit{AdvisedBy}(V0, V3) \wedge \mathit{ProjectMember}(V4, V3), \lambda_{12})$

$(\mathit{AdvisedBy}(V0, V1) \leftarrow \mathit{Publication}(V3, V1) \wedge \mathit{FacultyAdjunct}(V1) \wedge \mathit{Publication}(V3, V0), \lambda_{11})$

$(\mathit{PostQuals}(V0) \leftarrow \mathit{AdvisedBy}(V0, V3) \wedge \mathit{PostQuals}(V4) \wedge \mathit{AdvisedBy}(V4, V3), \lambda_{10})$

$(\mathit{Publication}(V0, V1) \leftarrow \mathit{FacultyAffiliate}(V4), \mathit{FacultyAffiliate}(V1) \wedge \mathit{Publication}(V0, V4), \lambda_{9})$

$(\mathit{PostGenerals}(V0) \leftarrow \mathit{AdvisedBy}(V0, V3) \wedge \mathit{PostGenerals}(V4) \wedge \mathit{AdvisedBy}(V4, V3), \lambda_{8})$

$(\mathit{Faculty}(V0) \leftarrow \mathit{TempAdvisedBy}(V2, V0), \lambda_{7})$

$(\mathit{PostQuals}(V0) \leftarrow \mathit{AdvisedBy}(V0, V3) \wedge \mathit{FacultyAdjunct}(V3), \lambda_{6})$

$(\mathit{Faculty}(V0) \leftarrow \mathit{AdvisedBy}(V2, V0), \lambda_{5})$

$(\mathit{Publication}(V0, V1) \leftarrow \mathit{FacultyAdjunct}(V4), \mathit{FacultyAdjunct}(V1) \wedge \mathit{Publication}(V0, V4), \lambda_{4})$

$(\mathit{PreQuals}(V0) \leftarrow \mathit{TempAdvisedBy}(V0, V3), \lambda_{3})$

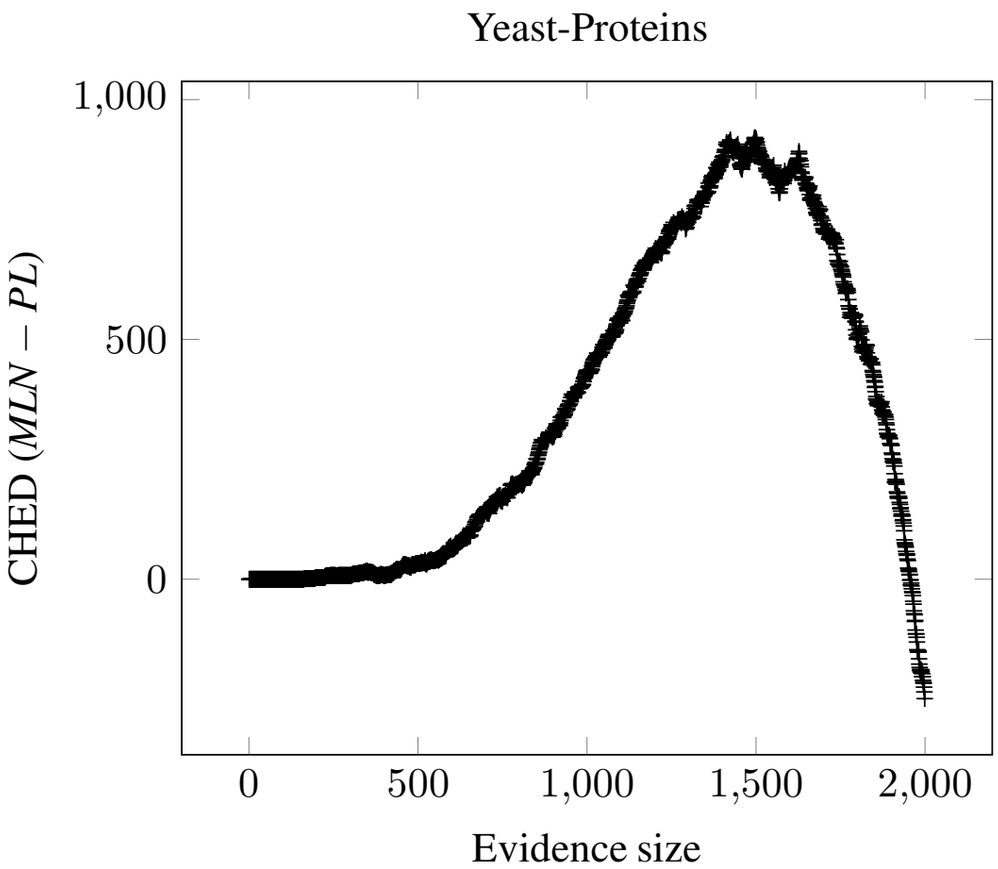
$(\mathit{FacultyAdjunct}(V0) \leftarrow \mathit{Publication}(V2, V4), \mathit{PreQuals}(V4) \wedge \mathit{Publication}(V2, V0), \lambda_{2})$

