Statistical **R**elational **L**earning: A Tutorial

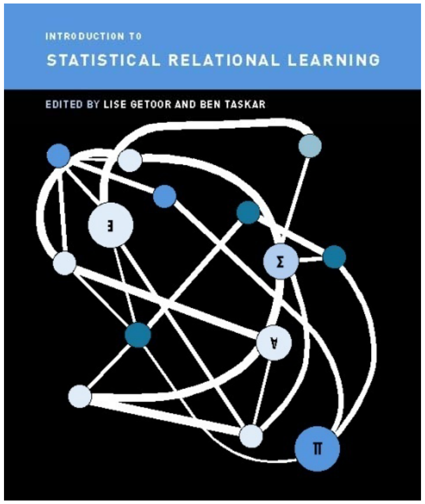
Lise Getoor

University of Maryland, College Park

Acknowledgements

- This tutorial is a synthesis of ideas of many individuals who have participated in various SRL events, workshops and classes (SRL00, SRL03, SRL04, SRL06, Dagstuhl05, Dagstuhl07, others)

Dagstuhl April 2007



- Hendrik Blockeel, Mark Craven, James Cussens, Bruce D'Ambrosio, Luc De Raedt, Tom Dietterich, Pedro Domingos, Saso Dzeroski, Peter Flach, Rob Holte, Manfred Jaeger, David Jensen, Kristian Kersting, Heikki Mannila, Andrew McCallum, Tom Mitchell, Ray Mooney, Stephen Muggleton, Kevin Murphy, Jen Neville, David Page, Avi Pfeffer, Claudia Perlich, David Poole, Foster Provost, Dan Roth, Stuart Russell, Taisuke Sato, Jude Shavlik, Ben Taskar, Lyle Ungar and many others

Roadmap

- SRL: What is it?
- SRL Tasks & Challenges
- 4 SRL Approaches
- Applications and Future directions

Why SRL?

- Traditional statistical machine learning approaches assume:
  - A random sample of homogeneous objects from single relation
- Traditional ILP/relational learning approaches assume:
  - No noise or uncertainty in data
- Real world data sets:
  - Multi-relational, heterogeneous and semi-structured
  - Noisy and uncertain
- Statistical Relational Learning:
  - newly emerging research area at the intersection of research in social network and link analysis, hypertext and web mining, graph mining, relational learning and inductive logic programming
- Sample Domains:
  - web data, bibliographic data, epidemiological data, communication data, customer networks, collaborative filtering, trust networks, biological data, natural language, vision

What is SRL?

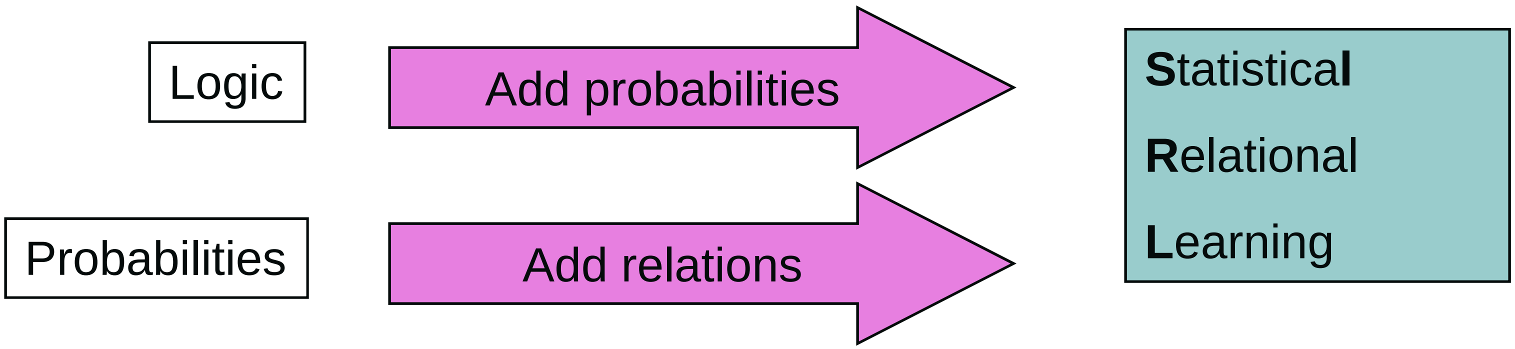
- Three views…

View 1: Alphabet Soup

LBN
PRISM
SRM
RDBN
RPM
CLP(BN)
SLR
BLOG
pRN
PER
PLL
RMN
SLP
HMRF
PRM
MLN
RNM
DAPER
SGLR
RDBN
BLP
RDN

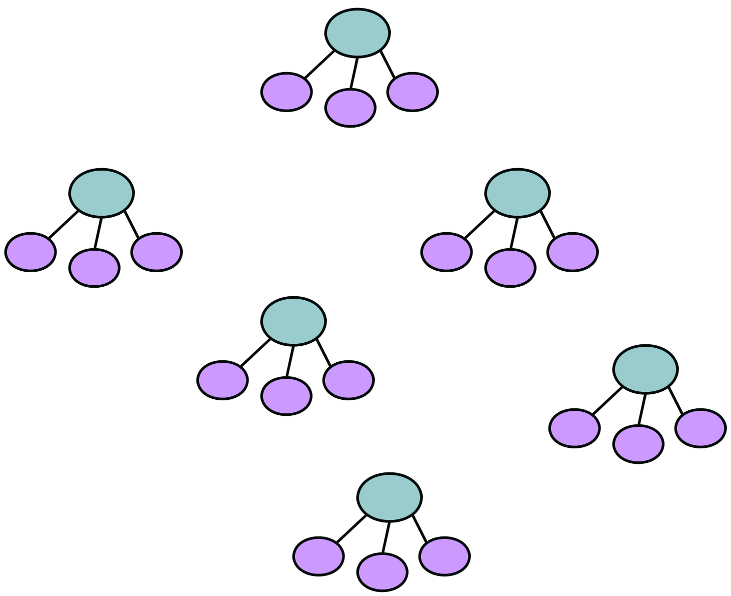
View 2: Representation Soup

- Hierarchical Bayesian Model + Relational Representation

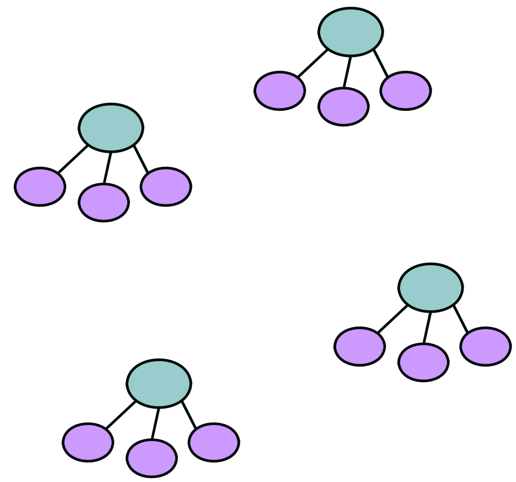


View 3: Data Soup

Training Data

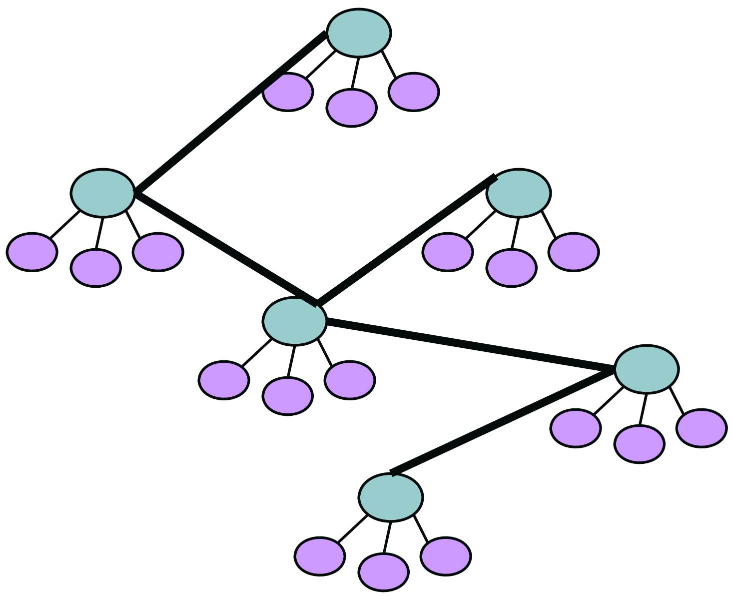


Test Data

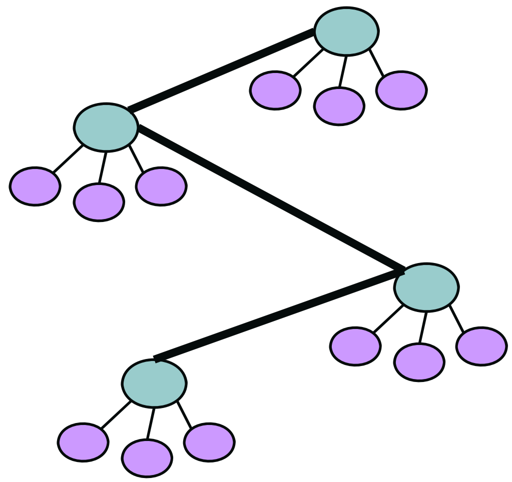


View 3: Data Soup

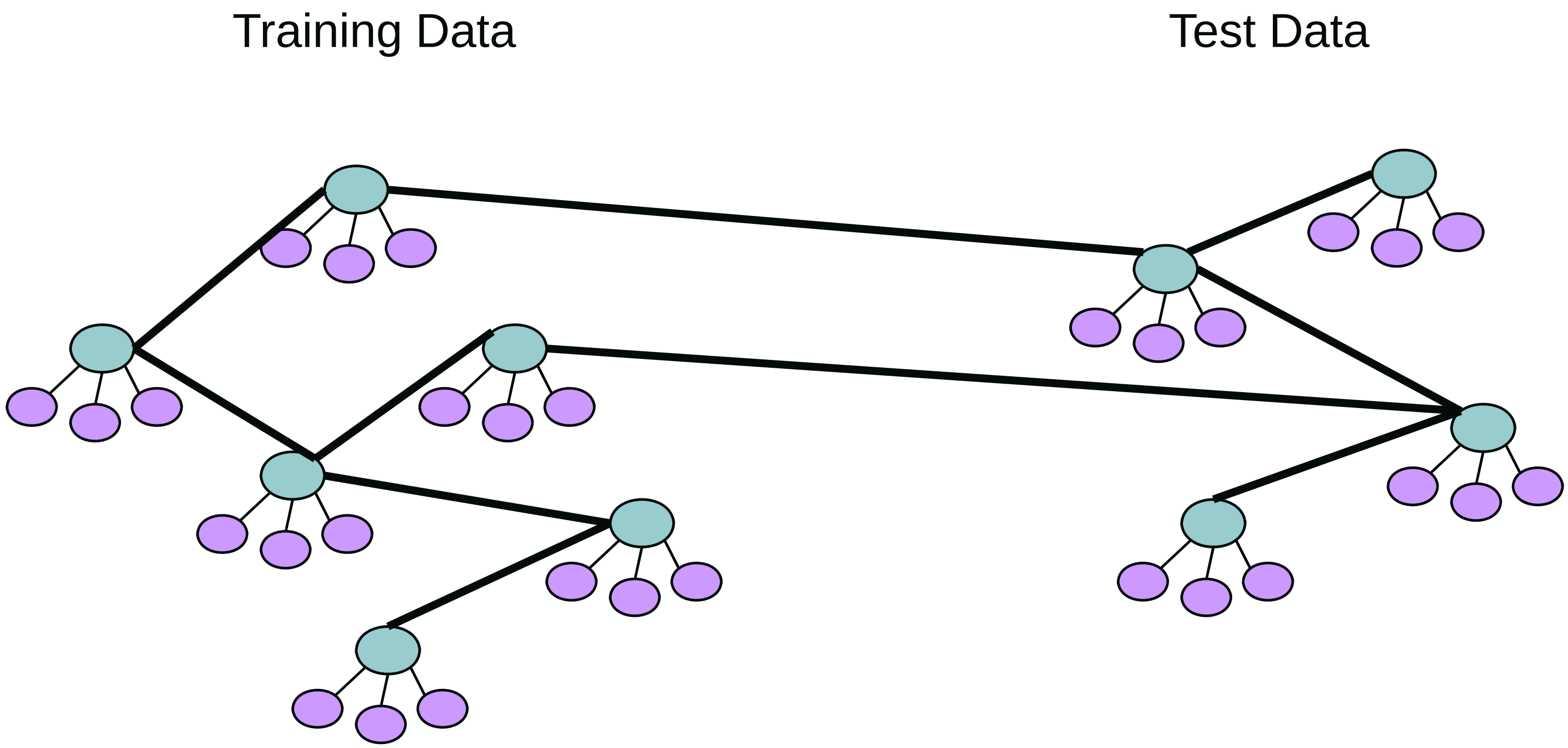
Training Data



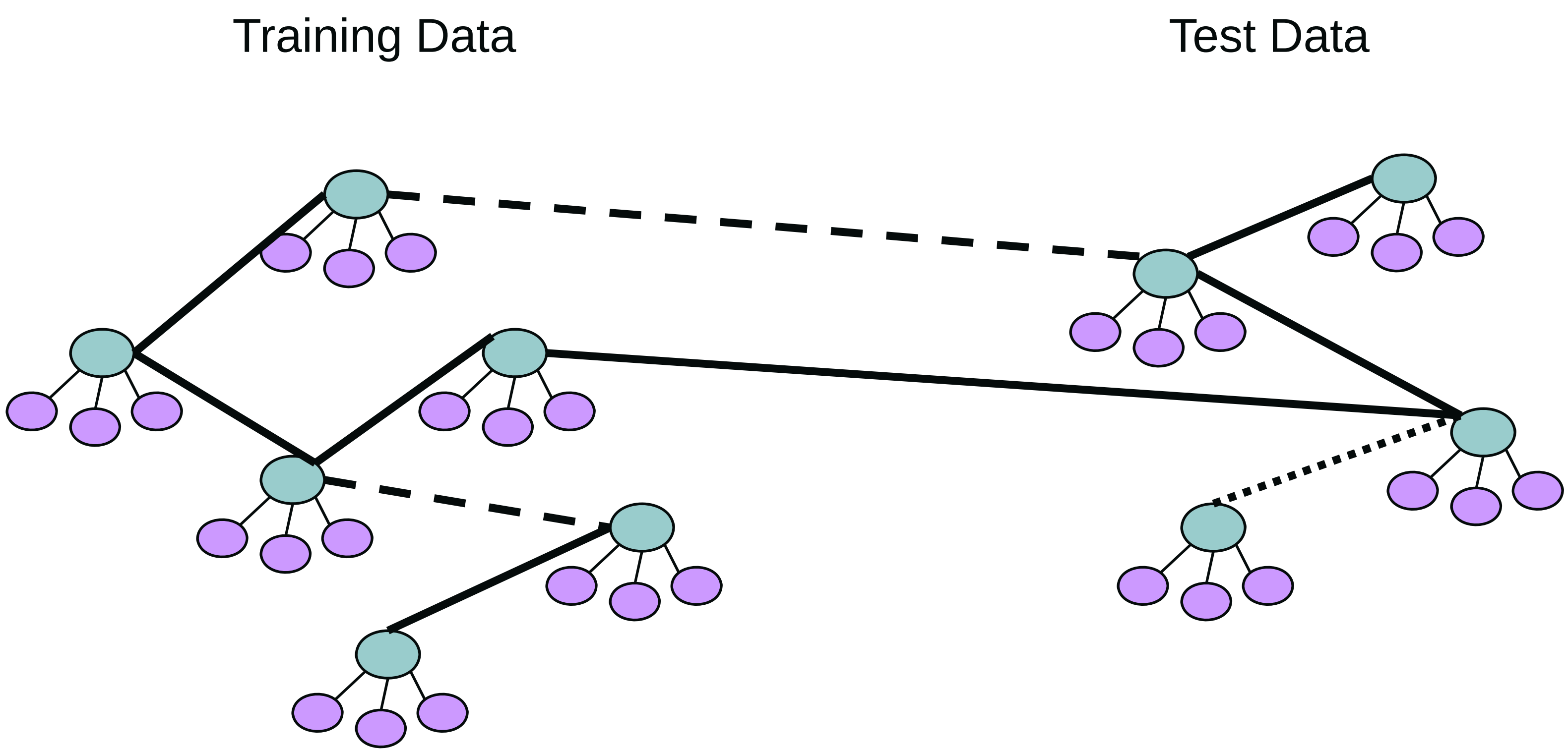
Test Data



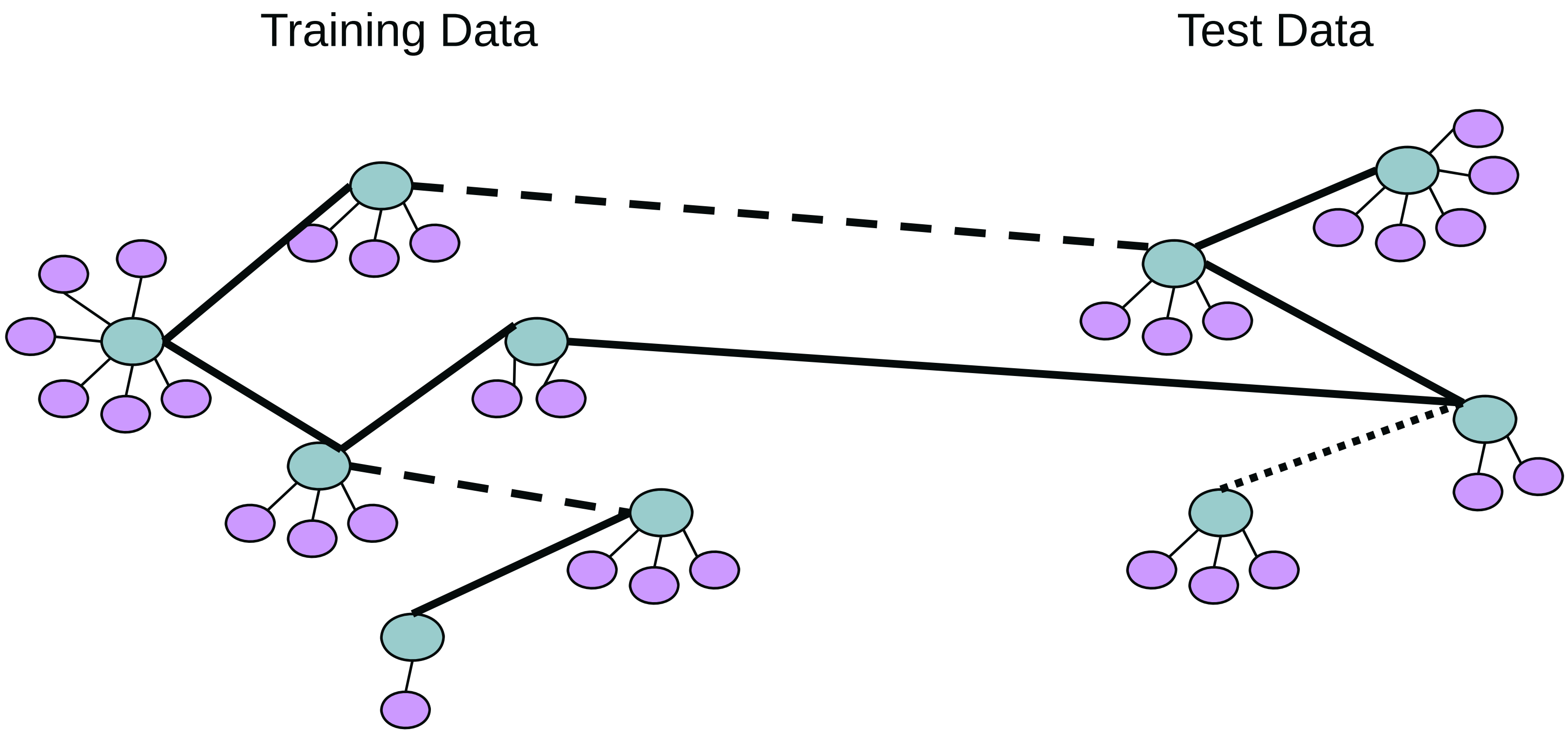
View 3: Data Soup



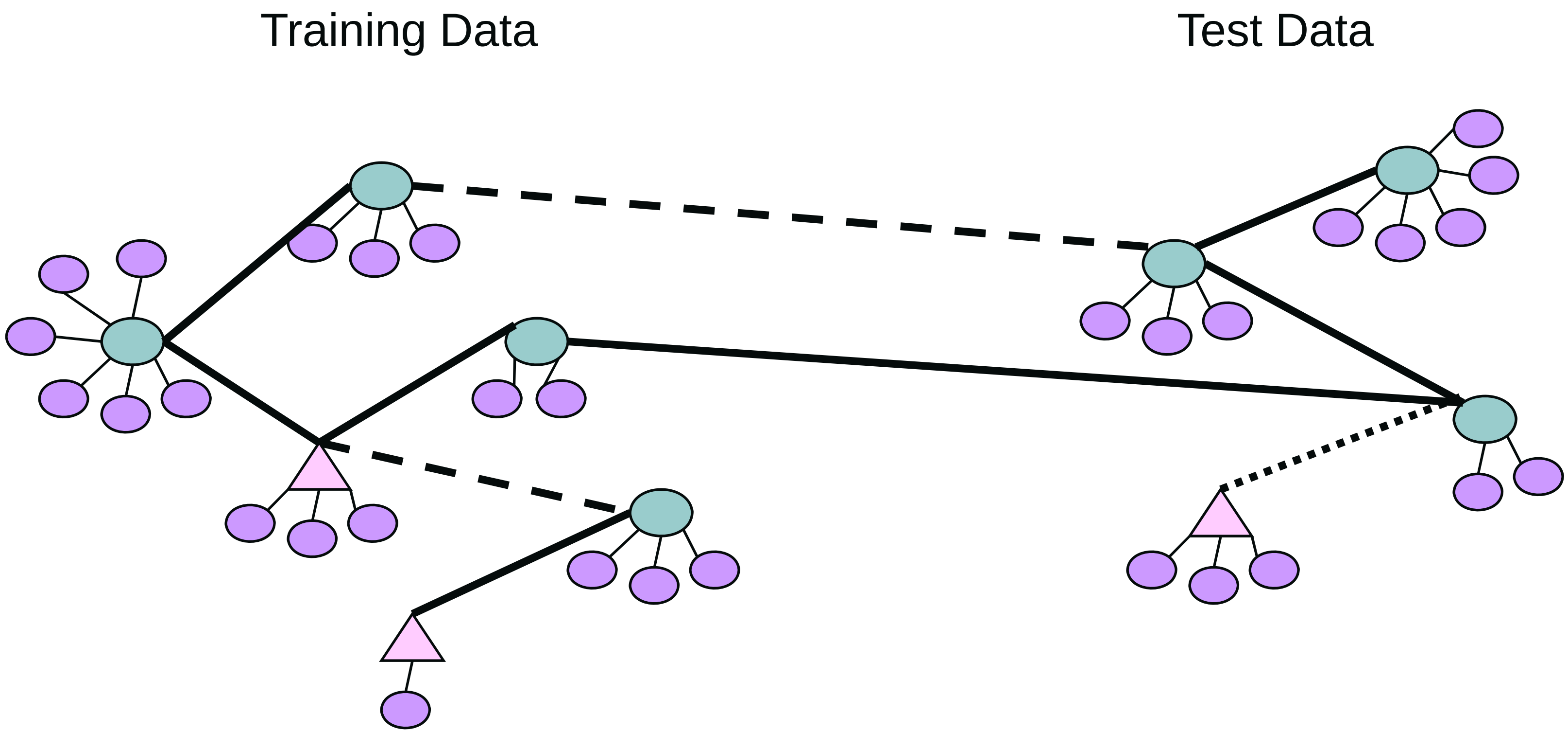
View 3: Data Soup



View 3: Data Soup



View 3: Data Soup



Goals

- By the end of this tutorial, hopefully, you will be:
  1. able to distinguish among different SRL tasks
  2. able to represent a problem in one of several SRL representations
  3. excited about SRL research problems and practical applications