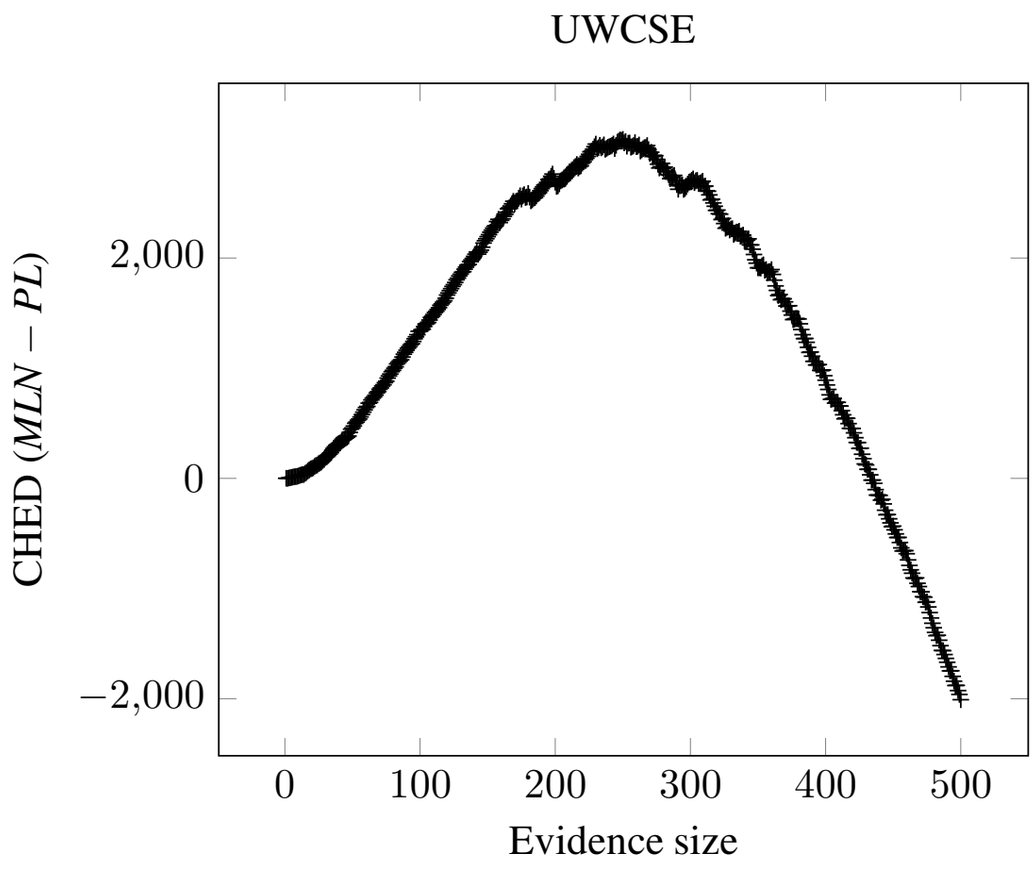
$(\mathit{PostQuals}(V0) \leftarrow \mathit{AdvisedBy}(V0, V3), \lambda_{1})$

$(\bot, \lambda_{\bot})$

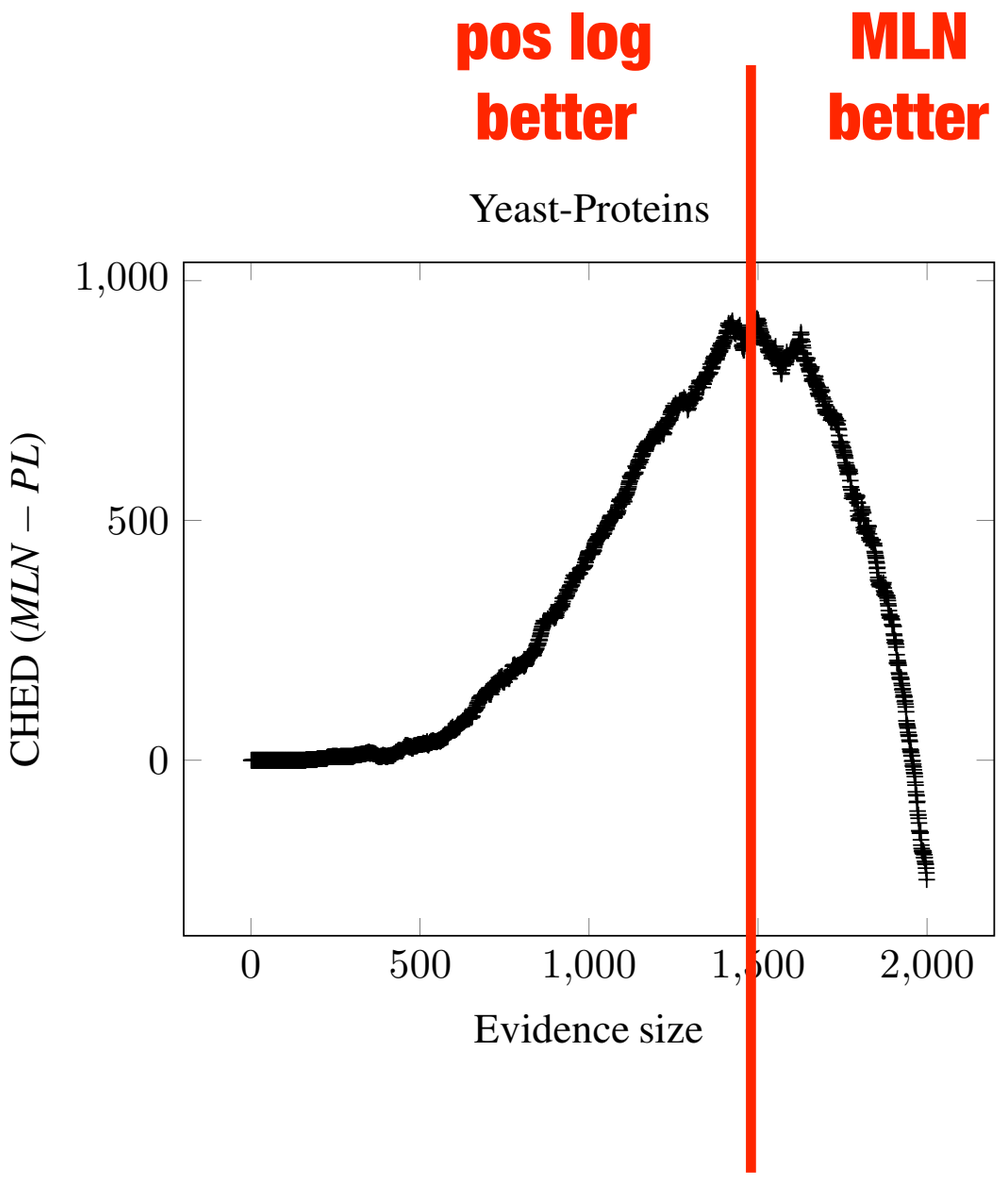
UWCSE

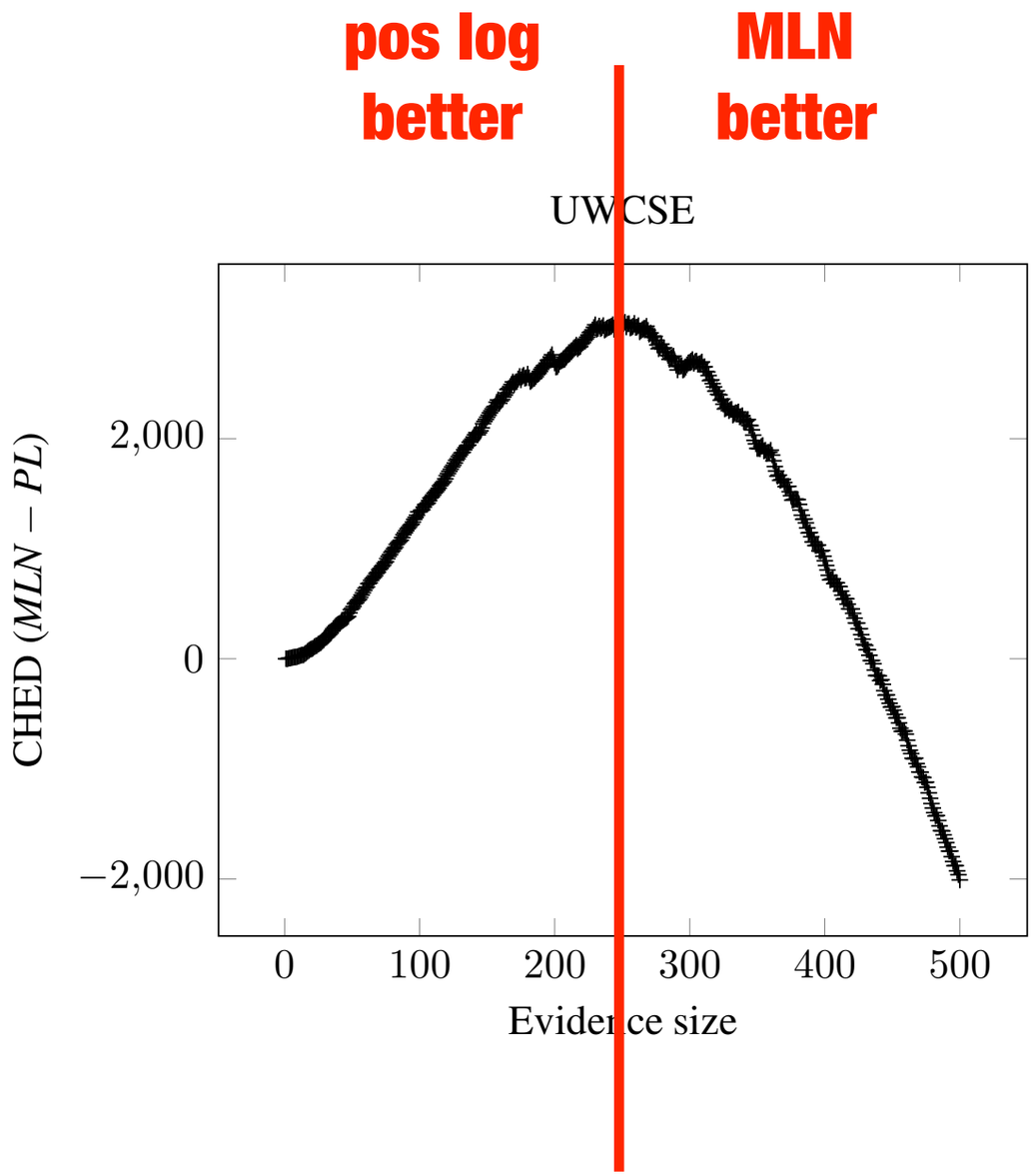
Results


Yeast-Proteins
1,000
500
0
CHED (MLN − PL)
0
500
1,000
1,500
2,000
Evidence size


UWCSE
2,000
0
−2,000
CHED (MLN − PL)
0
100
200
300
400
500
Evidence size

Results


pos log better
MLN better
Yeast-Proteins
CHED (MLN − PL)
1,000
500
0
0
500
1,000
1,500
2,000
Evidence size


pos log better
MLN better
UWCSE
CHED (MLN − PL)
2,000
0
−2,000
0
100
200
300
400
500
Evidence size