Roadmap

- SRL: What is it?
- **SRL Tasks & Challenges**
- 4 SRL Approaches
- Applications and Future directions

SRL Tasks

- Tasks
  - Object Classification
  - Object Type Prediction
  - Link Type Prediction
  - Predicting Link Existence
  - Link Cardinality Estimation
  - Entity Resolution
  - Group Detection
  - Subgraph Discovery
  - Metadata Mining

Object Prediction

- **Object Classification**
  - Predicting the category of an object based on its attributes *and* its links *and* attributes of linked objects
  - e.g., predicting the topic of a paper based on the words used in the paper, the topics of papers it cites, the research interests of the author
- **Object Type Prediction**
  - Predicting the type of an object based on its attributes and its links and attributes of linked objects
  - e.g., predict the venue type of a publication (conference, journal, workshop) based on properties of the paper

Link Prediction

- **Link Classification**
  - Predicting type or purpose of link based on properties of the participating objects
  - e.g., predict whether a citation is to foundational work, background material, gratuitous PC reference
- **Predicting Link Existence**
  - Predicting whether a link exists between two objects
  - e.g. predicting whether a paper will cite another paper
- **Link Cardinality Estimation**
  - Predicting the number of links to an object or predicting the number of objects reached along a path from an object
  - e.g., predict the number of citations of a paper

More complex prediction tasks

- **Group Detection**
  - Predicting when a set of entities belong to the same group based on clustering both object attribute values and link structure
  - e.g., identifying research communities

- **Entity Resolution**
  - Predicting when a collection of objects are the same, based on their attributes *and* their links (aka: record linkage, identity uncertainty)
  - e.g., predicting when two citations are referring to the same paper.

- **Predicate Invention**
  - Induce a new general relation/link from existing links and paths
  - e.g., propose concept of advisor from co-author and financial support

- **Subgraph Identification, Metadata Mapping**

SRL Challenges

- Collective Classification
- Collective Consolidation
- Logical vs. Statistical dependencies
- Feature Construction – aggregation, selection
- Flexible and Decomposable Combining Rules
- Instances vs. Classes
- Effective Use of Labeled & Unlabeled Data
- Link Prediction
- Closed vs. Open World

Challenges common to any SRL approach!!
Bayesian Logic Programs, Markov Logic Networks, Probabilistic Relational Models, Relational Markov Networks, Relational Probability Trees, Stochastic Logic Programming to name a few

Roadmap

- SRL: What is it?
- SRL Tasks & Challenges
- **4 SRL Approaches**
- Applications and Future directions

Four SRL Approaches

- Directed Approaches
  - Rule-based Directed Models
  - Frame-based Directed Models
- Undirected Approaches
  - Frame-based Undirected Models
  - Rule-based Undirected Models
- Programming Language Approaches (oops, five!)

Emphasis in Different Approaches

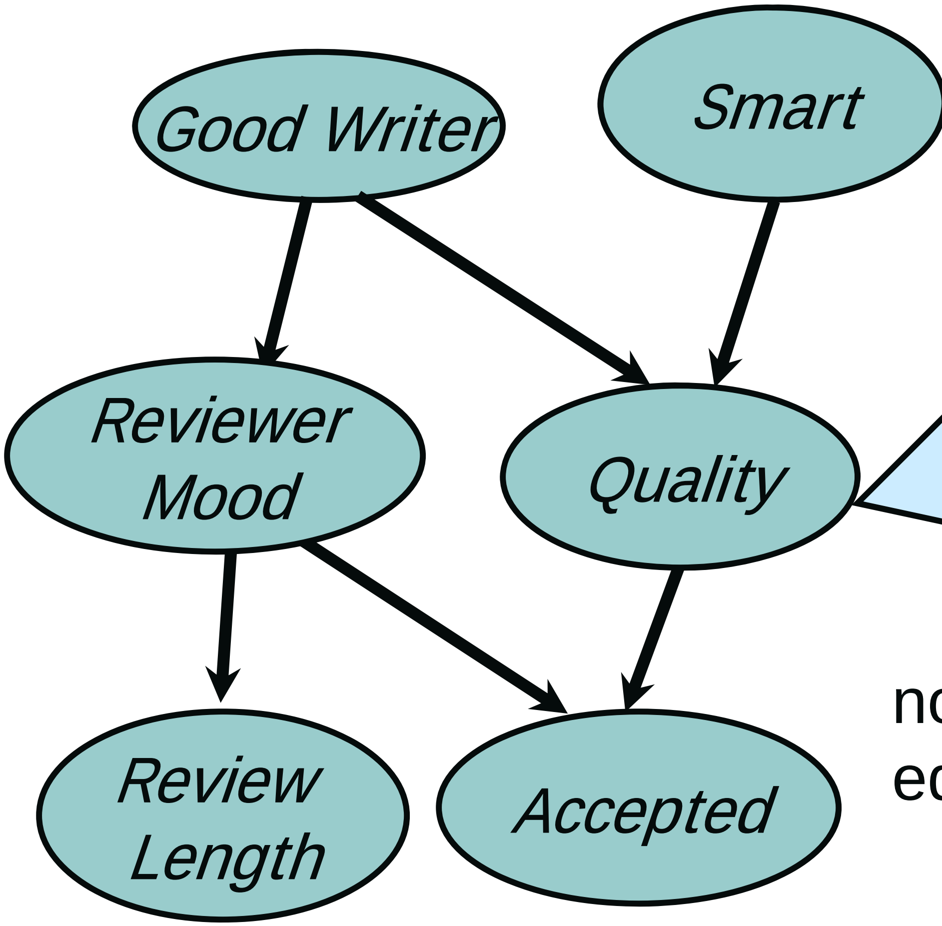
- Rule-based approaches focus on facts
  - what is true in the world?
  - what facts do other facts depend on?
- Frame-based approaches focus on objects and relationships
  - what types of objects are there, and how are they related to each other?
  - how does a property of an object depend on other properties (of the same or other objects)?
- Directed approaches focus on causal interactions
- Undirected approaches focus on symmetric, non-causal interactions
- Programming language approaches focus on processes
  - how is the world generated?
  - how does one event influence another event?

Four SRL Approaches

- Directed Approaches
  - **BN Tutorial**
  - Rule-based Directed Models
  - Frame-based Directed Models

- Undirected Approaches
  - Markov Network Tutorial
  - Rule-based Undirected Models
  - Frame-based Undirected Models

Bayesian Networks



conditional probability table (CPT)

W	S	$P(Q \mid W, S)$	
w	s	0.6	0.4
w	$\bar{s}$	0.3	0.7
$\bar{w}$	s	0.4	0.6
$\bar{w}$	$\bar{s}$	0.1	0.9

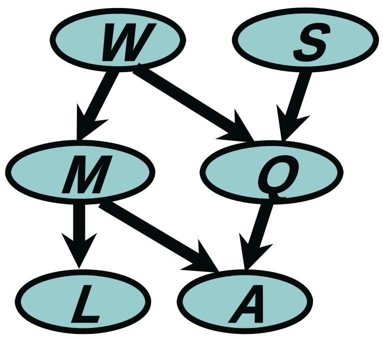
nodes = domain variables

edges = direct causal influence

Network structure encodes conditional independencies:

I(Review-Length , Good-Writer | Reviewer-Mood)

BN Semantics



conditional independencies in BN structure + local CPTs = full joint distribution over domain

$$P(\overline{w}, s, m, q, \bar{l}, a) =$$

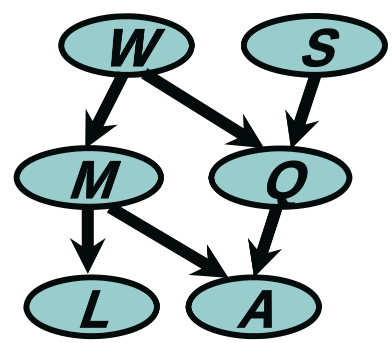
$$P(\overline{w})P(s)P(m \mid \overline{w})P(q \mid \overline{w}, s)P(\bar{l} \mid m)P(a \mid m, q)$$

- Compact & natural representation:
  - nodes $\leq k$ parents $\Rightarrow$ ***O(2^k n)*** vs. ***O(2^n)*** params
  - natural parameters

Reasoning in BNs

- Full joint distribution answers any query
  - *P(event | evidence)*

- Allows combination of different types of reasoning:
  - Causal: *P(Reviewer-Mood | Good-Writer)*
  - Evidential: *P(Reviewer-Mood | not Accepted)*
  - Intercausal: *P(Reviewer-Mood | not Accepted, Quality)*



Variable Elimination

- To compute $$P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$$

factors

$$\sum_{w,s,m,q,l} P(w)P(s)P(m \mid w)P(q \mid w,s)P(l \mid m)P(a \mid m,q)$$

mood	good writer	
pissy	false	1
pissy	true	0
good	false	0.7
good	true	0.3

A factor is a function from values of variables to positive real numbers

Variable Elimination

- To compute $$P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$$

$$\sum_{w,s,m,q} \sum_{l} P(w)P(s)P(m \mid w)P(q \mid w,s)P(l \mid m)P(a \mid m,q)$$

Variable Elimination

- To compute $$P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$$

$$\sum_{w,s,m,q} P(w)P(s)P(m \mid w)P(q \mid w,s)P(a \mid m,q)\sum_{l} P(l \mid m)$$

sum out l

Variable Elimination

- To compute $P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$

$$\sum_{w,s,m,q} P(w)P(s)P(m \mid w)P(q \mid w,s)P(a \mid m,q)\underset{\substack{\uparrow \\ \text{new factor}}}{f_1(m)}$$

Variable Elimination

- To compute $P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$

$$\sum_{s,m,q} P(s)P(a \mid m,q) f_1(m) \sum_{w} P(w)P(m \mid w)P(q \mid w,s)$$

multiply factors together then sum out w

Variable Elimination

- To compute $P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$

$$\sum_{s,m,q} P(s)P(a \mid m,q) f_1(m) f_2(m,q,s)$$

$$\uparrow$$

new factor

Variable Elimination

- To compute $P(a) = \sum_{w,s,m,q,l} P(w,s,m,q,l,a)$

$$P(a)$$

Other Inference Algorithms

- Exact
  - Junction Tree [Lauritzen & Spiegelhalter 88]
  - Cutset Conditioning [Pearl 87]
- Approximate
  - Loopy Belief Propagation [McEliece et al 98]
  - Likelihood Weighting [Shwe & Cooper 91]
  - Markov Chain Monte Carlo [eg MacKay 98]
    - Gibbs Sampling [Geman & Geman 84]
    - Metropolis-Hastings [Metropolis et al 53, Hastings 70]
  - Variational Methods [Jordan et al 98]

Learning BNs

	Parameters only	Structure and Parameters
Complete Data	Easy: counting	Structure search
Incomplete Data	EM [Dempster et al 77] or gradient descent [Russell et al 95]	Structural EM [Friedman 97]

See [Heckerman 98] for a general introduction

BN Parameter Estimation

- Assume known dependency structure *G*
- Goal: estimate BN parameters θ
  - entries in local probability models,

$$\theta_{x,u} = P(X = x \mid \mathrm{Pa}[X] = u)$$

- θ is good if it's likely to generate observed data.

$$l(\theta : D, G) = \log P(D \mid \theta, G)$$

- MLE Principle: Choose θ^* so as to maximize l
- Alternative: incorporate a prior

Learning With Complete Data

- Fully observed data: data consists of set of instances, each with a value for all BN variables

- With fully observed data, we can compute $N_{q,\overline{w},s}$ = number of instances with q, $\overline{w}$ and s
  - and similarly for other counts

- We then estimate

$$\theta_{q,\overline{w},s} = P(q \mid \overline{w}, s) = \frac{N_{q,\overline{w},s}}{N_{\overline{w},s}}$$

Dealing w/ missing values

- Can't compute $N_{q,\overline{w},s}$
- But can use Expectation Maximization (EM)
  1. Given parameter values, can compute expected counts: $E[N_{q,\overline{w},s}] = \sum_{\text{instances } i} P(q^i, \overline{w}^i, s^i \mid \text{evidence}^i)$

     this requires BN inference
  2. Given expected counts, estimate parameters:

$$\theta_{q,\overline{w},s} = P(q \mid \overline{w}, s) = \frac{E[N_{q,\overline{w},s}]}{E[N_{\overline{w},s}]}$$

- Begin with arbitrary parameter values
- Iterate these two steps
- Converges to local maximum of likelihood

Structure search

- Begin with an empty network
- Consider all neighbors reached by a search operator that are acyclic
  - add an edge
  - remove an edge
  - reverse an edge

- For each neighbor
  - compute ML parameter values θ_s^*
  - compute score(*s*) = $\log P(D \mid s, \theta_s^*) + \log P(s)$

- Choose the neighbor with the highest score
- Continue until reach a local maximum

Mini-BN Tutorial Summary

- Representation – probability distribution factored according to the BN DAG
- Inference – exact + approximate
- Learning – parameters + structure

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - **Rule-based Directed Models**
  - Frame-based Directed Models

- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Directed Rule-based Flavors

- Goldman & Charniak [93]
- Breese [92]
- Probabilistic Horn Abduction [Poole 93]
- Probabilistic Logic Programming [Ngo & Haddawy 96]
- Relational Bayesian Networks [Jaeger 97]
- Bayesian Logic Programs [Kersting & de Raedt 00]
- Stochastic Logic Programs [Muggleton 96]
- PRISM [Sato & Kameya 97]
- CLP(BN) [Costa et al. 03]
- Logical Bayesian Networks [Fierens et al 04, 05]
- etc.

Intuitive Approach

In logic programming,

```
accepted(P) :- author(P,A), famous(A).
```

means

For all P,A if A is the author of P and A is famous, then P is accepted

This is a categorical inference

But this may not be true in all cases

Fudge Factors

Use

accepted(P) :- author(P,A), famous(A). (0.6)

This means

For all P,A if A is the author of P and A is famous, then P is accepted with probability 0.6

But what does this mean when there are other possible causes of a paper being accepted?

e.g. accepted(P) :- high_quality(P). (0.8)

Intuitive Meaning

accepted(P) :- author(P,A), famous(A). (0.6)

means

For all P,A if A is the author of P and A is famous, then P is accepted with probability 0.6, *provided no other possible cause of the paper being accepted holds*

If more than one possible cause holds, a combining rule is needed to combine the probabilities

Meaning of Disjunction

In logic programming

```
accepted(P) :- author(P,A), famous(A).
accepted(P) :- high_quality(P).
```

means

For all P,A if A is the author of P and A is famous, or if P is high quality, then P is accepted

Probabilistic Disjunction

Now

accepted(P) :- author(P,A), famous(A). (0.6)

accepted(P) :- high_quality(P). (0.8)

means

For all P,A, if (A is the author of P and A is famous *successfully* cause P to be accepted) or (P is high quality *successfully* causes P to be accepted), then P is accepted.

If A is the author of P and A is famous, they successfully cause P to be accepted with probability 0.6.

If P is high quality, it successfully causes P to be accepted with probability 0.8.

- All causes act independently to produce effect (*causal independence)*
- Leak probability: effect may happen with no cause
  - e.g. accepted(P). (0.1)

Computing Probabilities

- What is *P*(**accepted(p1)**) given that Alice is an author and Alice is famous, and that the paper is high quality, but no other possible cause is true?

$$
\begin{aligned}
P &= \text{P(at least one true cause succeeds)} \\
&= 1 - P(\text{all true possible causes fail}) \\
&= 1 - \prod_{\text{true possible causes } i} (1 - p_{\text{success}}(i)) \\
&= 1 - (1-0.6)(1-0.8)(1-\underbrace{0.1}_{\boxed{\text{leak}}}) = 0.928
\end{aligned}
$$

Combination Rules

- Other combination rules are possible
- e.g., max

$$P(\text{effect}) = \max_{\text{true possible causes } i} p_{\text{success}}(i)$$

- In our case,

  P(**accepted(p1)**) = max {0.6,0.8,0.1} = 0.8
- Harder to interpret in terms of logic program

KBMC

- Knowledge-Based Model Construction (KBMC) [Wellman et al. 92, Ngo & Haddawy 95]
- Method for computing more complex probabilities
- Construct a Bayesian network, given a query *Q* and evidence *E*
  - query and evidence are sets of ground atoms, i.e., predicates with no variable symbols
    - e.g. **author(p1,alice)**
- Construct network by searching for possible proofs of the query and the variables
- Use standard BN inference techniques on constructed network

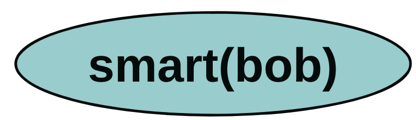
KBMC Example

smart(alice). (0.8)
smart(bob). (0.9)
author(p1,alice). (0.7)
author(p1,bob). (0.3)
high_quality(P) :- author(P,A), smart(A). (0.5)
high_quality(P). (0.1)
accepted(P) :- high_quality(P). (0.9)

Query is **accepted(p1).**
Evidence is **smart(bob).**

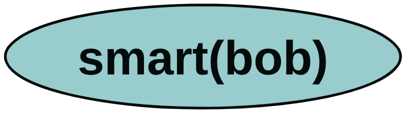
Backward Chaining

Start with evidence variable **smart(bob)**



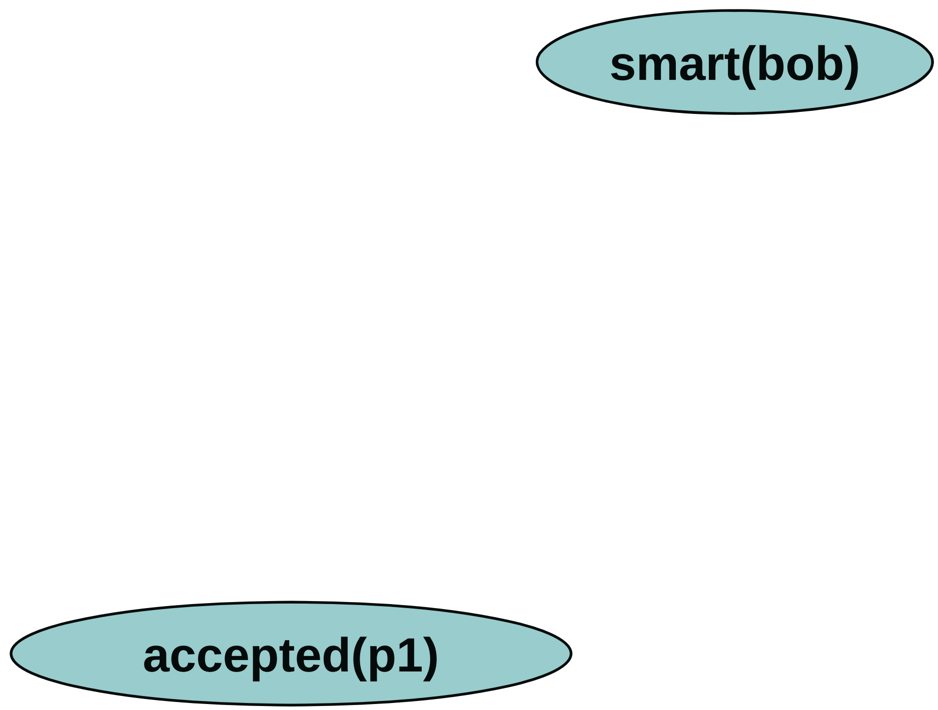
Backward Chaining

Rule for **smart(bob)** has no antecedents – stop backward chaining



Backward Chaining

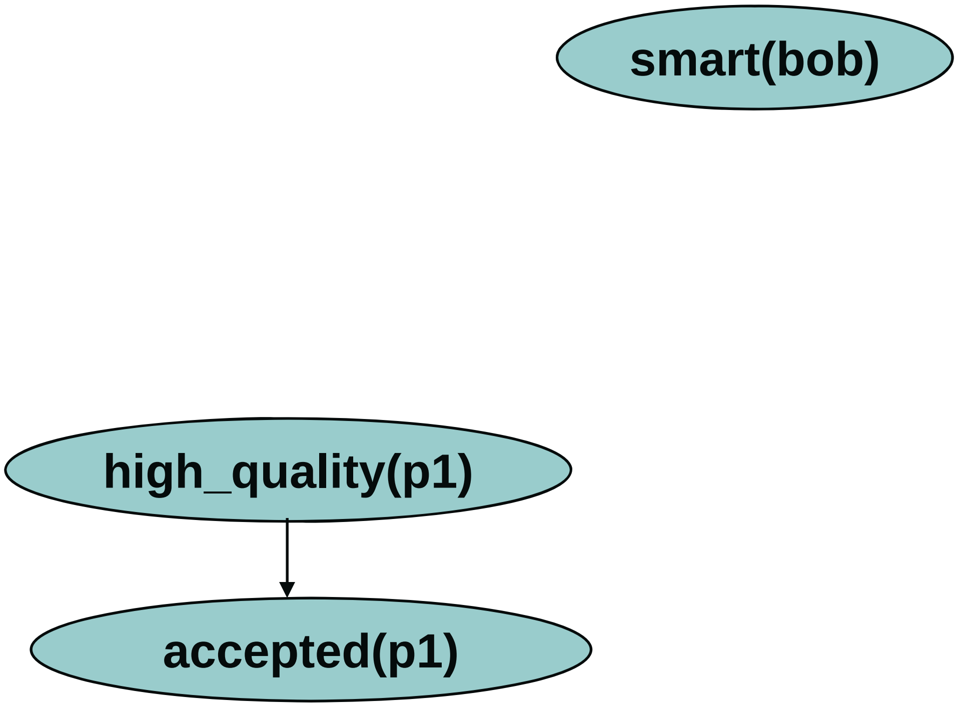
Begin with query variable **accepted(p1)**