Cautious inference with learned rules

Learned rules are noisy, which means that we cannot simply use classical logic

Possibilistic logic addresses this by prioritising the most certain rules in case of conflict

Another possibility is to weaken the classical entailment relation

- AMIE: only allow 1-step inferences
- We want something in between

k-entailment

Given a set of rules Φ, a set of constraints Γ and a set of facts E, we say that a fact α is k-entailed if there exists a set of at most k constants C such that:

Restricting the facts in E to those that only contain the constants from C restores consistency

Restricting the facts in E still allows us to derive α

Stratified k-entailment (STRiKE)

α_1

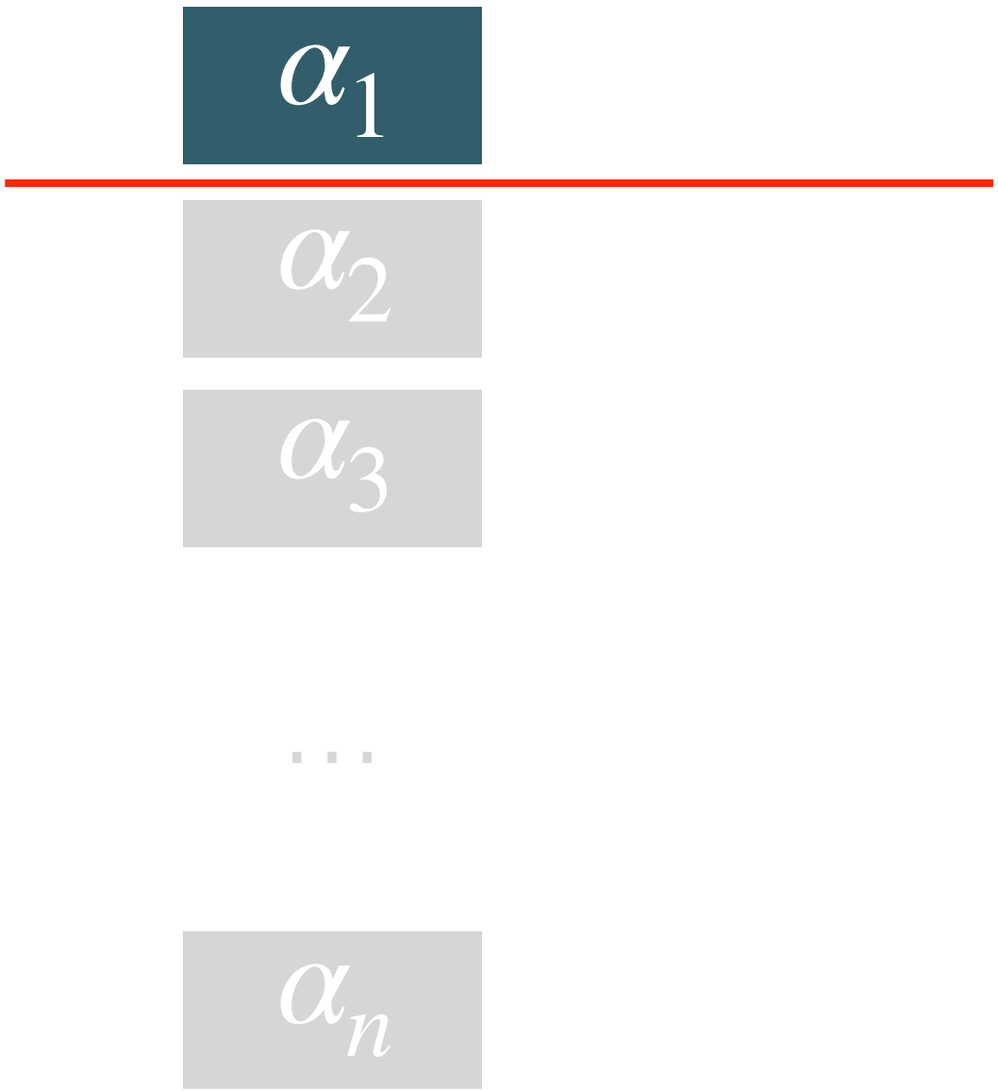
α_2

α_3

...