Backward Chaining

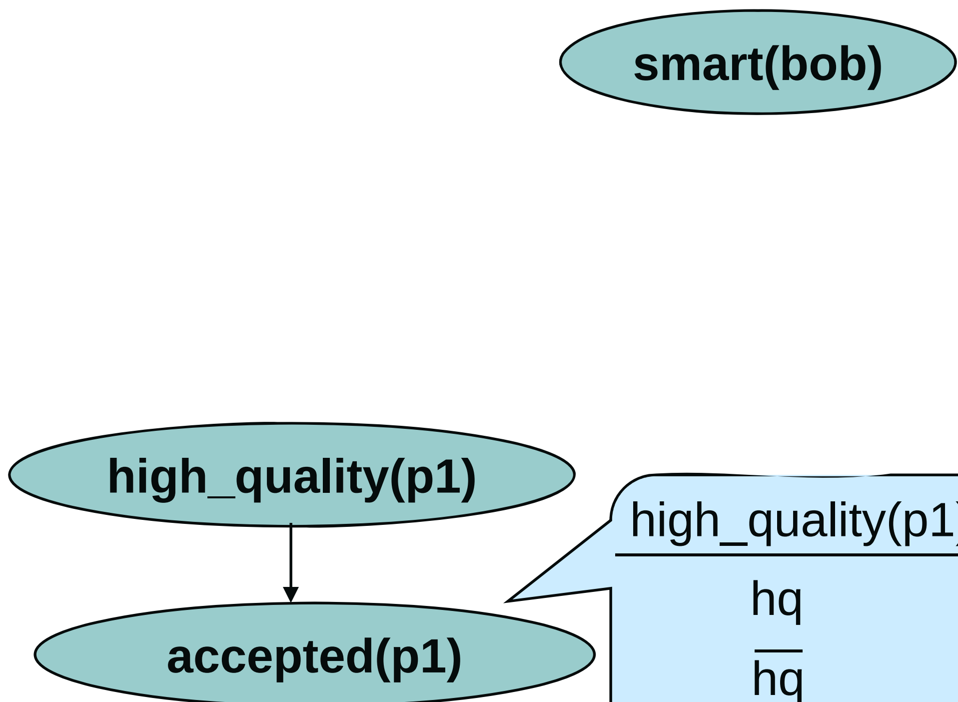
Rule for **accepted(p1)** has antecedent **high_quality(p1)**

add **high_quality(p1)** to network, and make parent of **accepted(p1)**



Backward Chaining

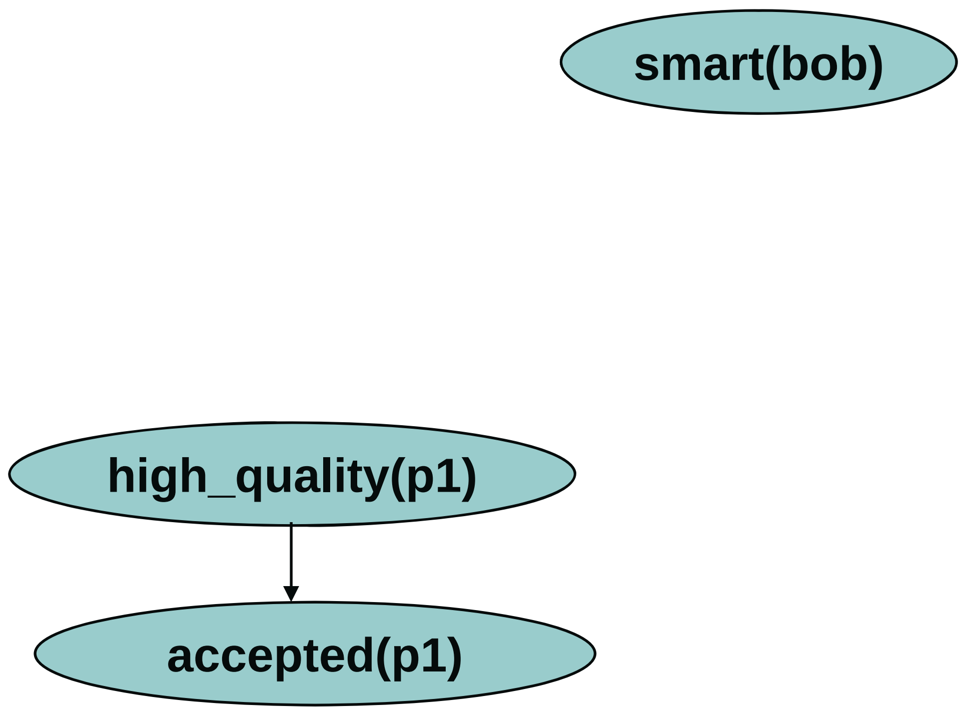
All of **accepted(p1)**'s parents have been found – create its conditional probability table (CPT)



high_quality(p1)	accepted(p1)	
hq	0.9	0.1
$\overline{hq}$	0	1

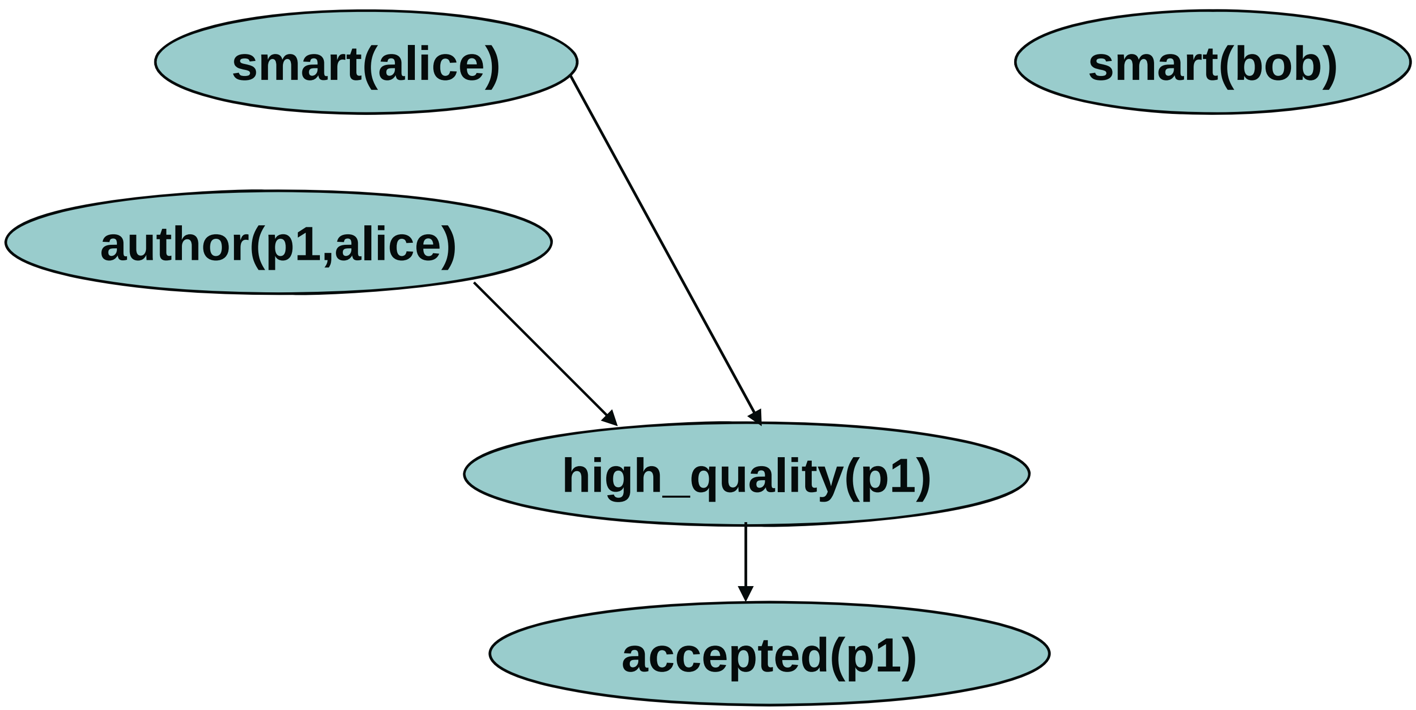
Backward Chaining

high_quality(p1) :- author(p1,A), smart(A) has two groundings: **A=alice** and **A=bob**



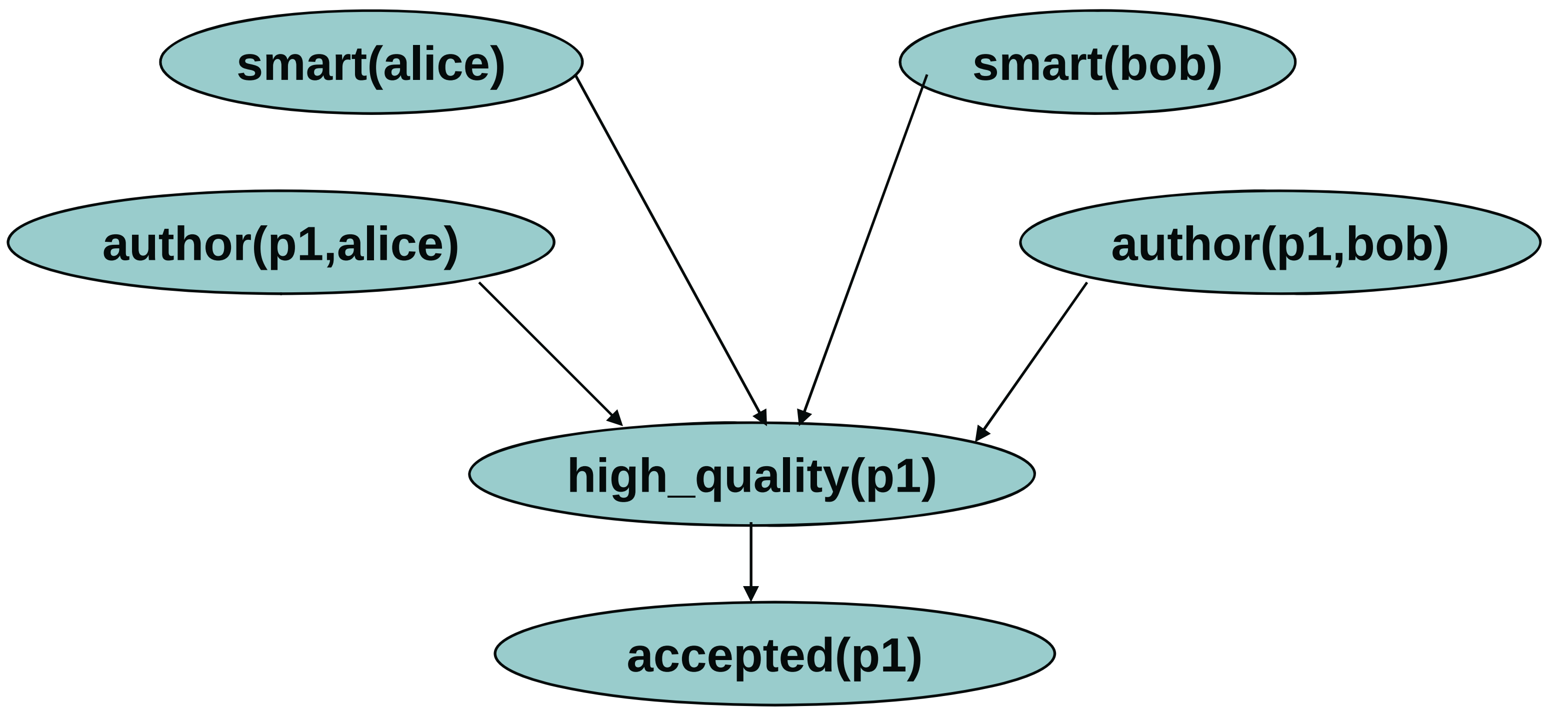
Backward Chaining

For grounding **A=alice,** add **author(p1,alice)** and **smart(alice**) to network, and make parents of **high_quality(p1)**



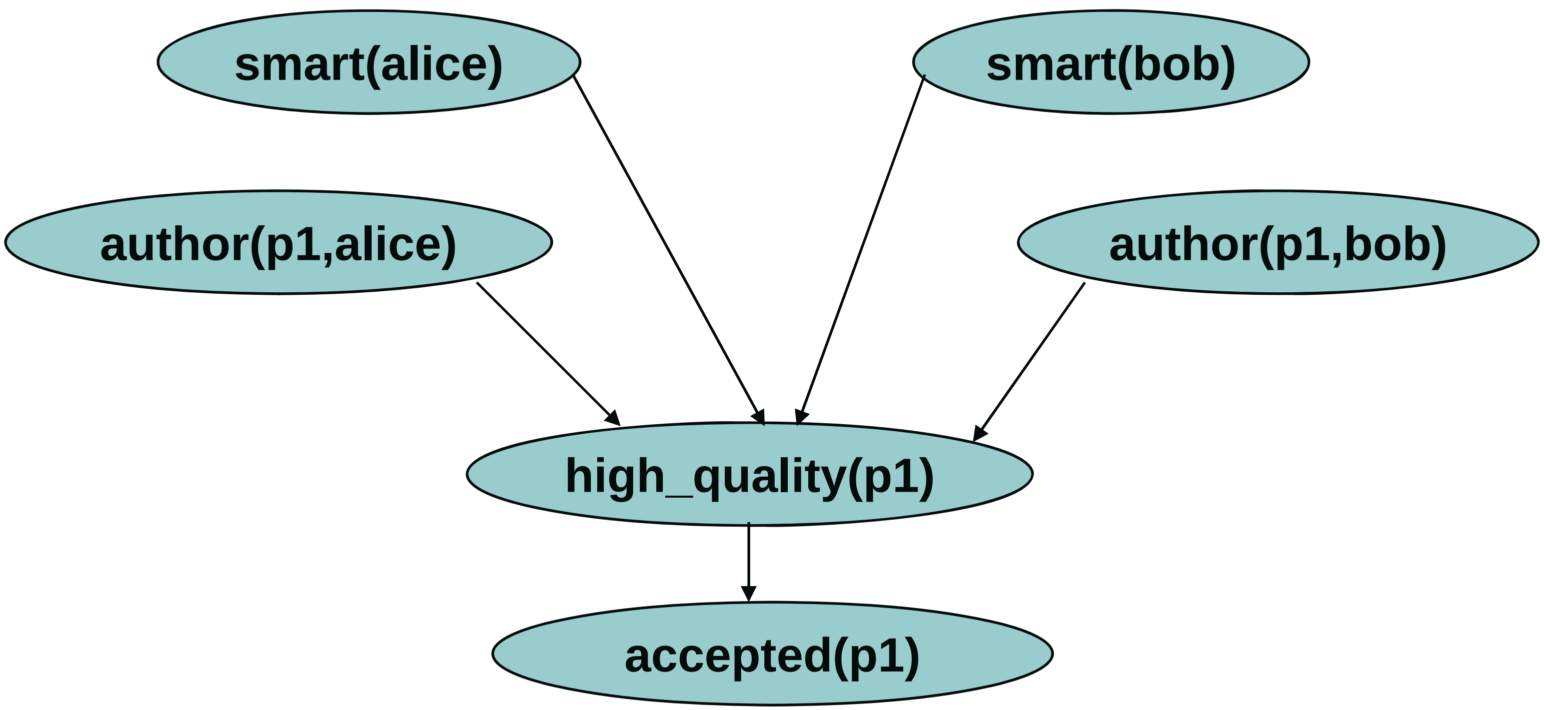
Backward Chaining

For grounding **A=bob**, add **author(p1,bob)** to network. **smart(bob)** is already in network. Make both parents of **high_quality(p1)**



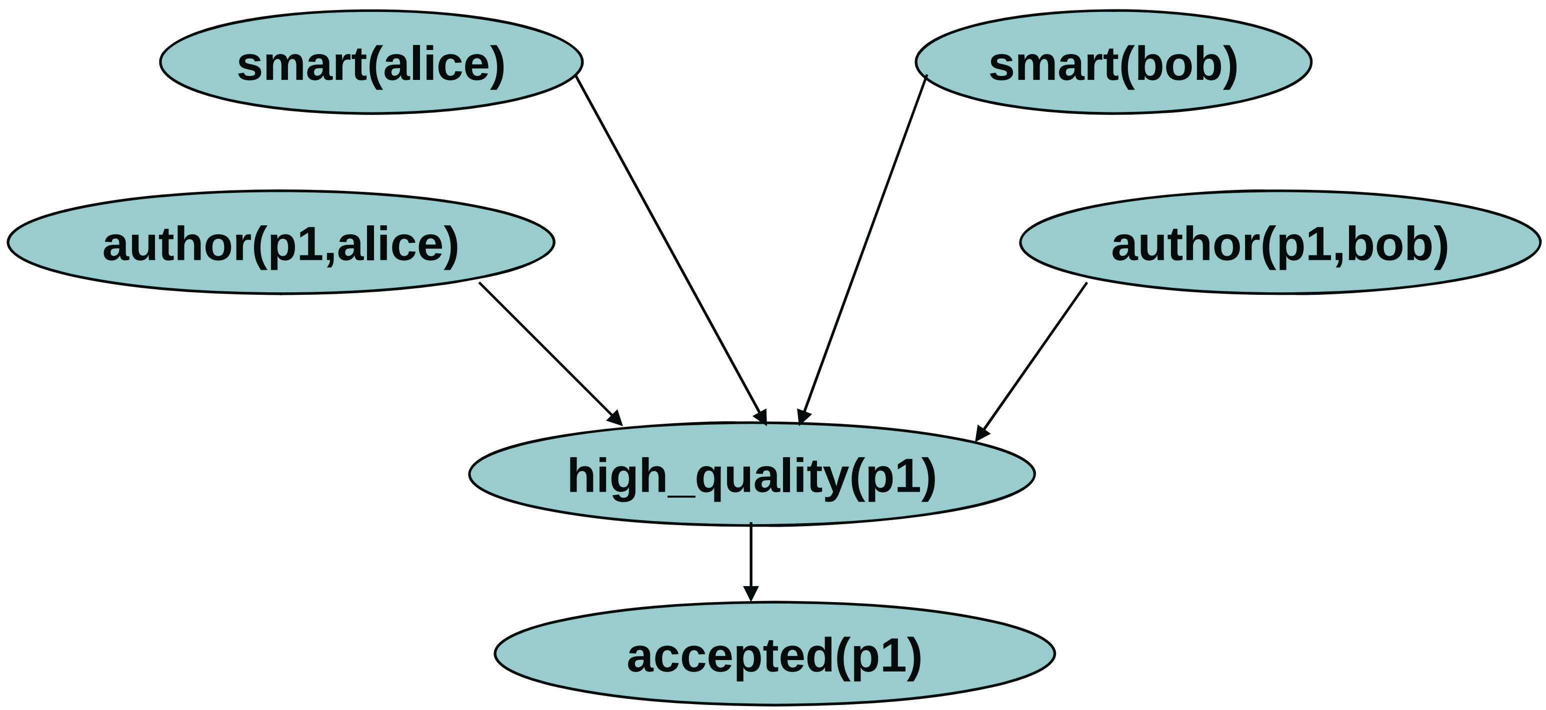
Backward Chaining

Create CPT for **high_quality(p1)** – make noisy-or



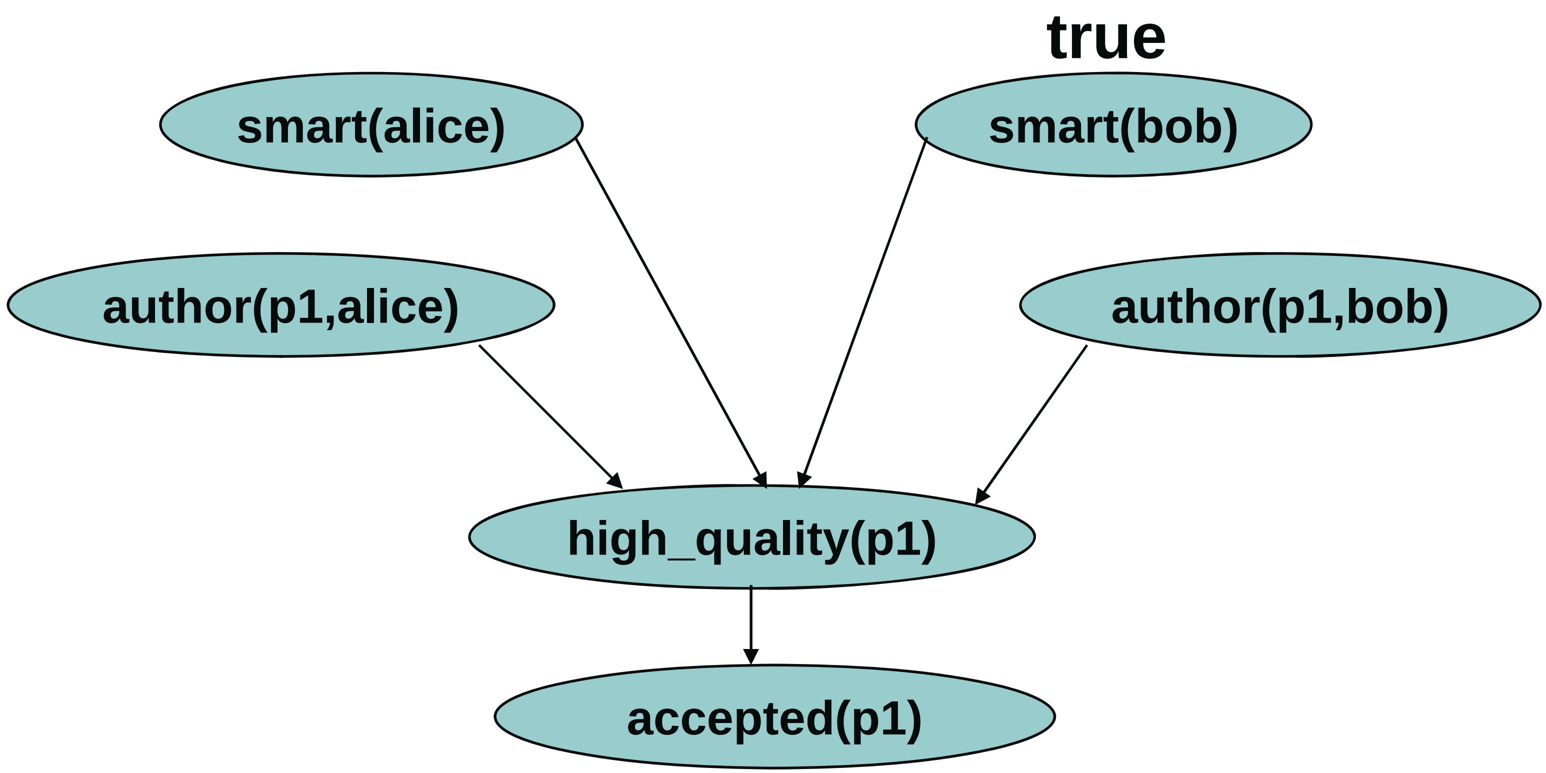
Backward Chaining

author(p1,alice), smart(alice) and **author(p1,bob)** have no antecedents – stop backward chaining



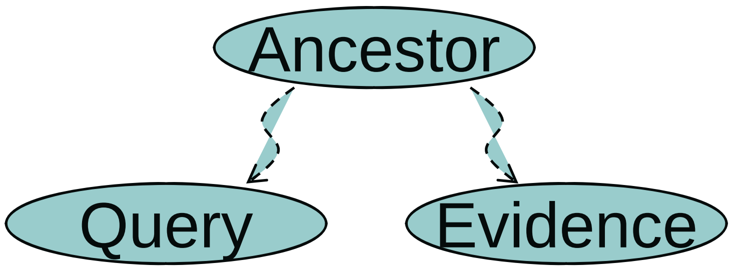
Backward Chaining

- assert evidence **smart(bob) = true**, and compute ***P*(accepted(p1) | smart(bob) = true)**



Backward Chaining on Both Query and Evidence

- Necessary, if query and evidence have common ancestor



- Sufficient. *P*(Query | Evidence) can be computed using only ancestors of query and evidence nodes
  - unobserved descendants are irrelevant

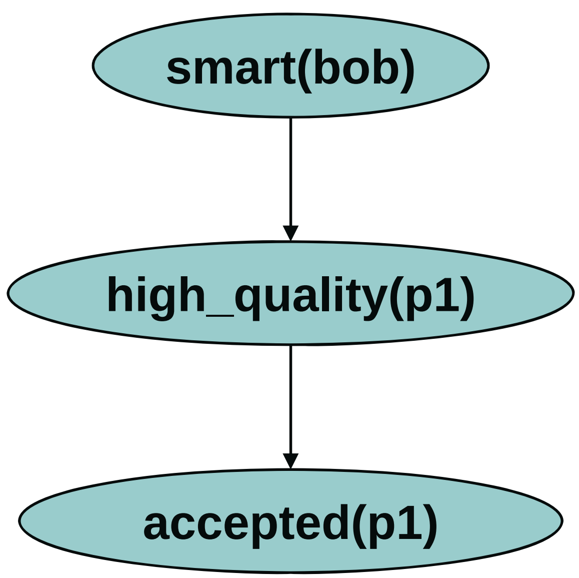
The Role of Context

- Context is deterministic knowledge known prior to the network being constructed
  - May be defined by its own logic program
  - Is not a random variable in the BN
- Used to determine structure of the constructed BN
  - If a context predicate P appears in the body of a rule R, only backward chain on R if P is true

Context example

Suppose **author(P,A)** is a context predicate, **author(p1,bob)** is true, and **author(p1,alice)** cannot be proven from deterministic KB (and is therefore false by assumption)

Network is



No **author(p1,bob)** node because it is a context predicate

No **smart(alice)** node because **author(p1,alice)** is false

Semantics

- Assumption: no cycles in resulting BN
  - If there are cycles, cannot interpret BN as definition of joint probability distribution
- Assuming BN construction process terminates, conditional probability of any query given any evidence is defined by the BN.
- Somewhat unsatisfying because
  1. meaning of program is query dependent (depends on constructed BN)
  2. meaning is not stated declaratively in terms of program but in terms of constructed network instead

Disadvantages of Approach

- Up until now, ground logical atoms have been random variables ranging over T,F
  - cumbersome to have a different random variable for **lead_author(p1,alice), lead_author(p1,bob)** and all possible values of **lead_author(p1,A)**
  - worse, since **lead_author(p1,alice)** and **lead_author(p1,bob)** are different random variables, it is possible for both to be true at the same time

Bayesian Logic Programs [Kersting and de Raedt]

- Now, ground atoms are random variables with any range (not necessarily Boolean)
  - now **quality** is a random variable, with values **high, medium, low**

- Any probabilistic relationship is allowed
  - expressed in CPT

- Semantics of program given once and for all
  - not query dependent

Meaning of Rules in BLPs

accepted(P) :- quality(P).

means

"For all P, *if* **quality(P)** is a random variable, then **accepted(P)** is a random variable"

Associated with this rule is a conditional probability table (CPT) that specifies the probability distribution over **accepted(P)** for any possible value of **quality(P)**

Combining Rules for BLPs

accepted(P) :- quality(P).

accepted(P) :- author(P,A), fame(A).

- Before, combining rules combined individual probabilities with each other
  - noisy-or and max rules easy to interpret
- Now, combining rules combine entire CPTs

Semantics of BLPs

- Random variables are all ground atoms that have finite proofs in logic programs
  - assumes acyclicity
  - assumes no function symbols
- Can construct BN over all random variables
  - parents derived from rules
  - CPTs derived using combining rules
- Semantics of BLP: joint probability distribution over all random variables
  - does not depend on query
- Inference in BLP by KBMC

An Issue

- How to specify uncertainty over single-valued relations?
- Approach 1: make **lead_author(P)** a random variable taking values **bob**, **alice** etc.
  - we can’t say **accepted(P) :- lead_author(P), famous(A)** because **A** does not appear in the rule head or in a previous term in the body
- Approach 2: make **lead_author(P,A)** a random variable with values **true, false**
  - we run into the same problems as with the intuitive approach (may have zero or many lead authors)
- Approach 3: make **lead_author** a function
  - say **accepted(P) :- famous(lead_author(P))**
  - need to specify how to deal with function symbols and uncertainty over them

First-Order Variable Elimination

- [Poole 03, Braz et al 05]
- Generalization of variable elimination to first order domains
- Reasons directly about first-order variables, instead of at the ground level
- Assumes that the size of the population for each type of entity is known

Learning Rule Parameters

- [Koller & Pfeffer 97, Sato & Kameya 01]
- Problem definition:
  - Given a skeleton rule base consisting of rules without uncertainty parameters
  - and a set of instances, each with
    - a set of context predicates
    - observations about some random variables
  - Goal: learn parameter values for the rules that maximize the likelihood of the data

Basic Approach

1. Construct a network BN^i for each instance *i* using KBMC, backward chaining on all the observed variables
2. Expectation Maximization (EM)
   - exploit parameter sharing

Parameter Sharing

- In BNs, all random variables have distinct CPTs
  - only share parameters between different instances, not different random variables
- In logical approaches, an instance may contain many objects of the same kind
  - multiple papers, multiple authors, multiple citations
- Parameters are shared within instances
  - same parameters used across different papers, authors, citations
- Parameter sharing allows faster learning, and learning from a single instance

Rule Parameters & CPT Entries

- In principle, combining rules produce complicated relationship between model parameters and CPT entries
- With a *decomposable* combining rule, each node is derived from a single rule
  - Most natural combining rules are decomposable
    - e.g. noisy-or decomposes into set of *and*s followed by *or*

Parameters and Counts

- Each time a node is derived from a rule *r*, it provides one experiment to learn about the parameters associated with *r*

- Each such node should therefore make a separate contribution to the count for those parameters

- $\theta^{r}_{x,u}$: the parameter associated with $P(X=x|\text{Parents}[X]=u)$ when rule *r* applies

- $N^{r}_{x,u}$: the number of times a node has value *x* and its parents have value *u* when rule *r* applies

EM With Parameter Sharing

- Given parameter values, compute expected counts:

$$E[N^r_{x,u}] = \sum_{\text{instances } i} \sum_{X} P(X = x, \text{Parents}[X] = u \mid \text{evidence}^i)$$

where the inner sum is over all nodes derived from rule *r* in BN^i

- Given expected counts, estimate:

$$\theta^r_{x,u} = \frac{E[N^r_{x,u}]}{E[N^r_u]}$$

- Iterate these two steps

Learning Rule Structure

- [Kersting and De Raedt 02]
- Problem definition:
  - Given a set of instances, each with
    - context predicates
    - observations about some random variables
  - Goal: learn
    - a skeleton rule base consisting of rules and parameter values for the rules
- Generalizes BN structure learning
  - define legal models
  - scoring function same as for BN
  - define search operators

Legal Models

- Hypothesis space consists of all rule sets using given predicates, together with parameter values
- A legal hypothesis:
  - is logically valid: rule set does not draw false conclusions for any data cases
  - the constructed BN is acyclic for every instance

Search operators

- Add a constant-free atom to the body of a single clause
- Remove a constant-free atom from the body of a single clause

accepted(P) :- author(P,A).
accepted(P) :- quality(P).

delete

accepted(P).
accepted(P) :- quality(P).

add

accepted(P) :- author(P,A), famous(A).
accepted(P) :- quality(P).

Summary: Directed Rule-based Approaches

- Provide an intuitive way to describe how one fact depends on other facts
- Incorporate relationships between entities
- Generalizes to many different situations
  - Constructed BN for a domain depends on which objects exist and what the known relationships are between them (context)
- Inference at the ground level via KBMC
  - or lifted inference via FOVE
- Both parameters and structure are learnable

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**

- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Frame-based Approaches

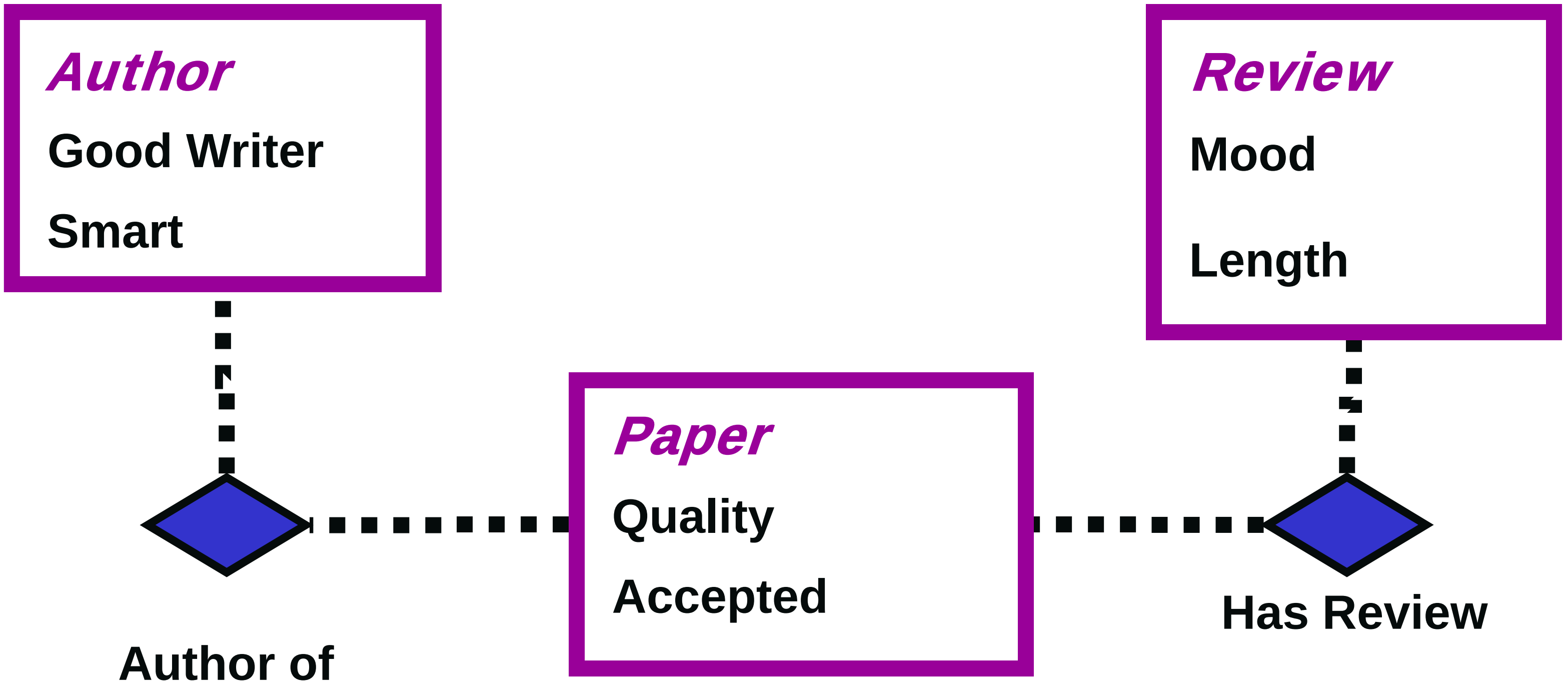
- Probabilistic Relational Models (PRMs)
  - Representation & Inference [Koller & Pfeffer 98, Pfeffer, Koller, Milch &Takusagawa 99, Pfeffer 00]
  - Learning [Friedman et al. 99, Getoor, Friedman, Koller & Taskar 01 & 02, Getoor 01]

- Probabilistic Entity Relation Models (PERs)
  - Representation [Heckerman, Meek & Koller 04]

Four SRL Approaches

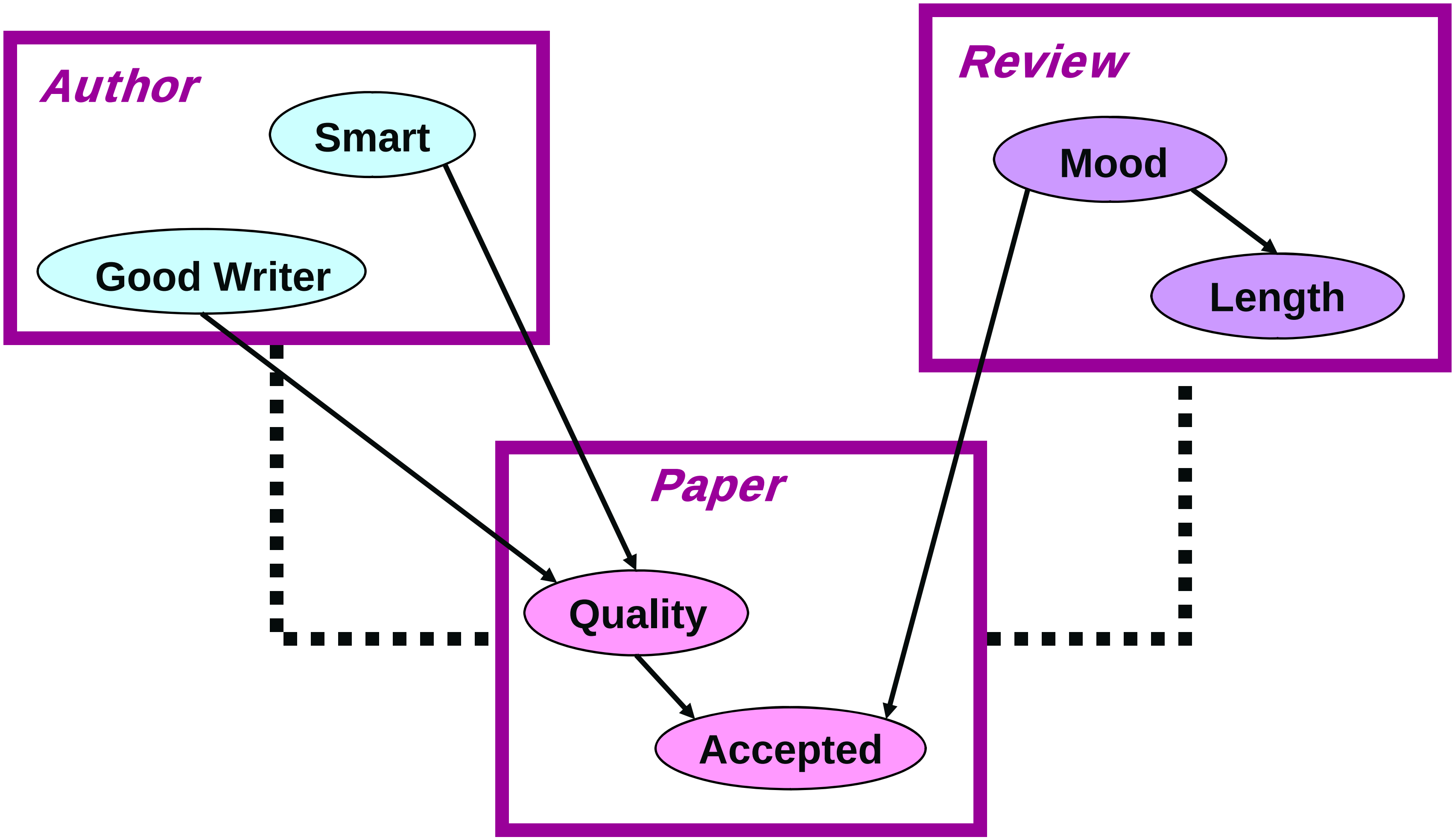
- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**
    - PRMs w/ Attribute Uncertainty
    - Inference in PRMs
    - Learning in PRMs
    - PRMs w/ Structural Uncertainty
    - PRMs w/ Class Hierarchies
- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Relational Schema