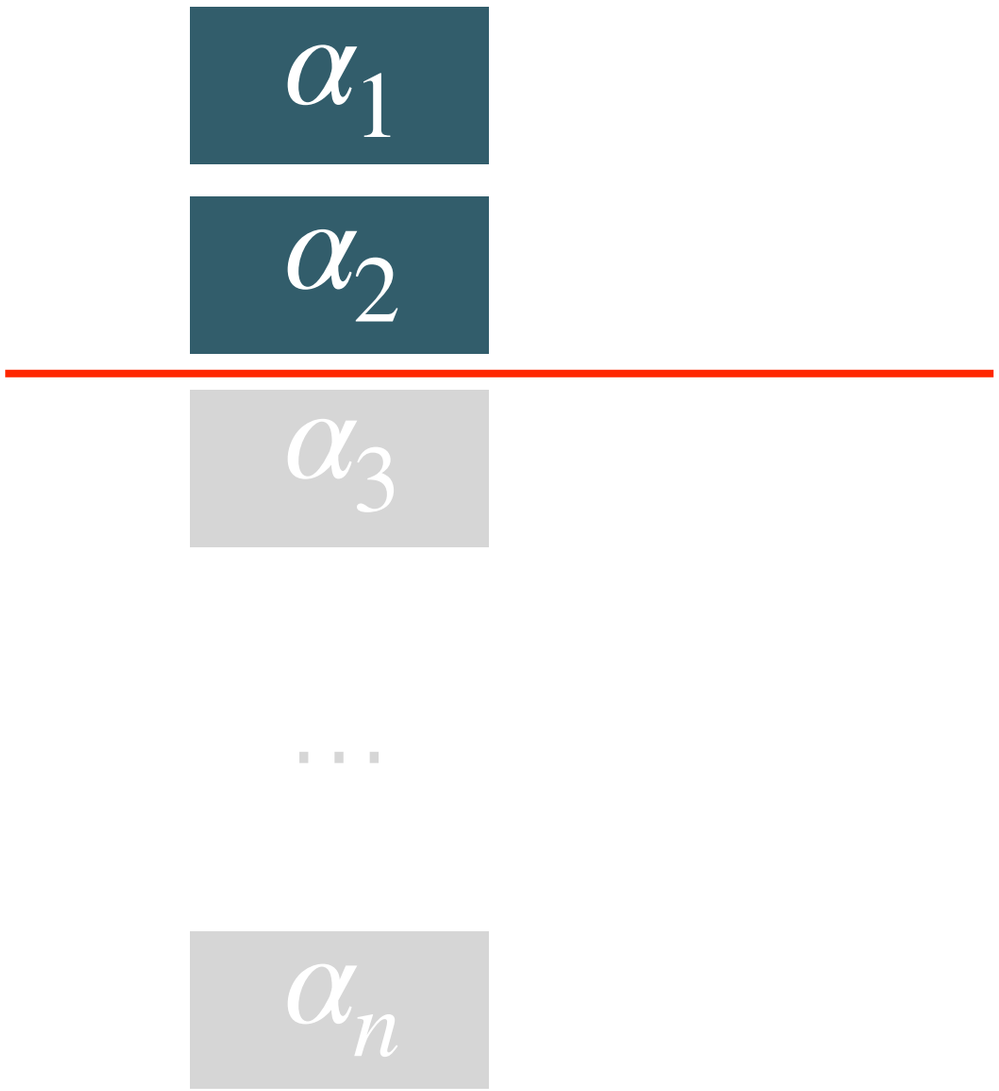
α_n

Stratified k-entailment (STRiKE)



Use k-entailment to find all facts that can be derived from α_1

Stratified k-entailment (STRiKE)



Use k-entailment to find all facts that can be derived from {α_1,α_2}, but assign a lower certainty level to these facts

Stratified k-entailment (STRiKE)

α_1

α_2

α_3

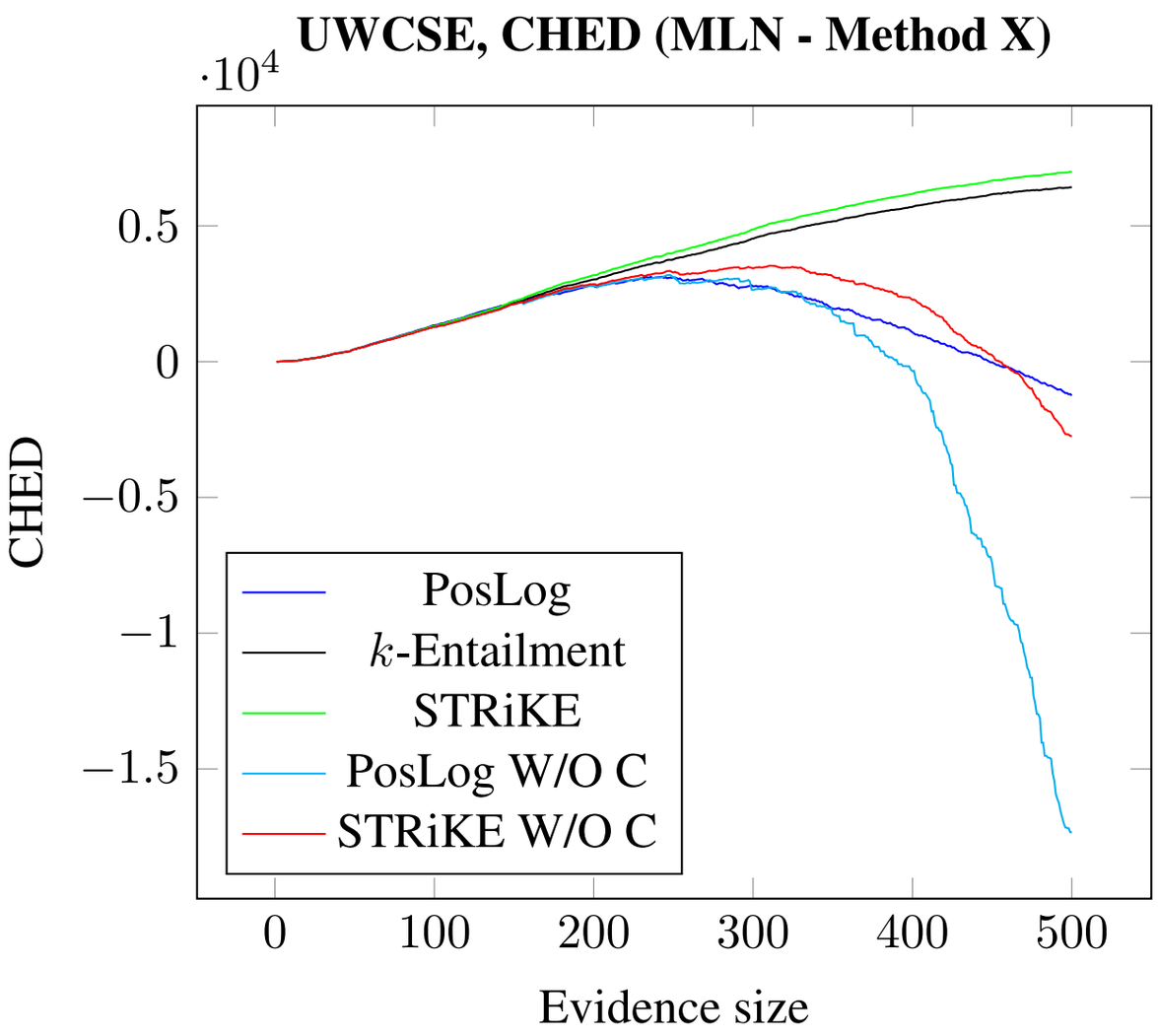
...