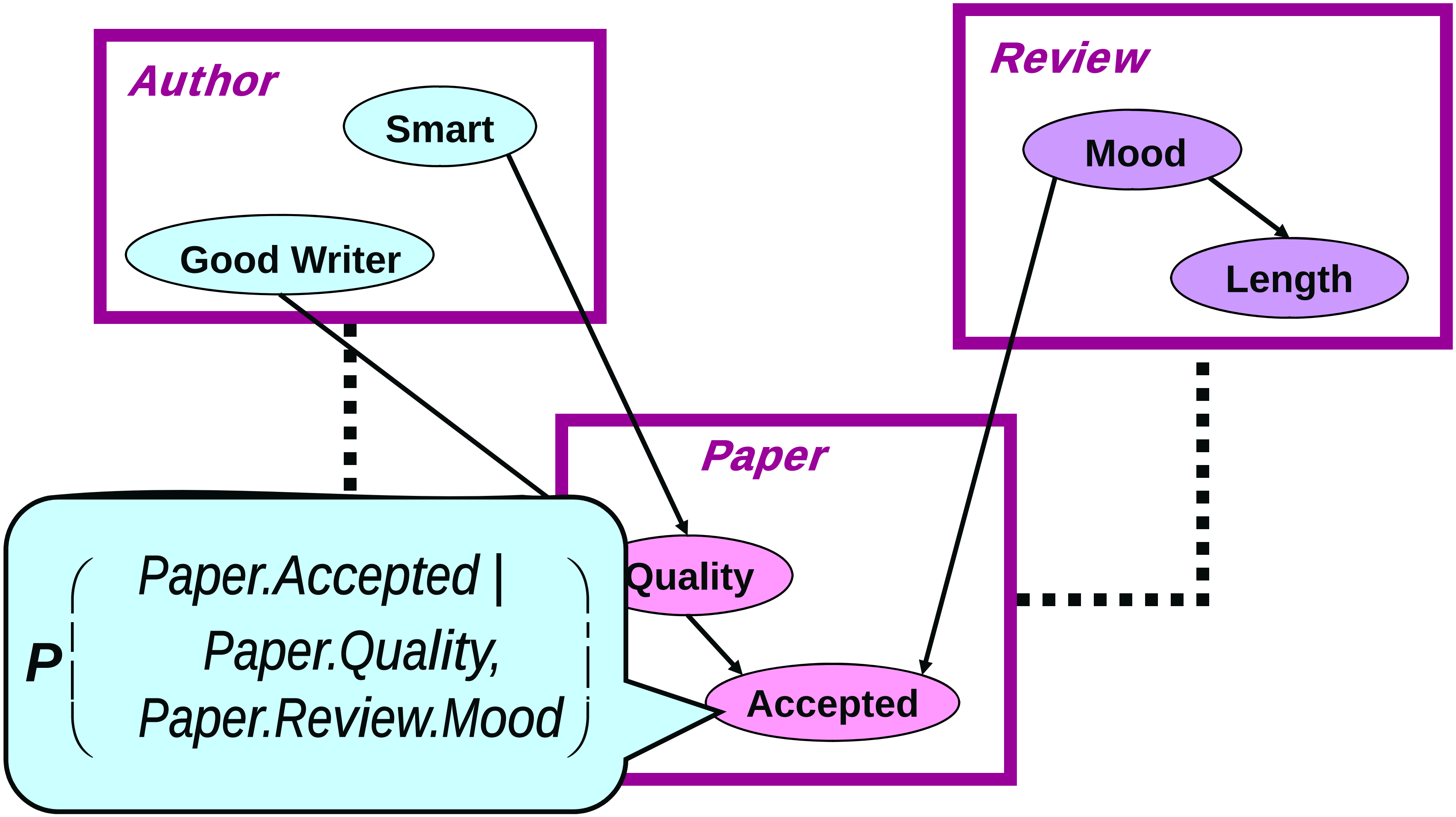


- Describes the types of objects and relations in the database

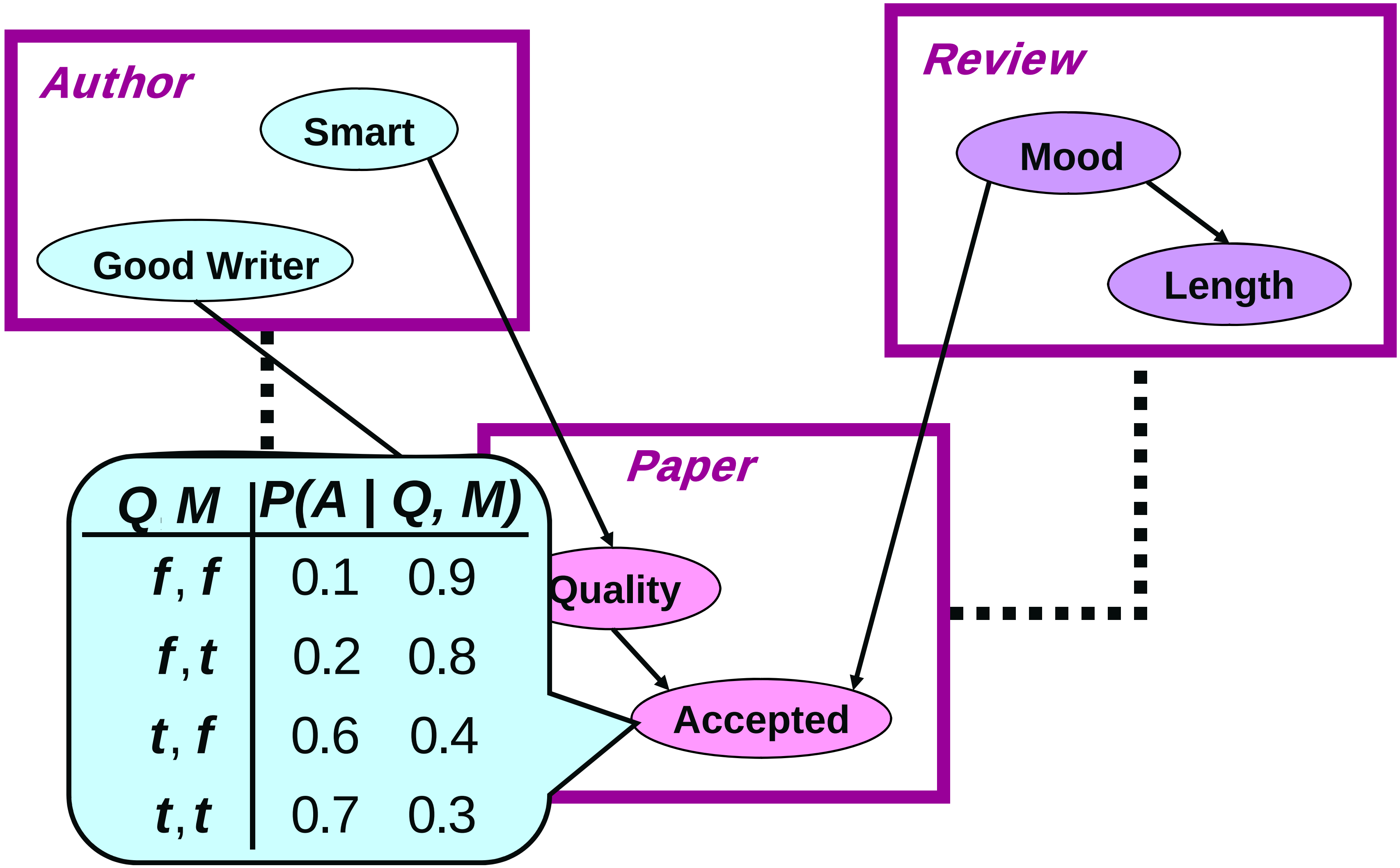
Probabilistic Relational Model



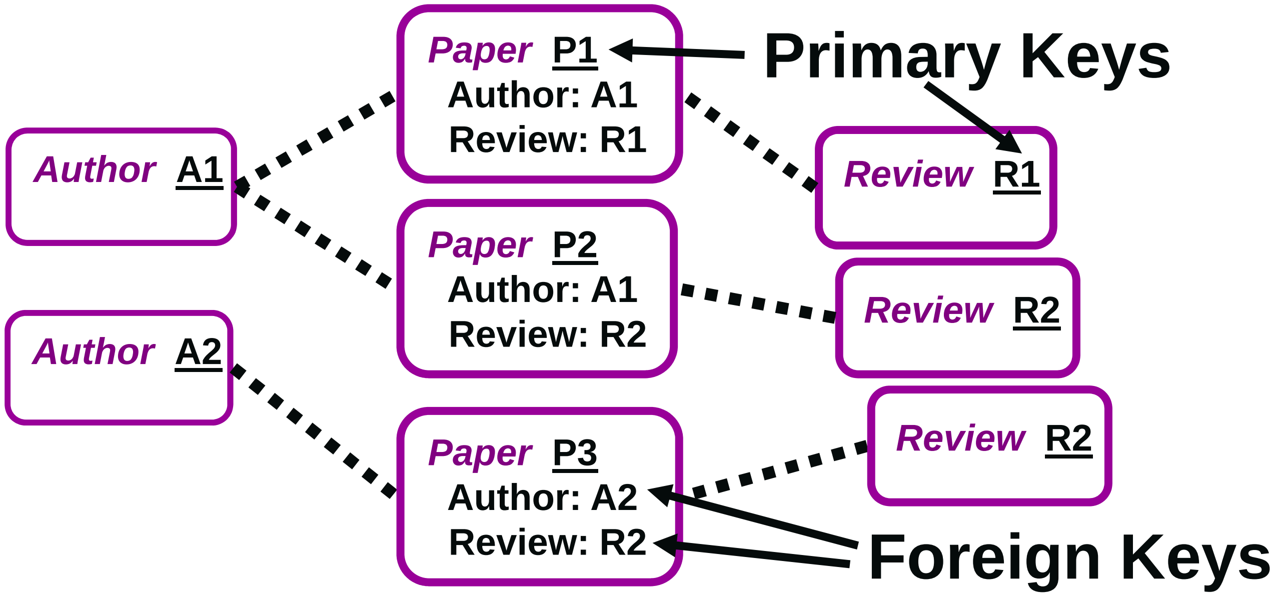
Probabilistic Relational Model



Probabilistic Relational Model



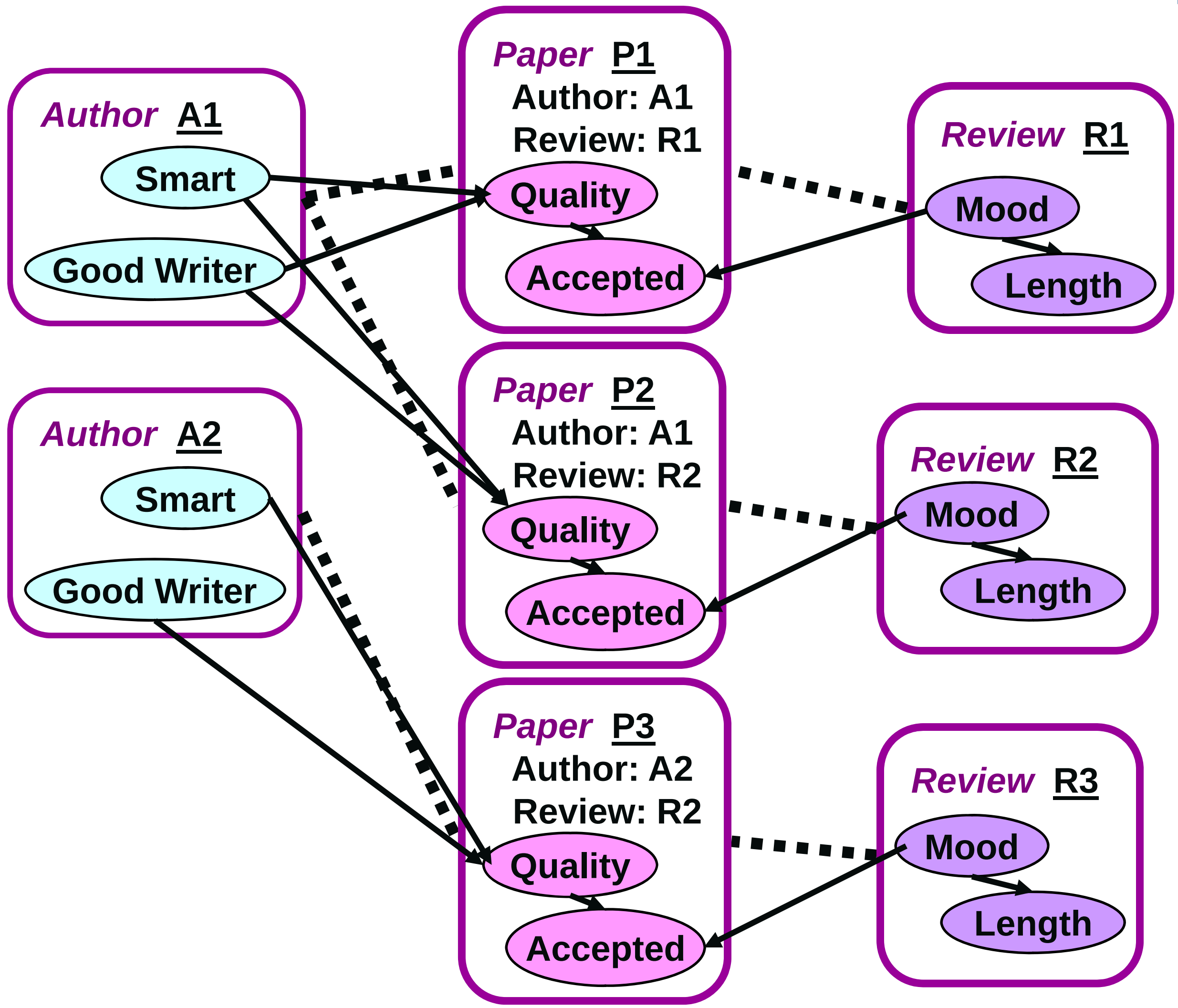
Relational Skeleton



Fixed relational skeleton σ:

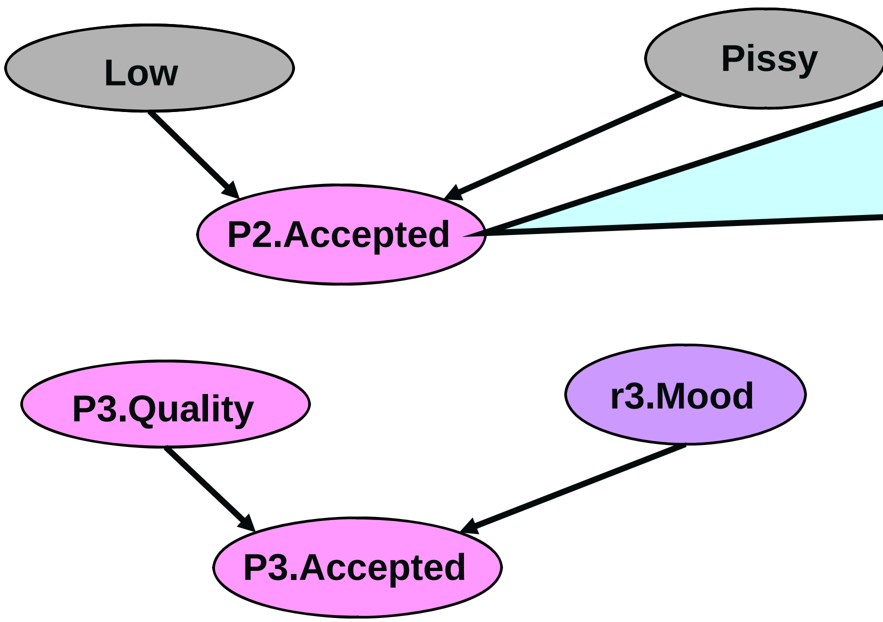
- set of objects in each class
- relations between them

PRM w/ Attribute Uncertainty



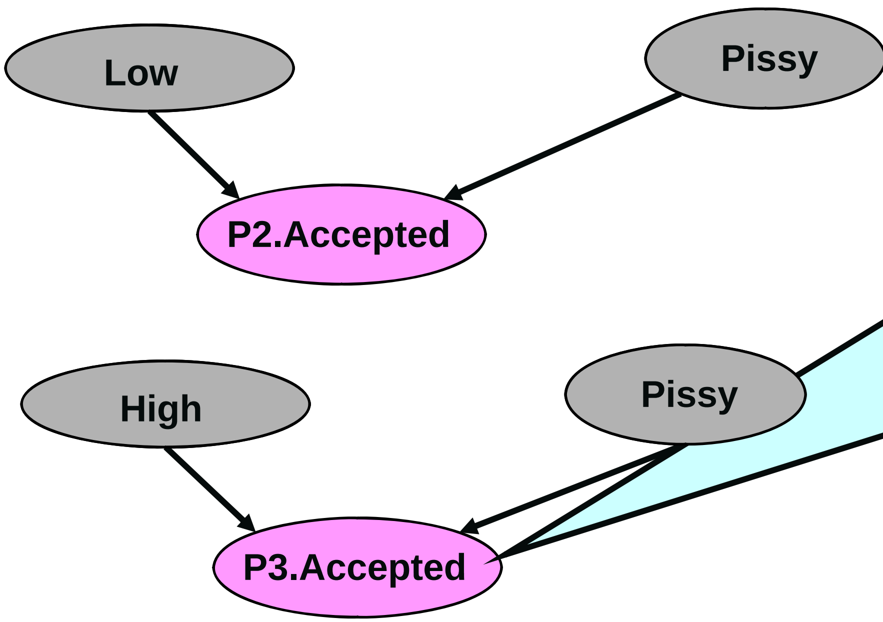
PRM defines distribution over instantiations of attributes

A Portion of the BN



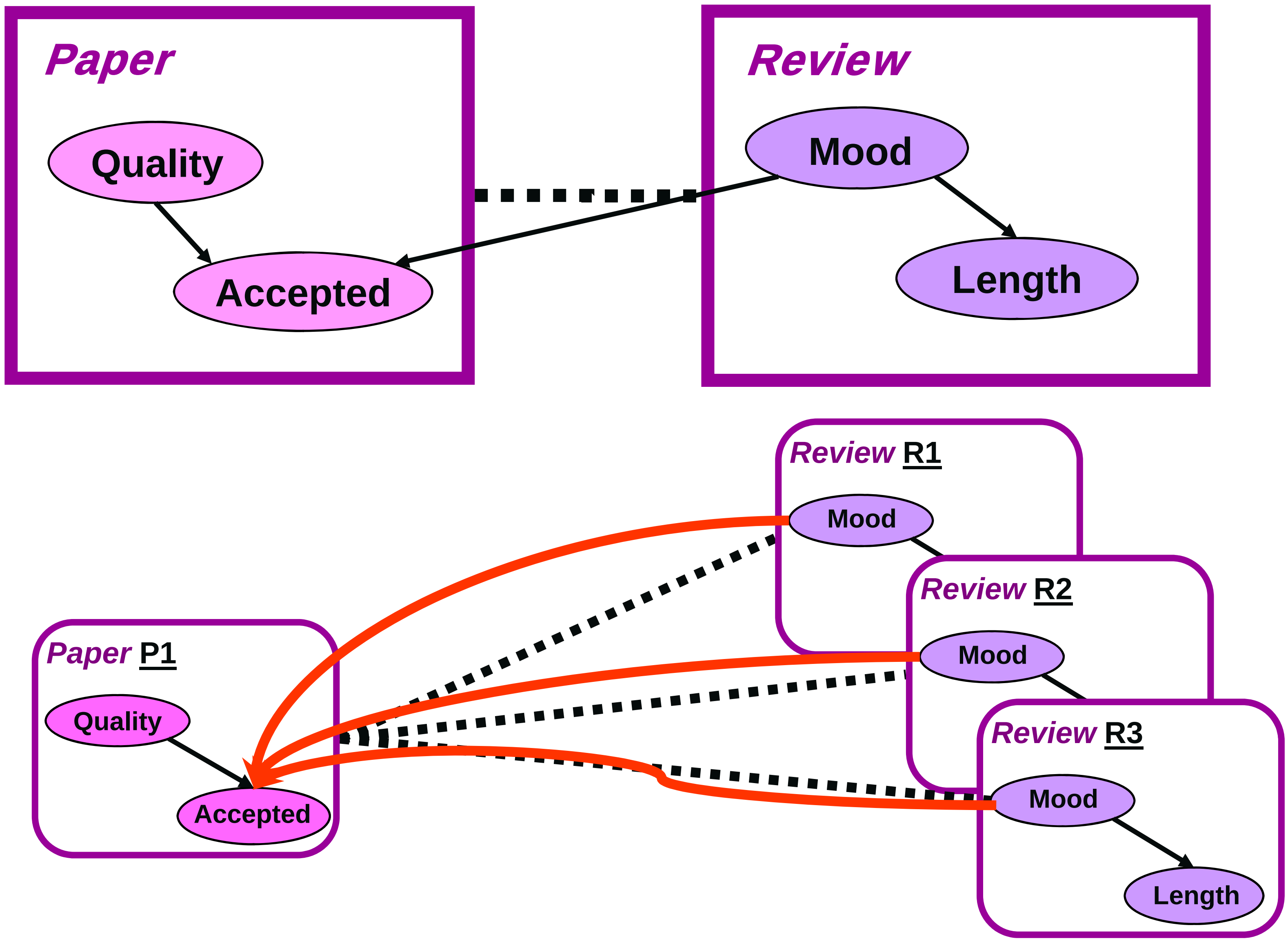
Q, *M*	*P(A* | *Q, M)*	
f, ***f***	0.1	0.9
f, ***t***	0.2	0.8
t, ***f***	0.6	0.4
t, ***t***	0.7	0.3

A Portion of the BN

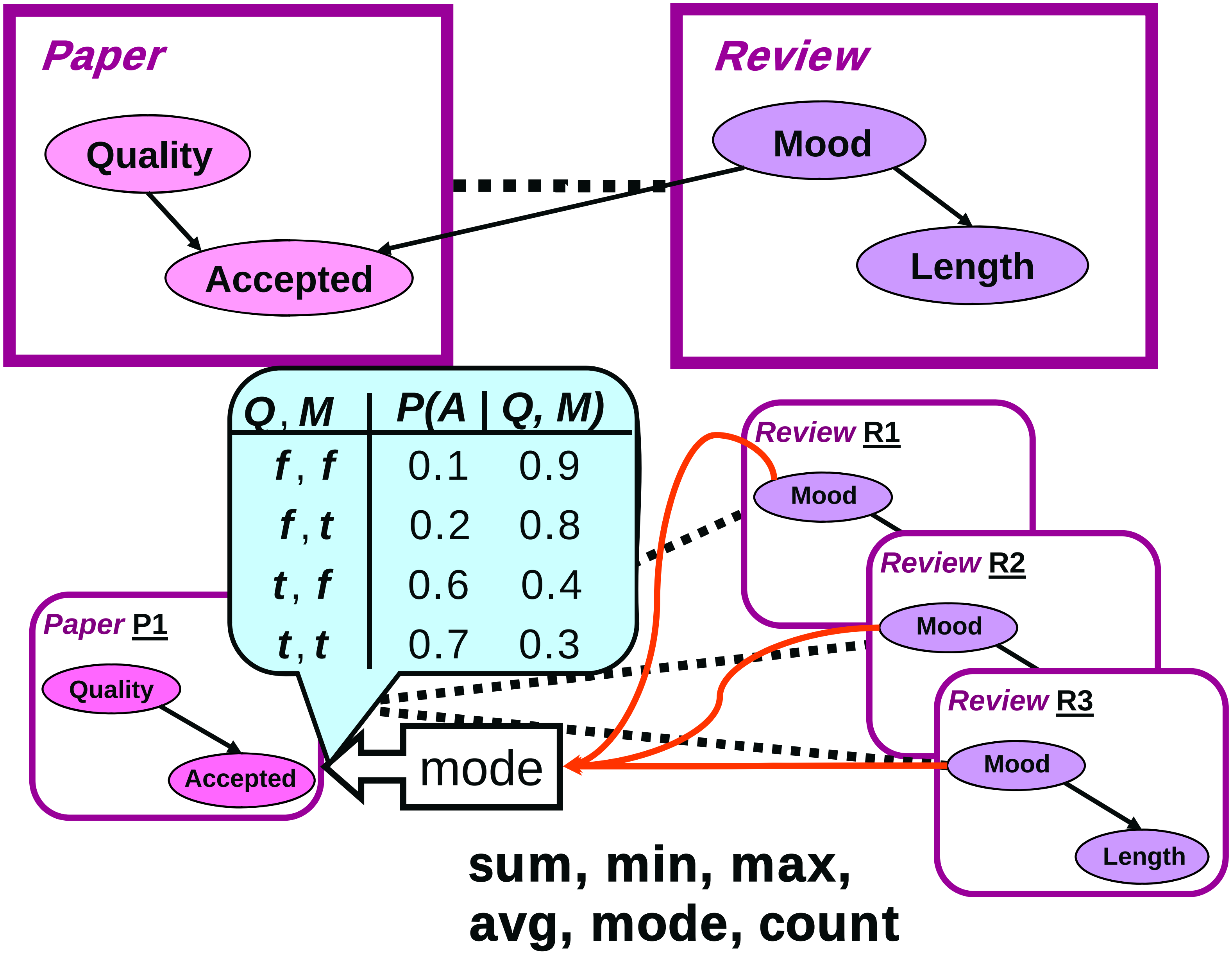


Q M	*P(A | Q, M)*	
f, ***f***	0.1	0.9
f, ***t***	0.2	0.8
t, ***f***	0.6	0.4
t, ***t***	0.7	0.3

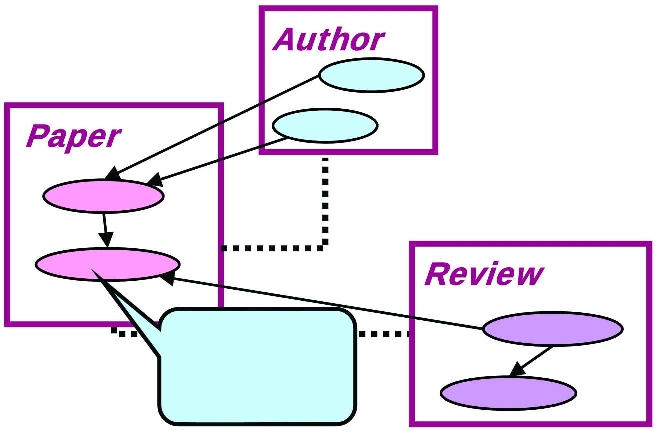
PRM: Aggregate Dependencies

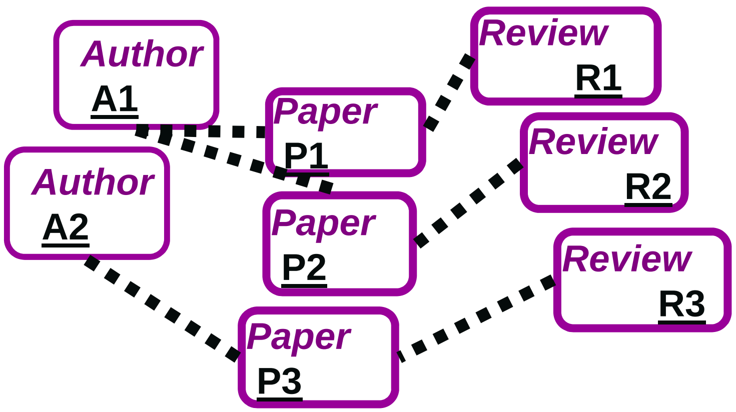


PRM: Aggregate Dependencies



PRM with AU Semantics





PRM + relational skeleton σ =

probability distribution over completions I:

$$P(\mathrm{I} \mid \sigma, S, \Theta) = \prod_{x \in \sigma} \prod_{x.A} P(x.A \mid parents_{S,\sigma}(x.A))$$

Objects   Attributes

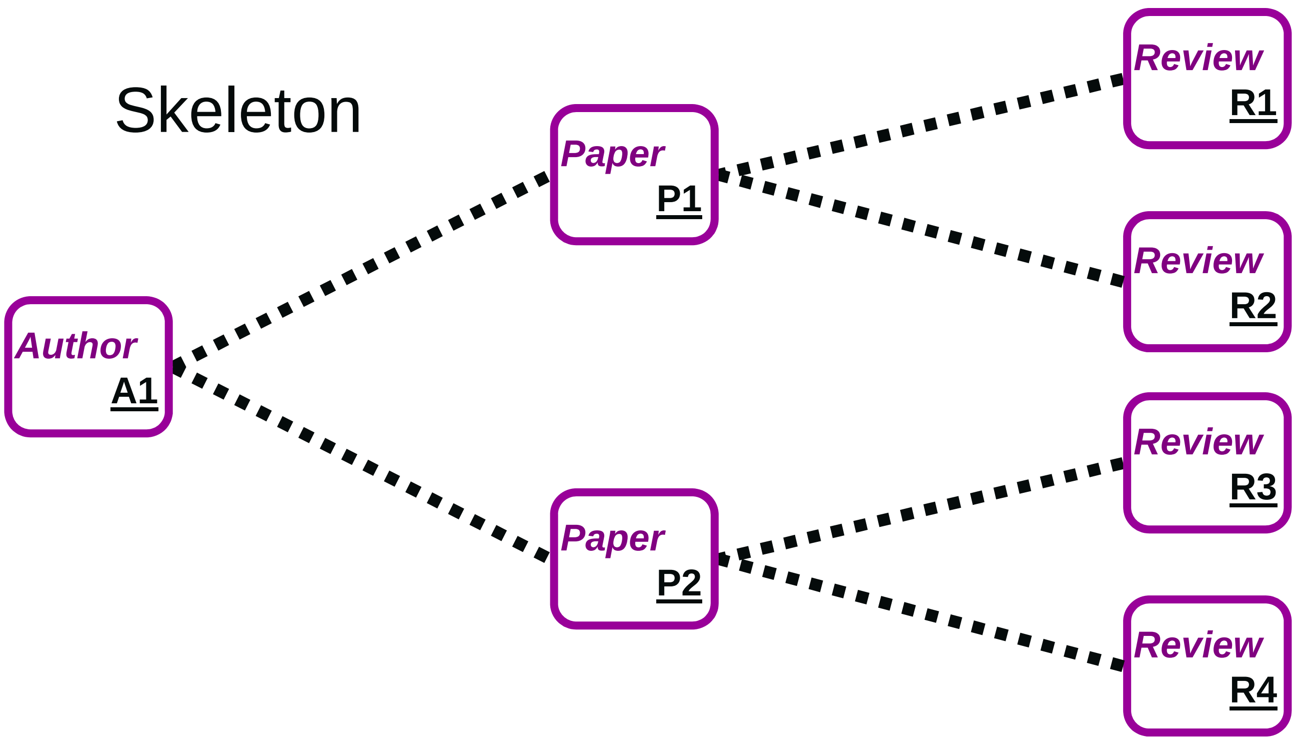
Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**
    - PRMs w/ Attribute Uncertainty
    - Inference in PRMs
    - Learning in PRMs
    - PRMs w/ Structural Uncertainty
    - PRMs w/ Class Hierarchies
- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

PRM Inference

- Simple idea: enumerate all attributes of all objects
- Construct a Bayesian network over all the attributes

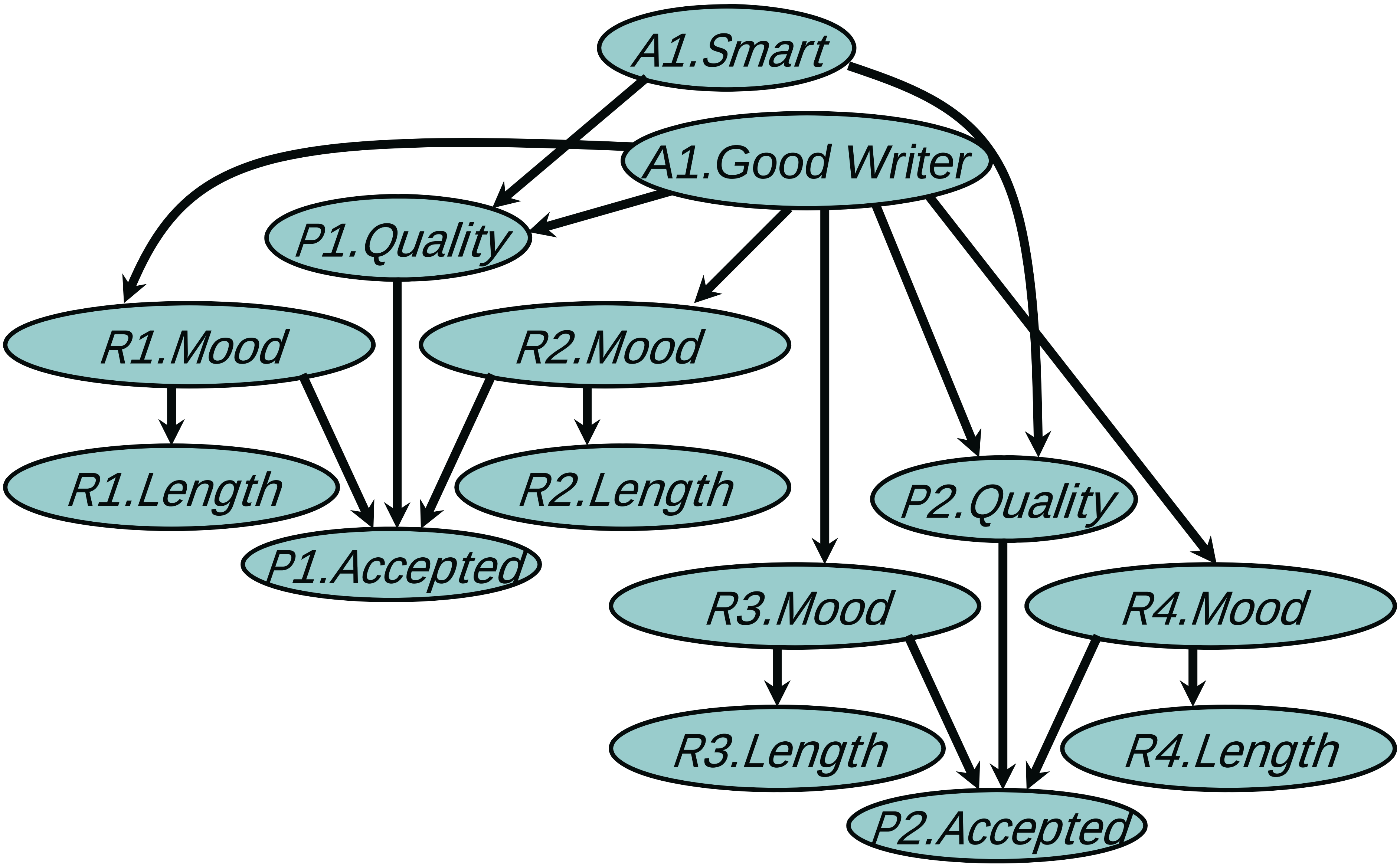
Inference Example



Query is ***P*****(A1.good-writer)**

Evidence is **P1.accepted = T, P2.accepted = T**

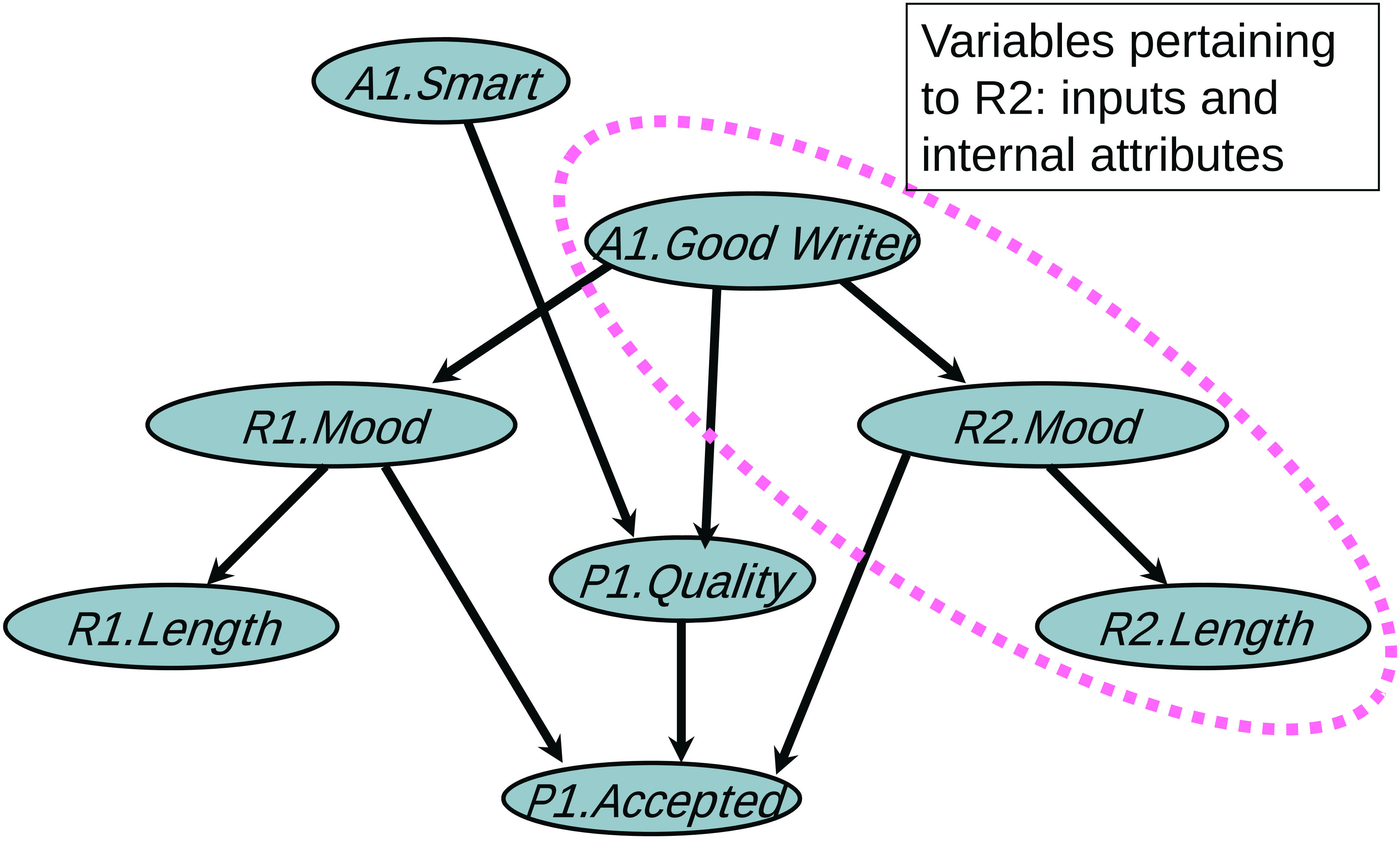
PRM Inference: Constructed BN



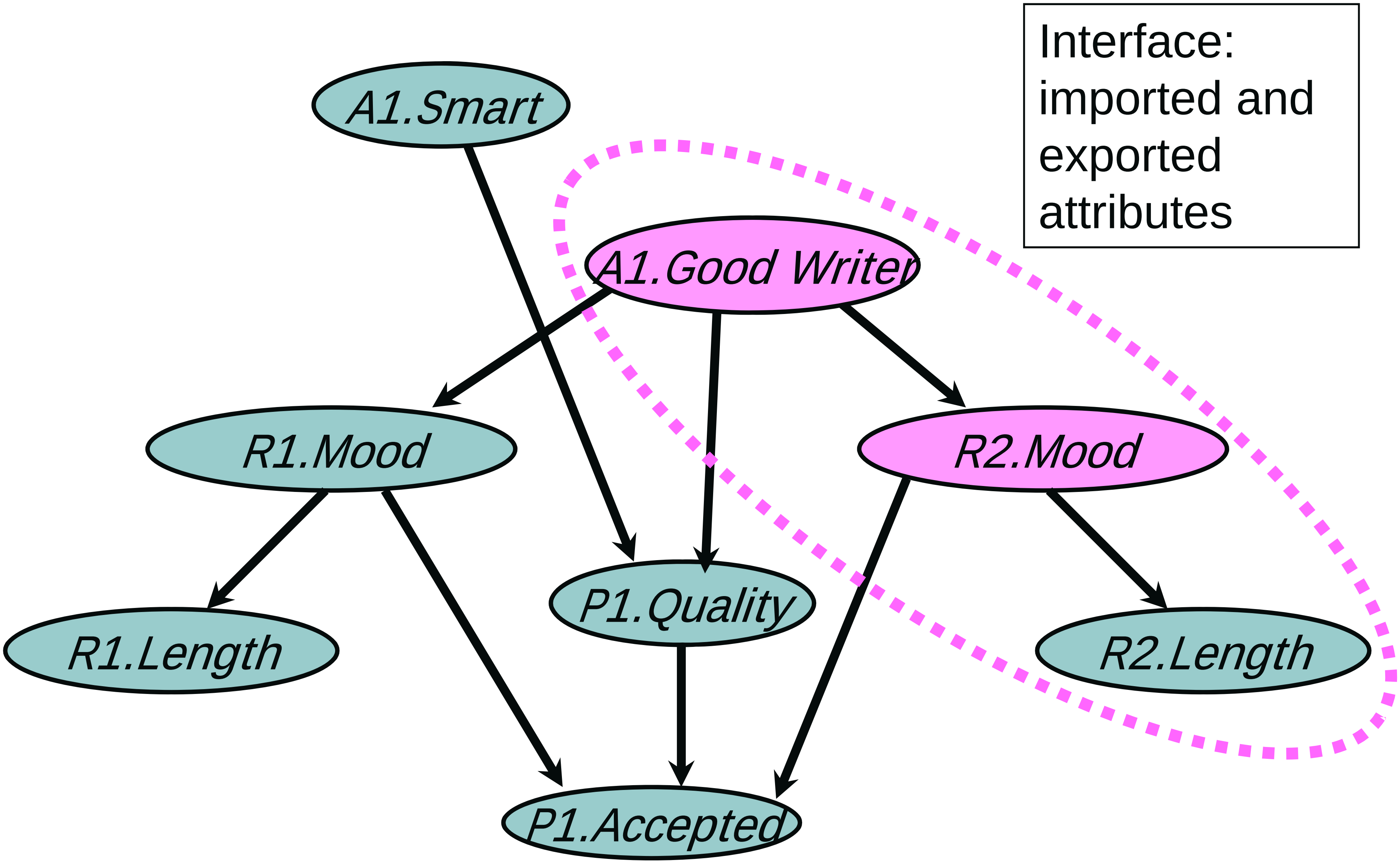
PRM Inference

- Problems with this approach:
  - constructed BN may be very large
  - doesn’t exploit object structure
- Better approach:
  - reason about objects themselves
  - reason about whole classes of objects
- In particular, exploit:
  - reuse of inference
  - encapsulation of objects