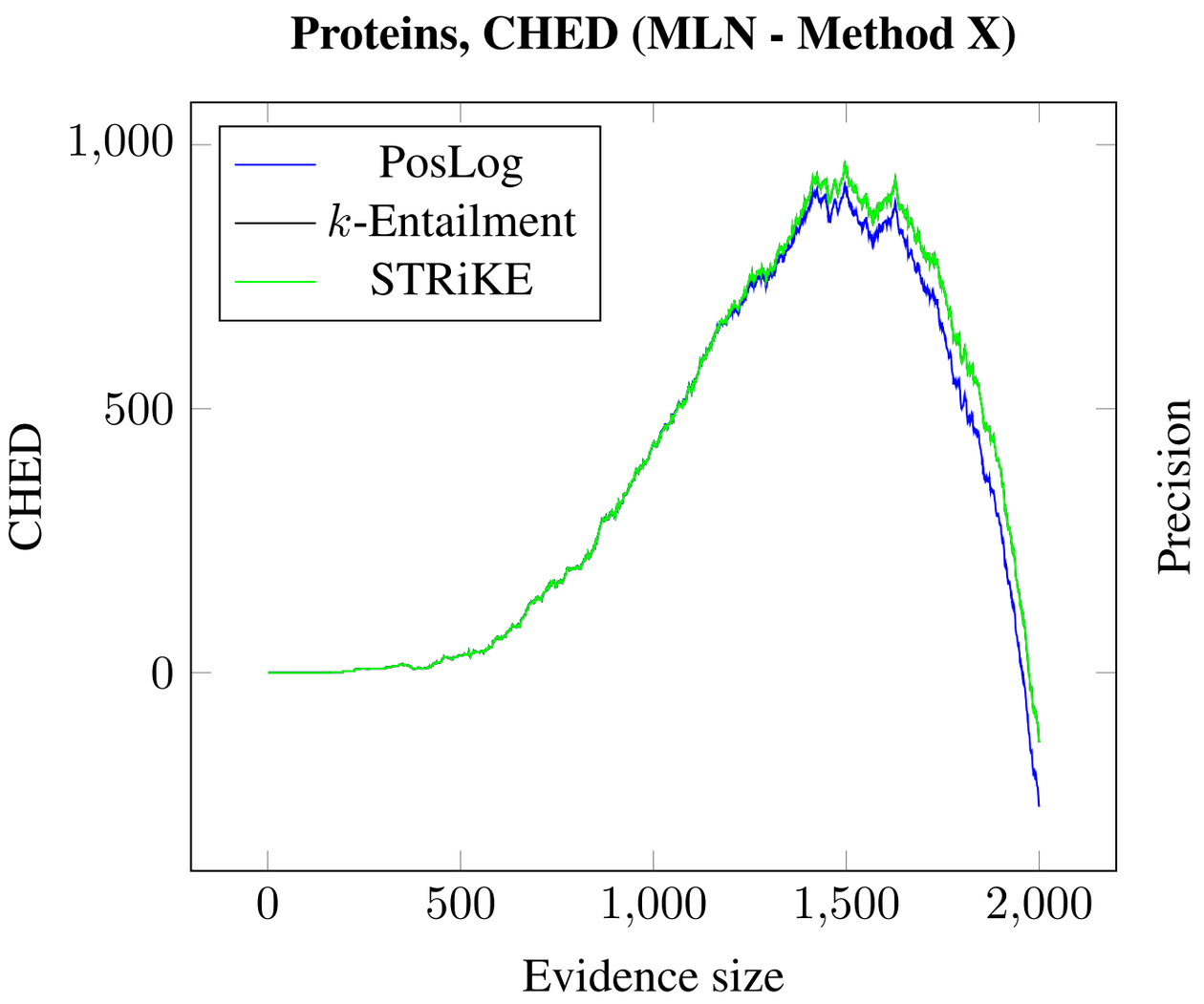
α_n

Use k-entailment to find all facts that can be derived from $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, and assign the lowest certainty level to these facts