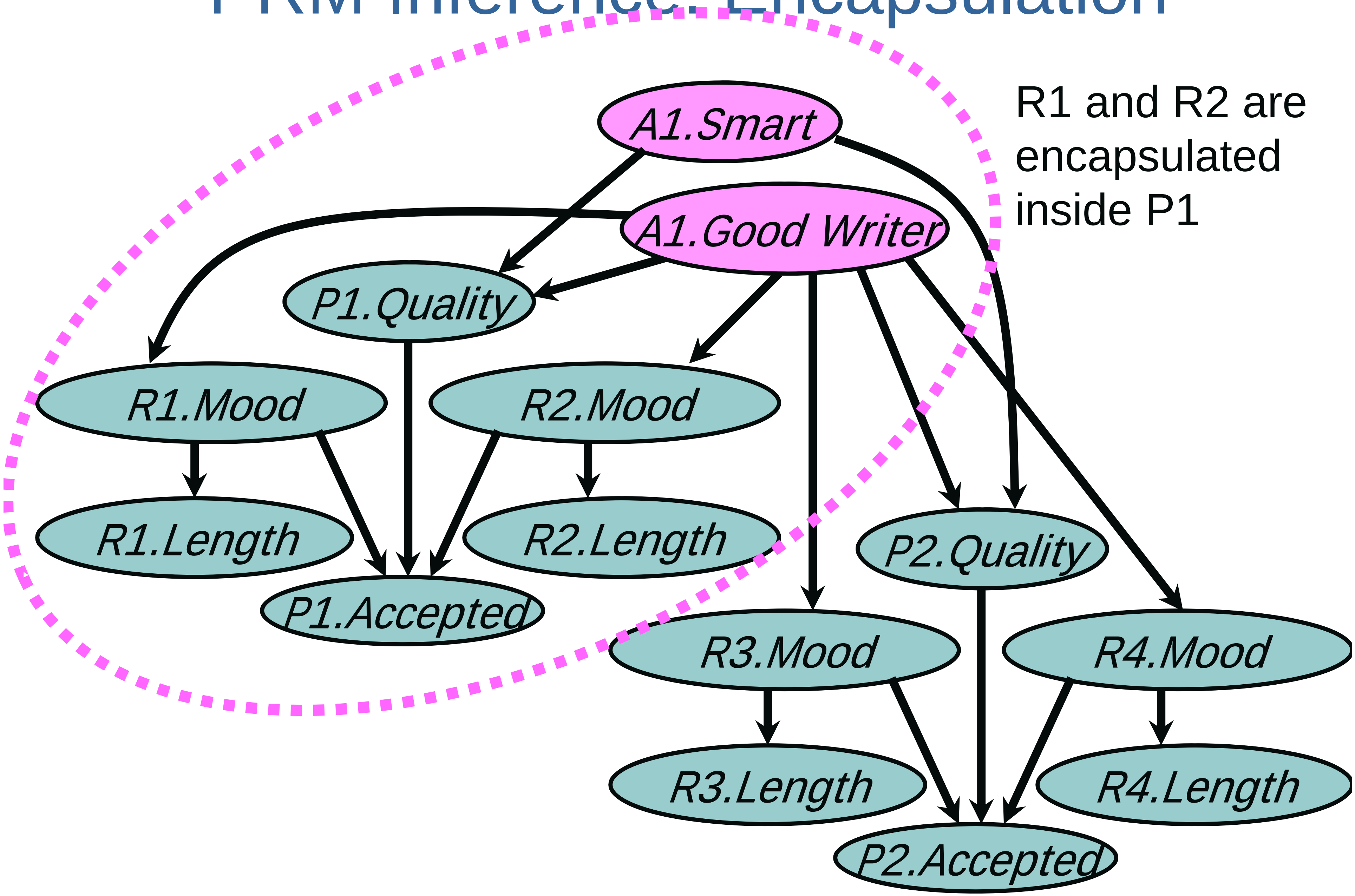
PRM Inference: Interfaces



PRM Inference: Interfaces

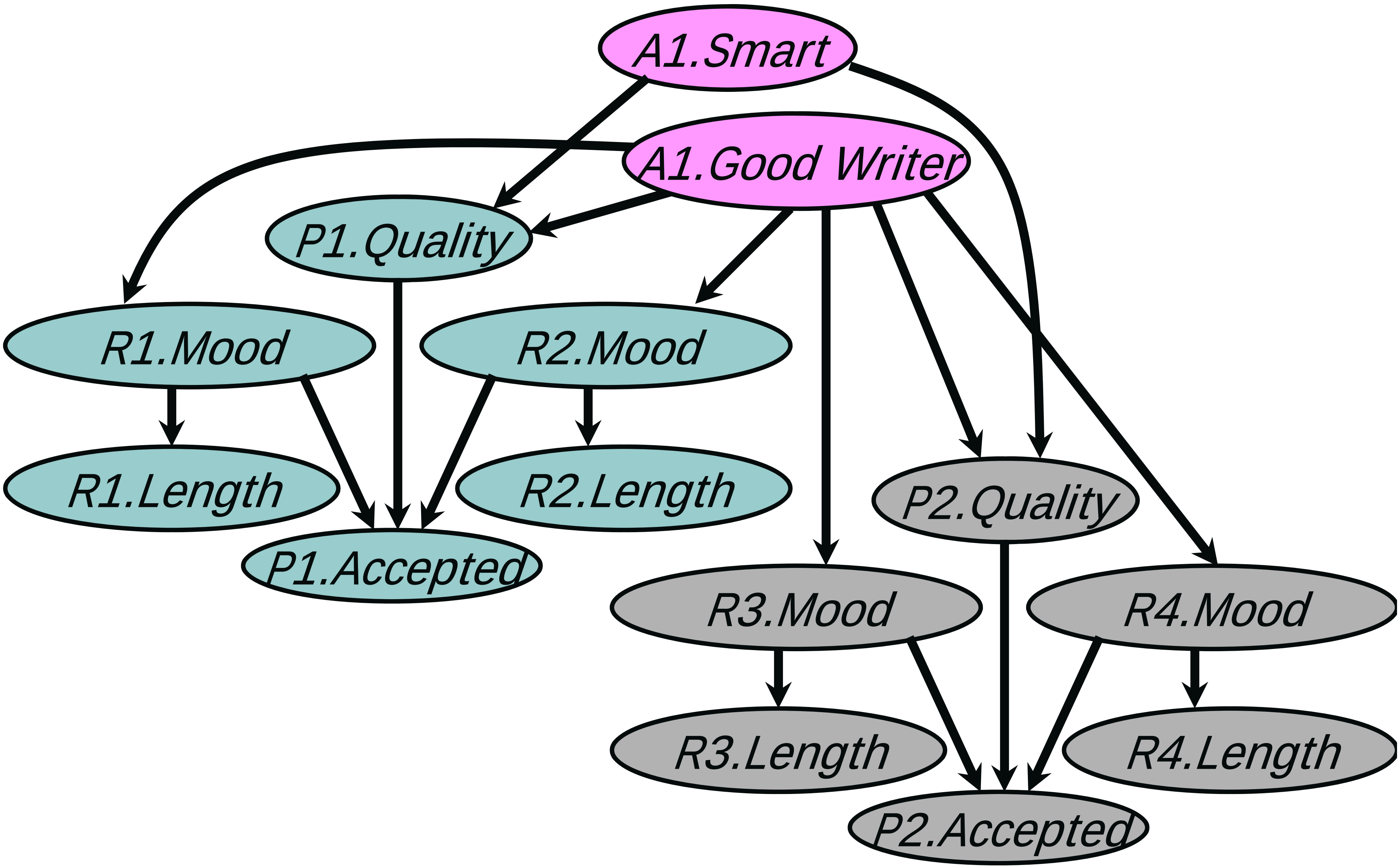


PRM Inference: Encapsulation

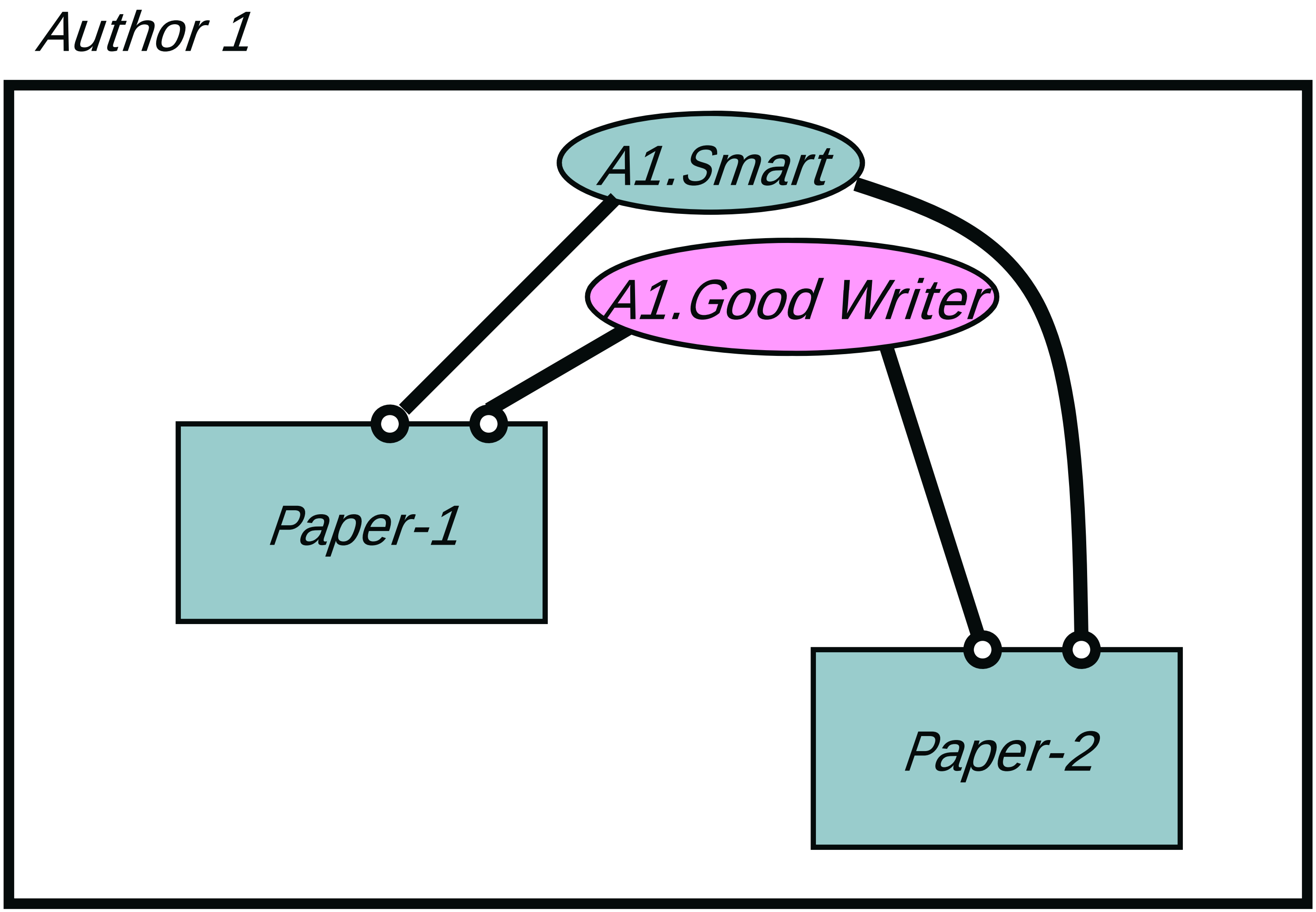


R1 and R2 are encapsulated inside P1

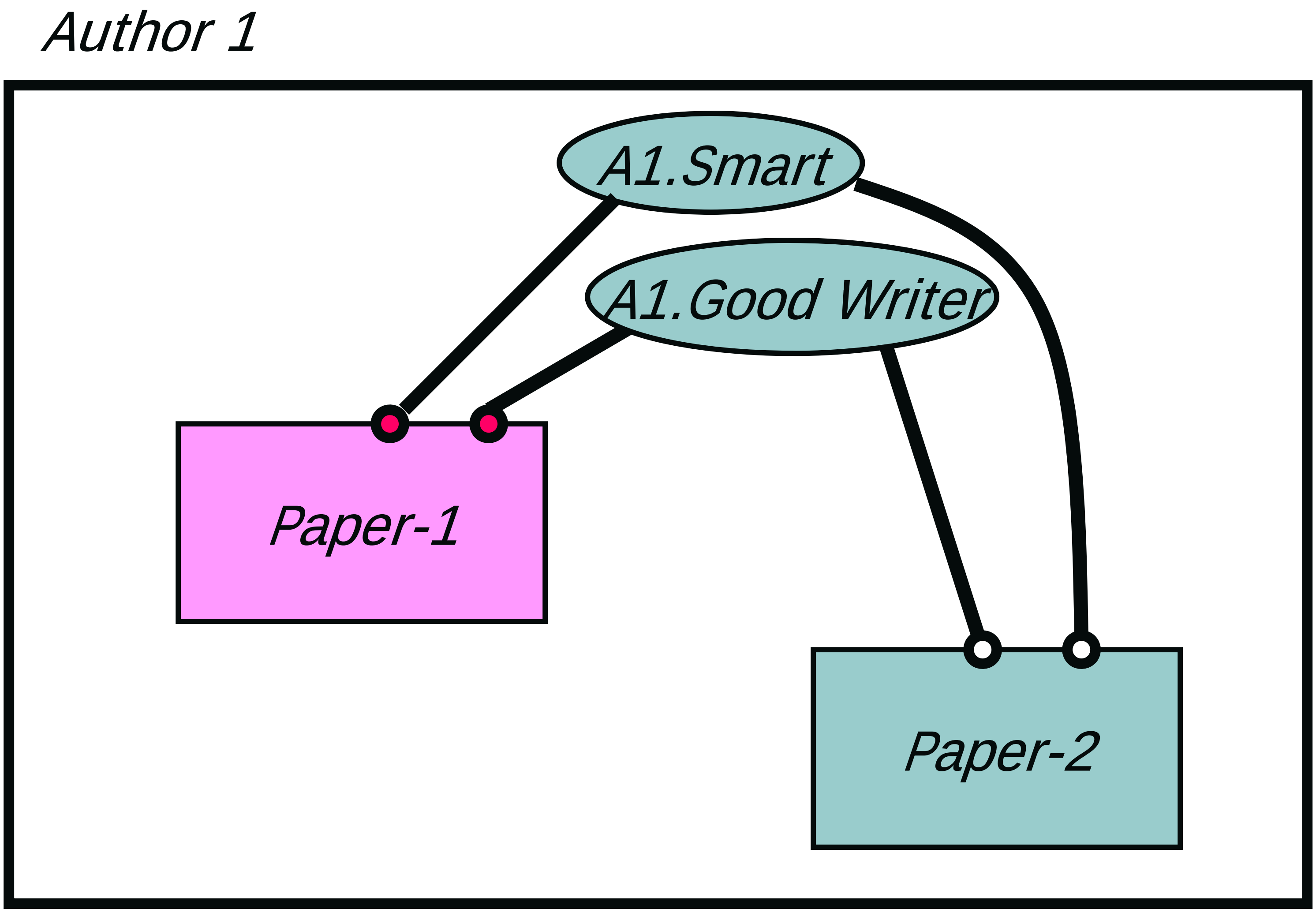
PRM Inference: Reuse



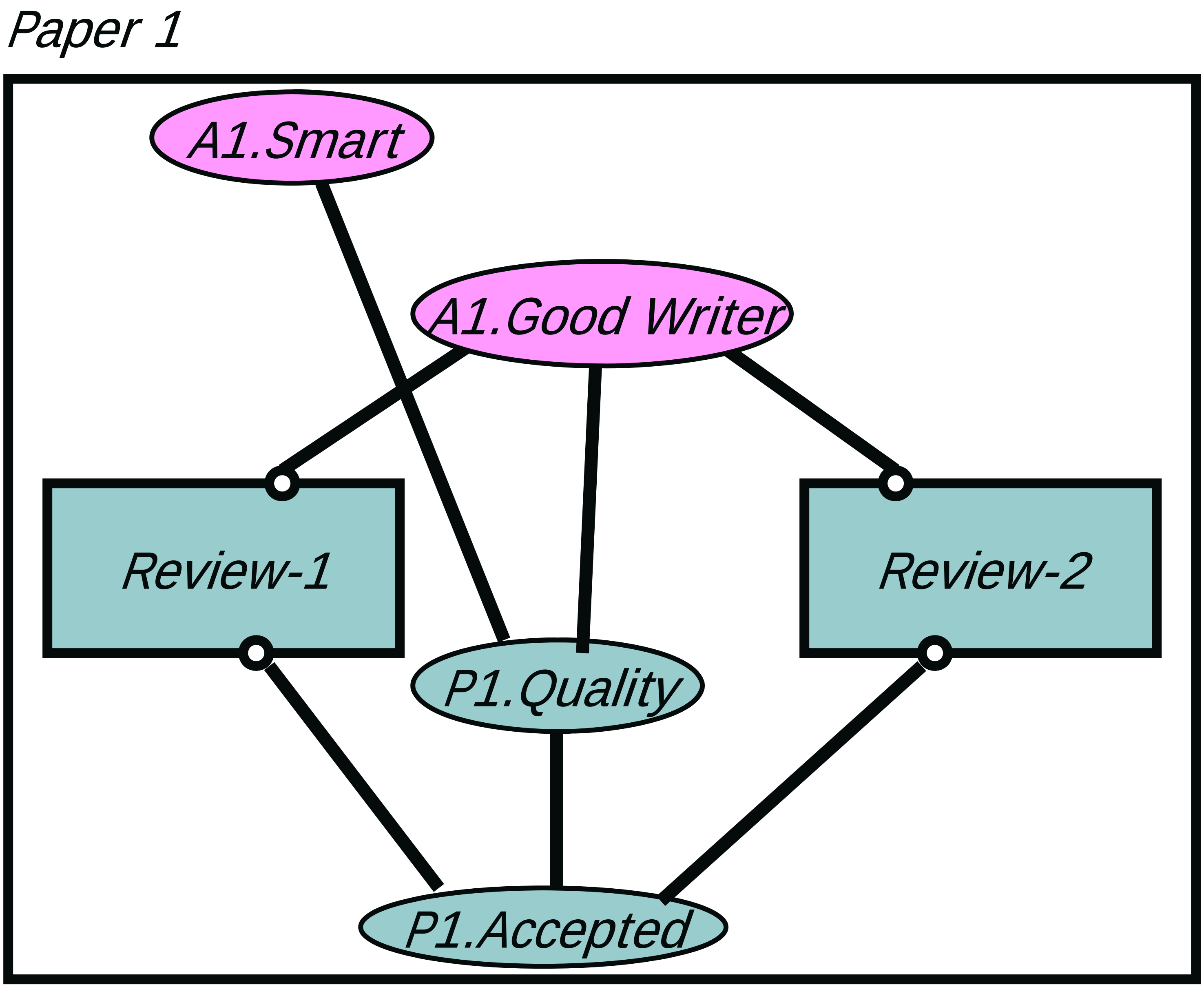
Structured Variable Elimination



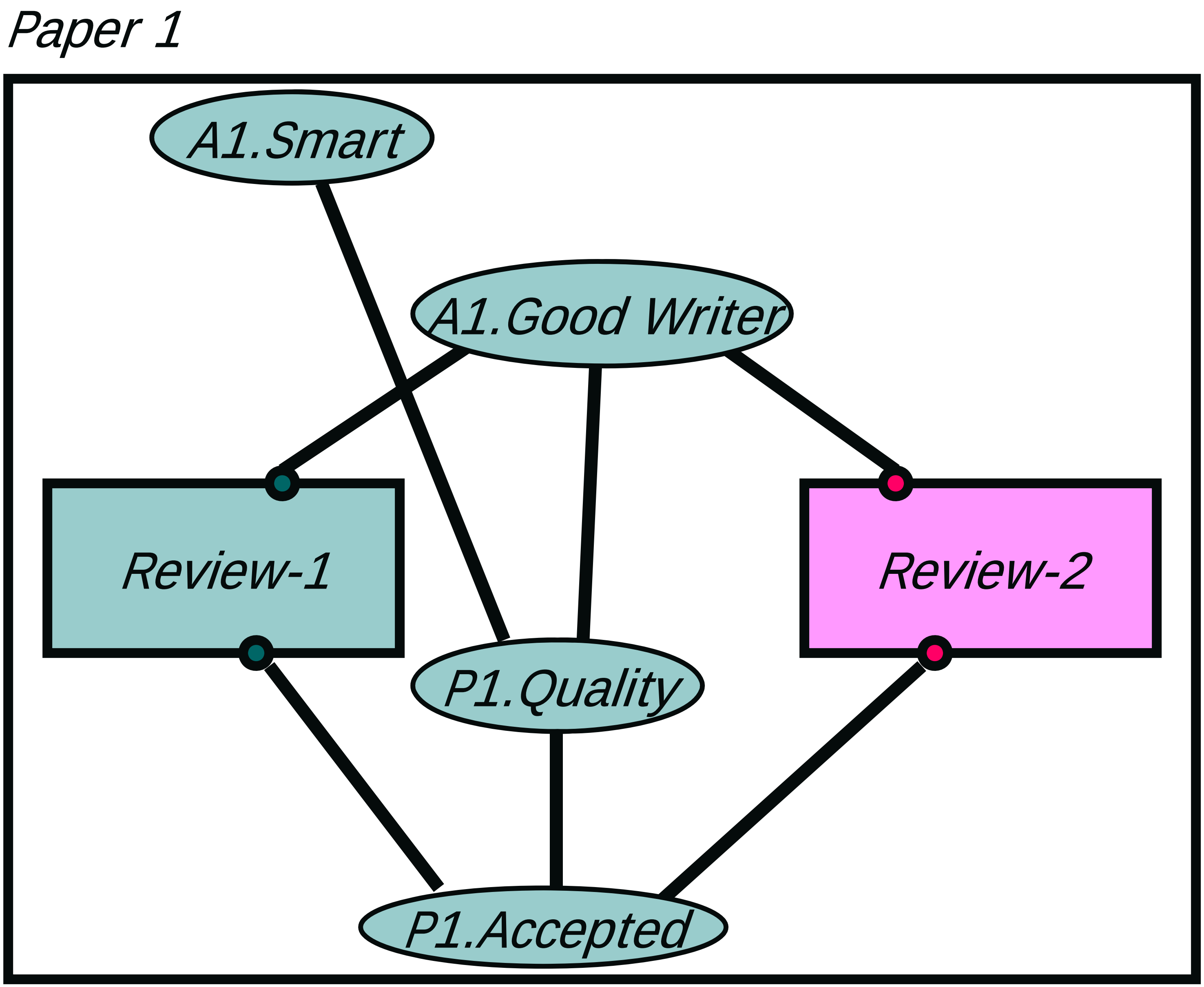
Structured Variable Elimination



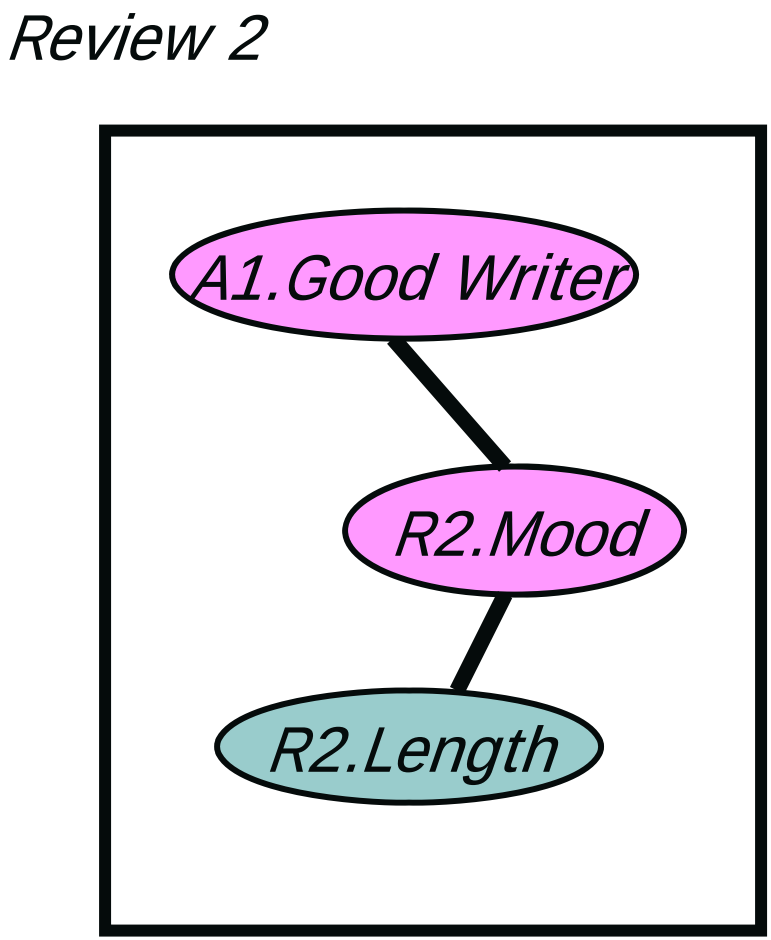
Structured Variable Elimination