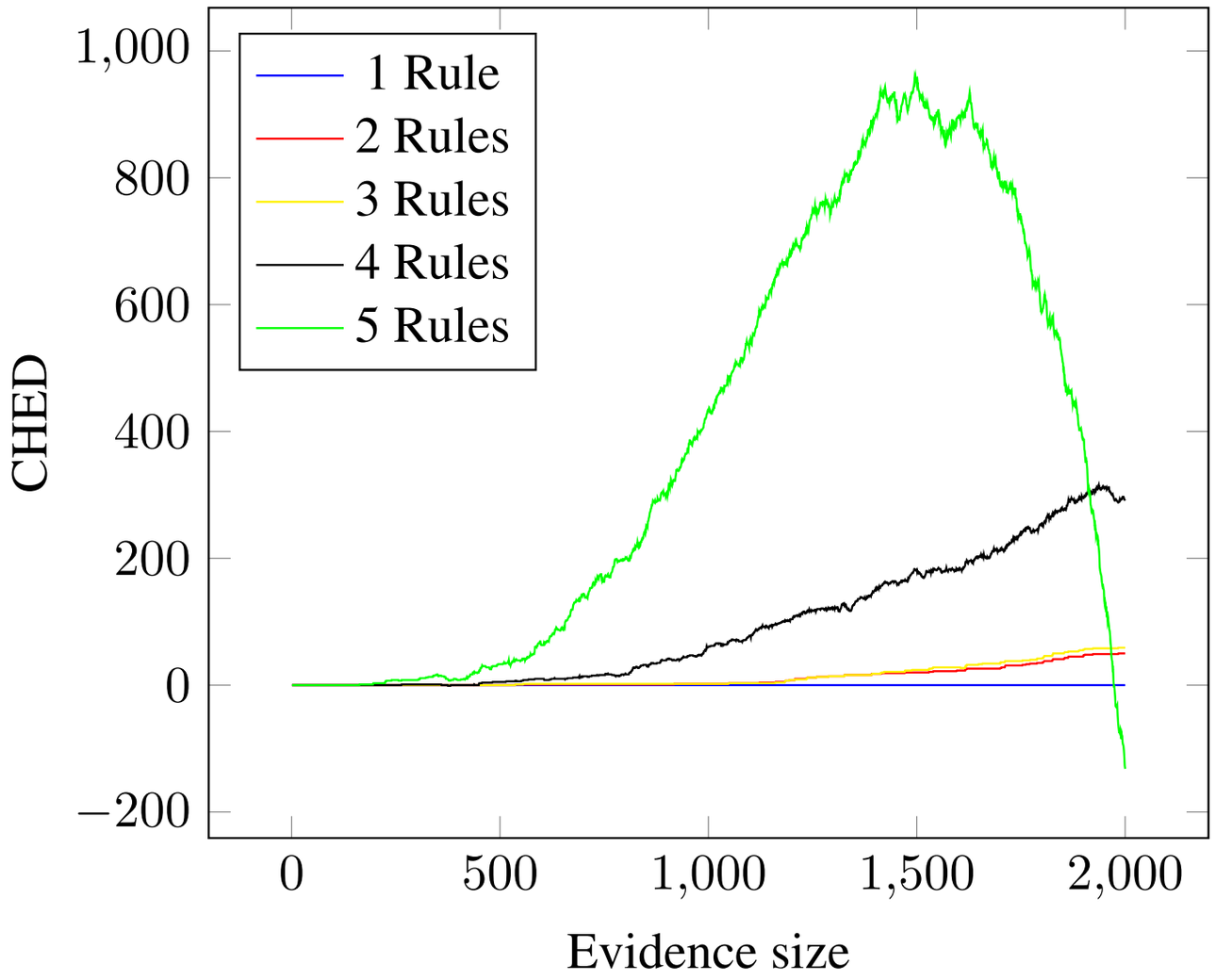
Note: set of entailed facts may not be consistent

Results


UWCSE, CHED (MLN - Method X)
$\cdot 10^4$
0.5
0
−0.5
−1
−1.5
CHED
PosLog
k-Entailment
STRiKE
PosLog W/O C
STRiKE W/O C
0
100
200
300
400
500
Evidence size

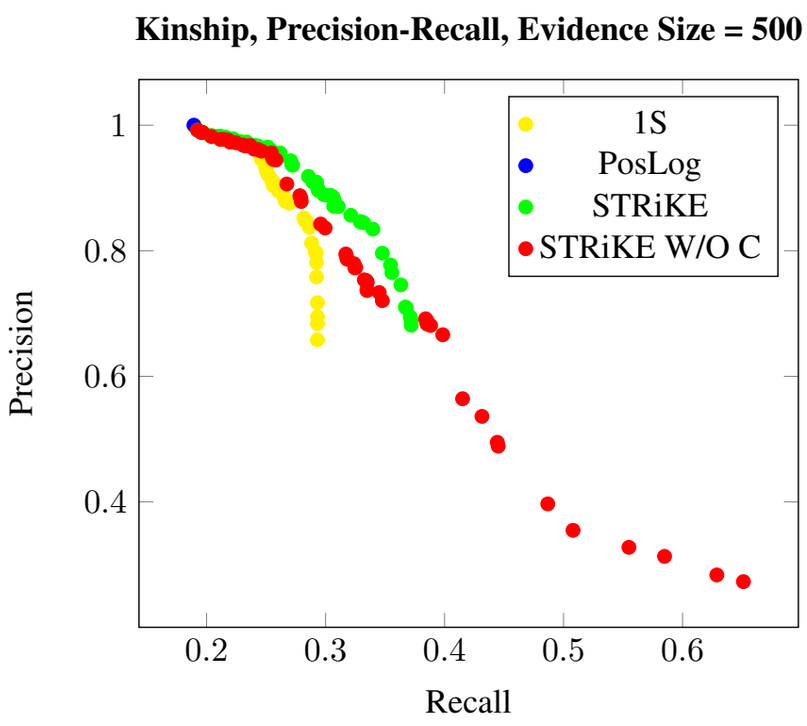

Proteins, CHED (MLN - Method X)
1,000
500
0
CHED
PosLog
k-Entailment
STRiKE
0
500
1,000
1,500
2,000
Evidence size
Precision

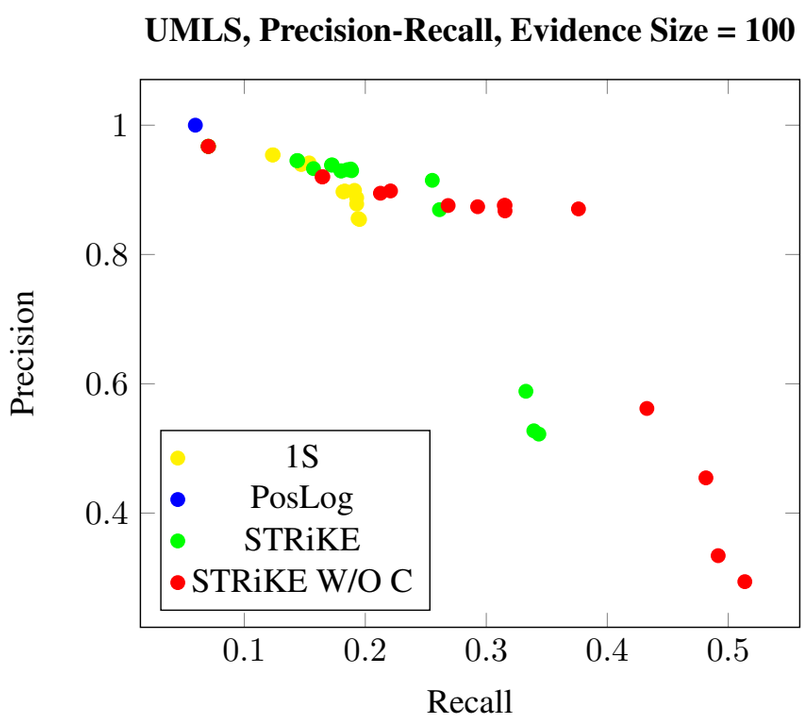
Results

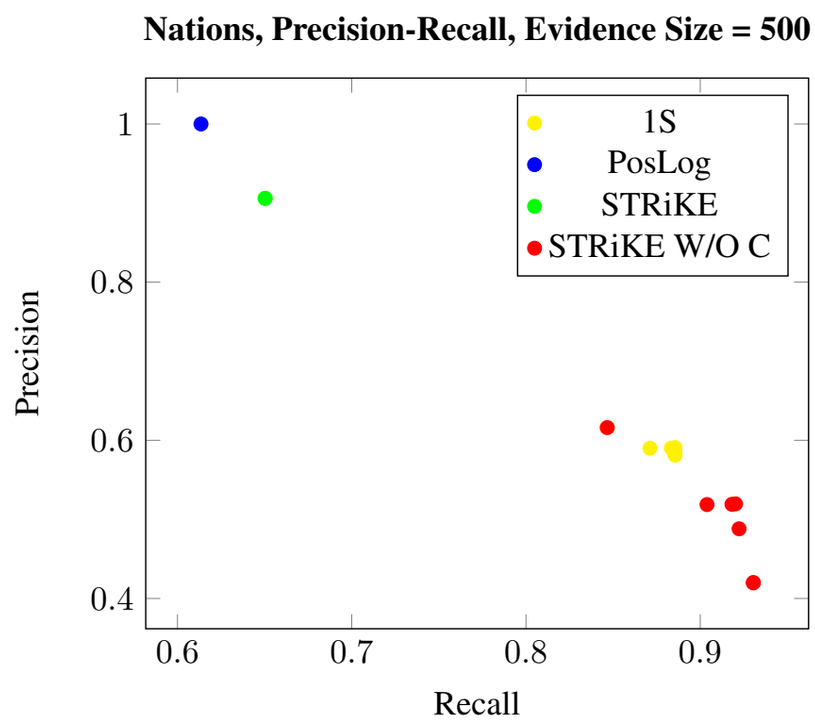


Yeast-proteins

Results







Note: a different rule learner was used for these experiments

Conclusions

Possibilistic logic (variants) provide us with a useful framework for encoding and exploiting learned relational knowledge

Compared to Markov logic networks, possibilistic logic theories are much easier/faster to learn, and perform better in most knowledge base completion setting (but they are far less suitable for marginal inference)

Best results were obtained with a variant based on k-entailment, to avoid the drowning issue

Different viewpoints are possible:

- Possibilistic logic theory encodes a probability distribution
- Rank rules according to how “safe” they are (PAC guarantees)