Structured Variable Elimination



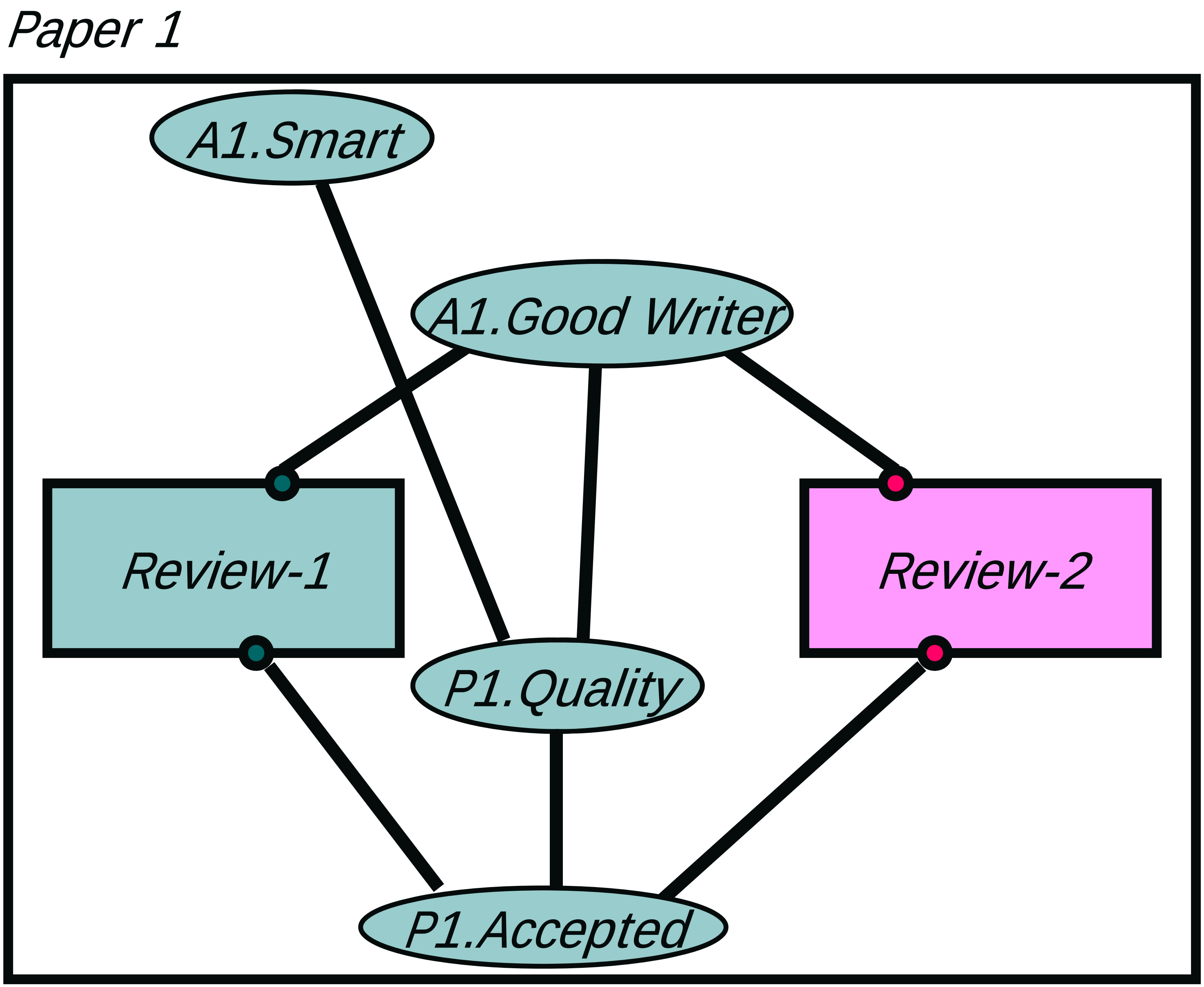
Structured Variable Elimination



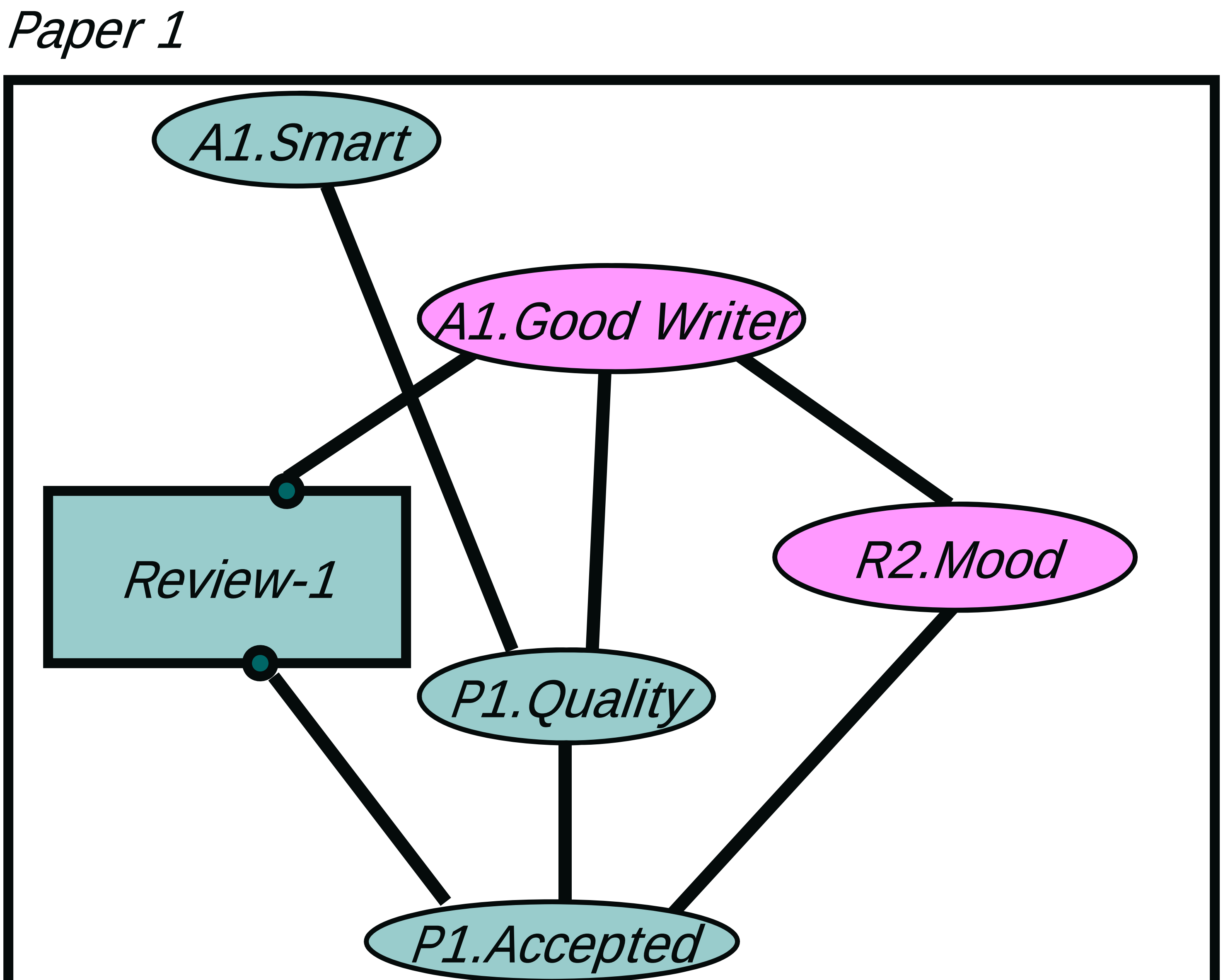
Structured Variable Elimination

Review 2

Structured Variable Elimination

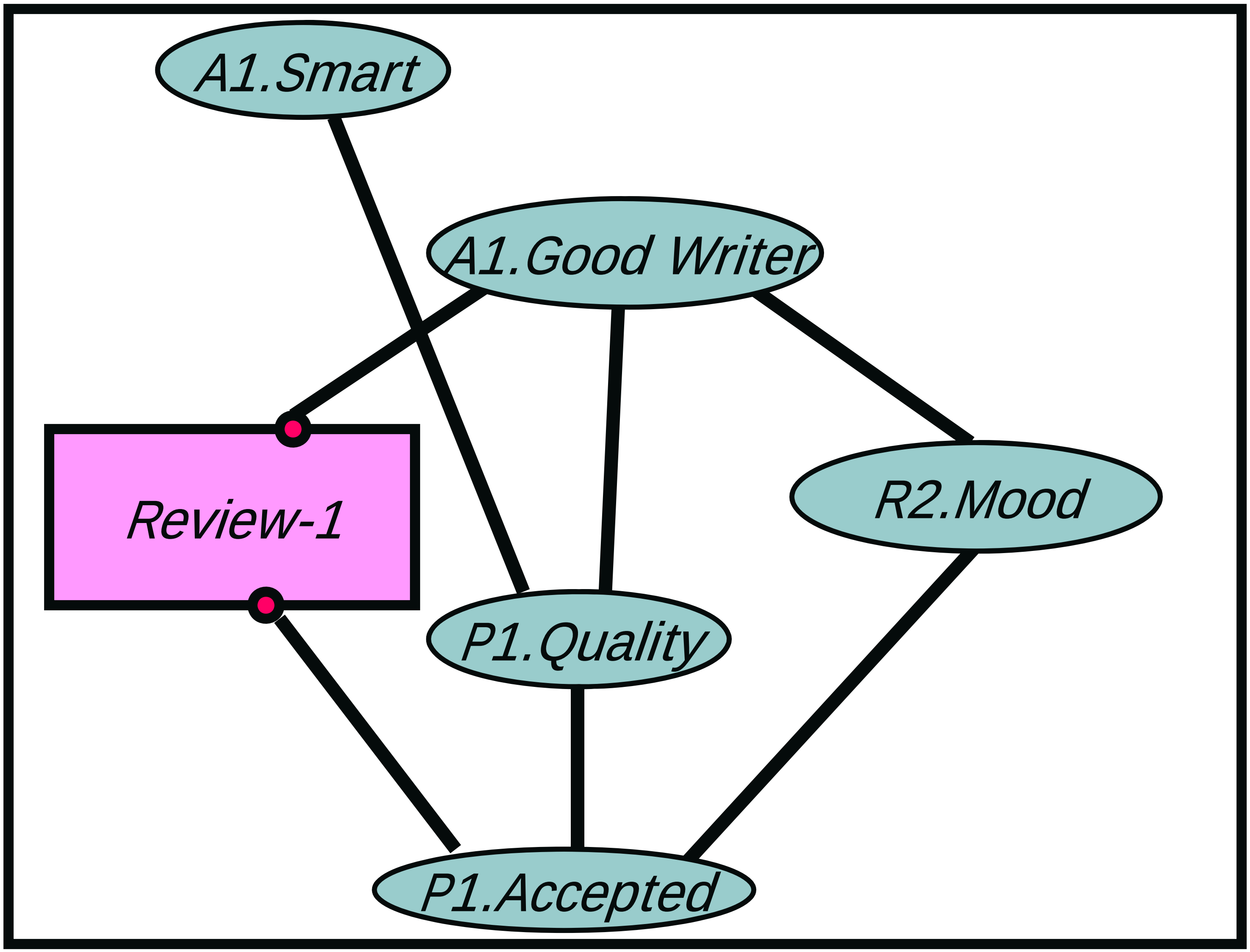


Structured Variable Elimination

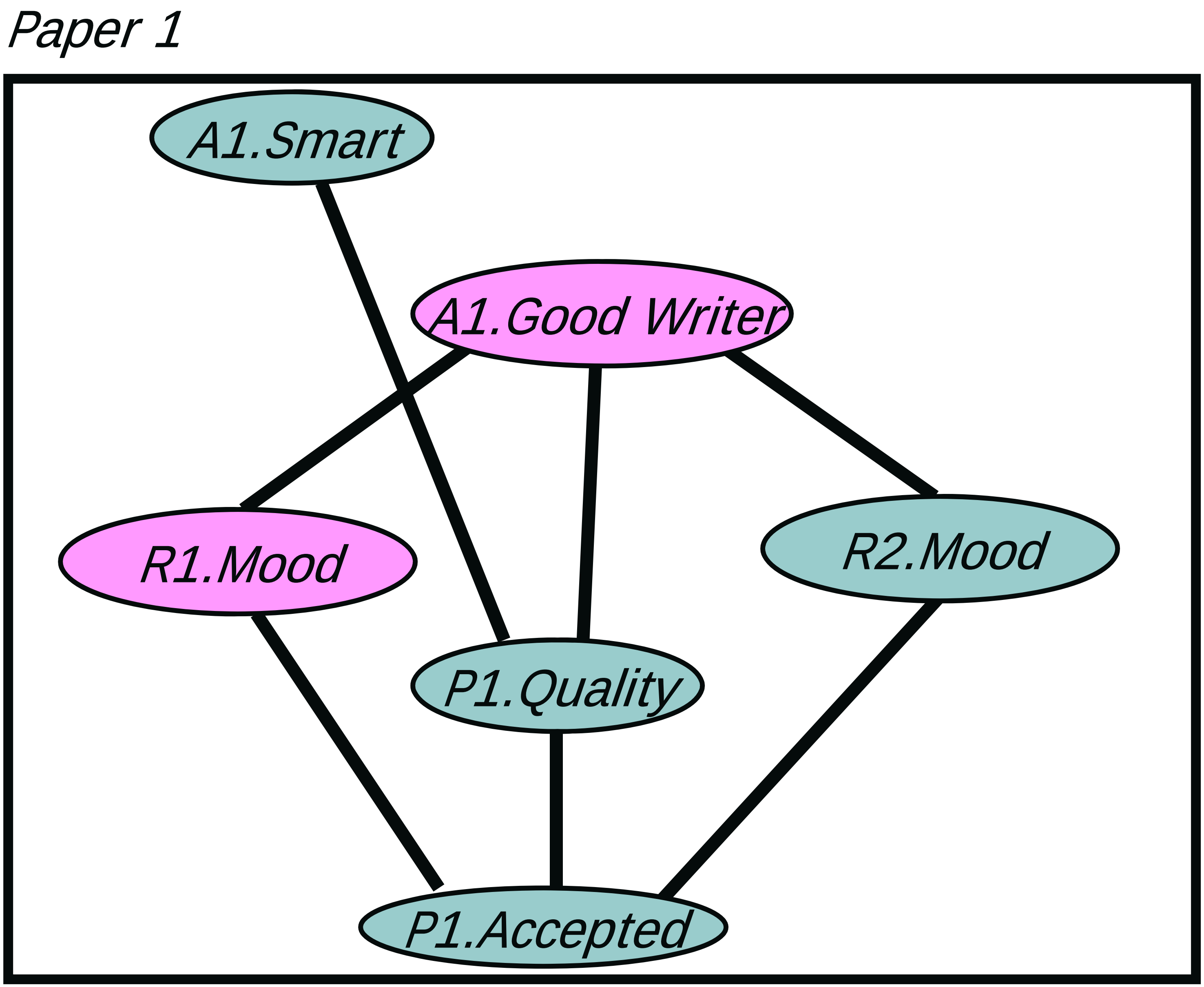


Structured Variable Elimination

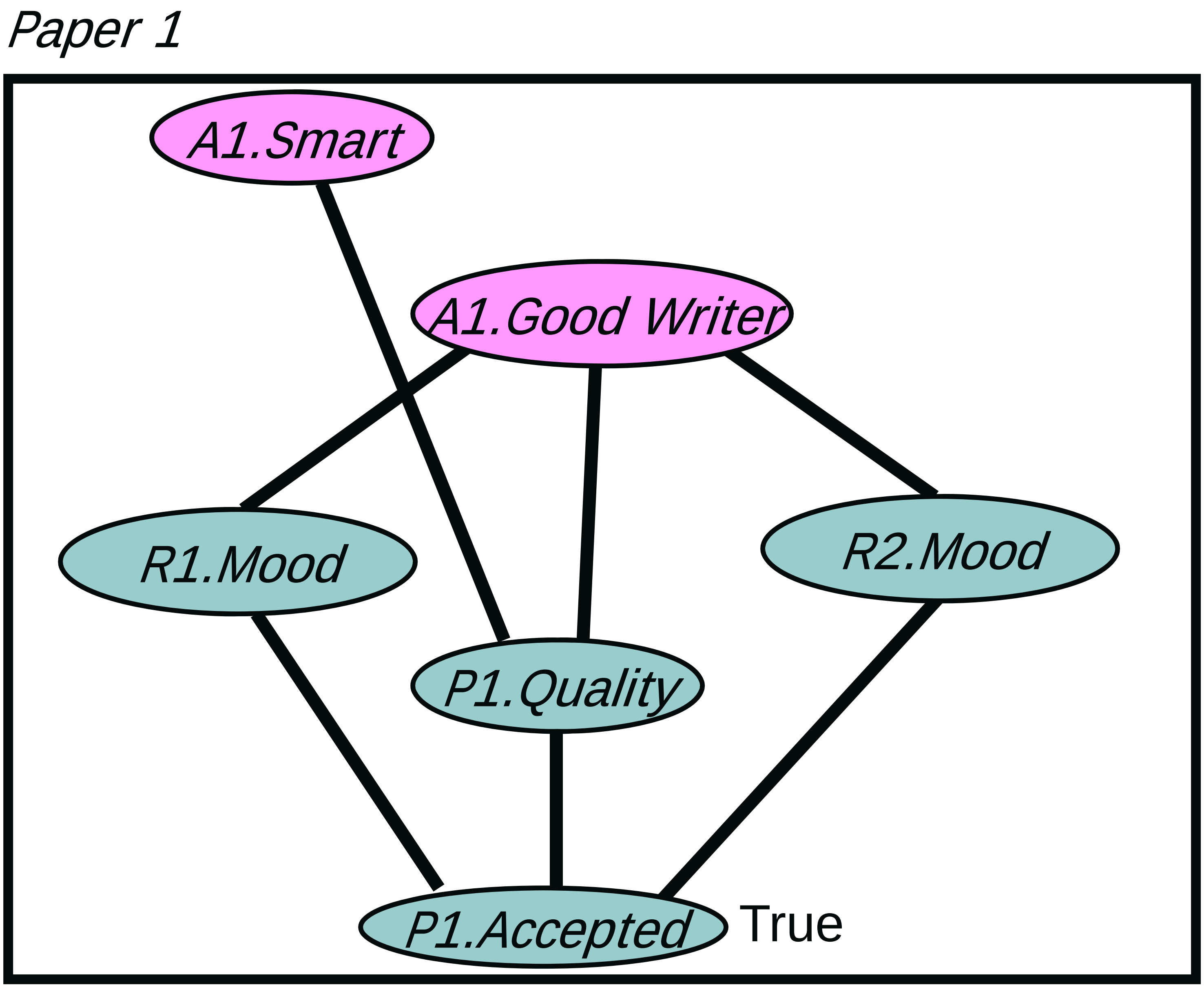
Paper 1



Structured Variable Elimination

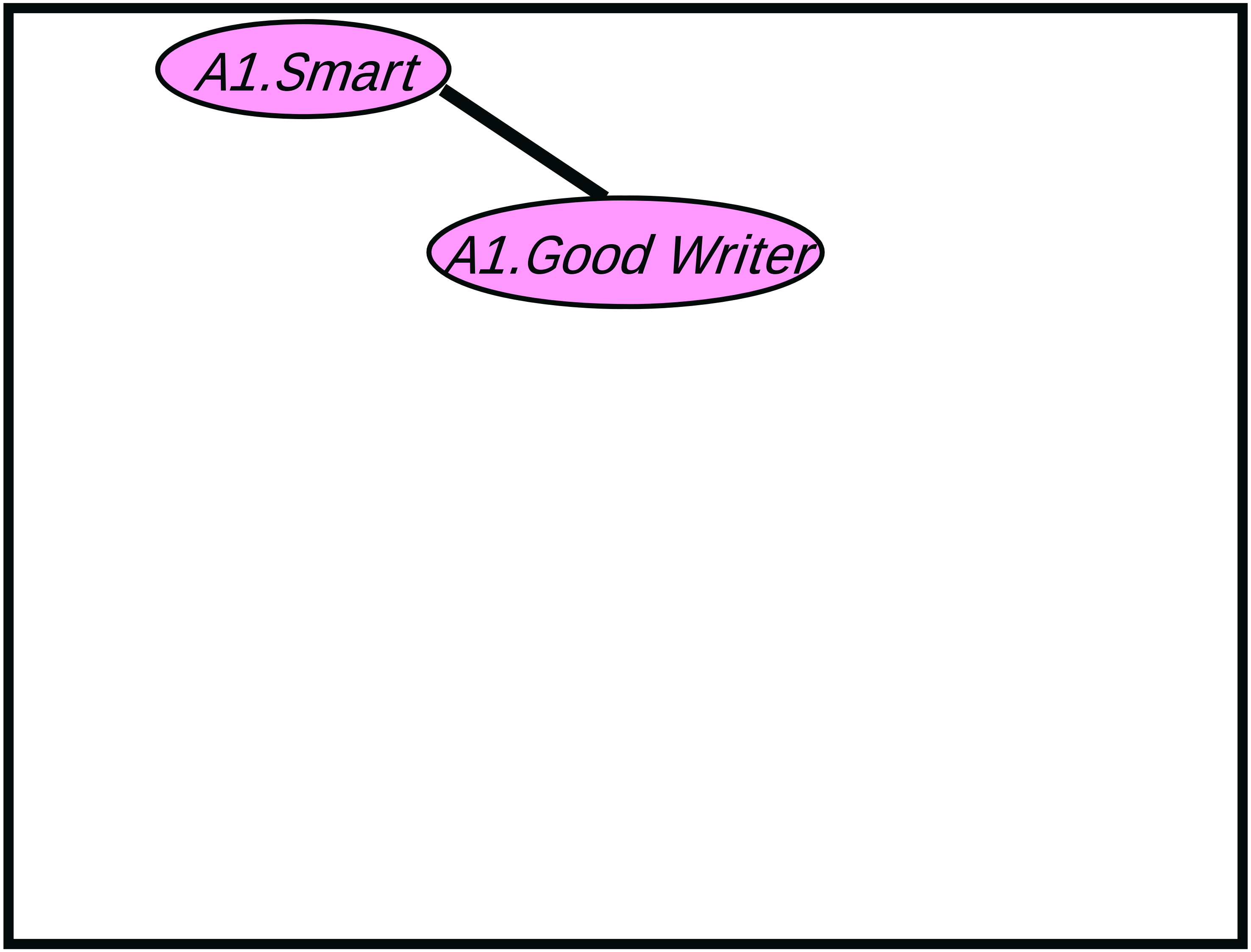


Structured Variable Elimination

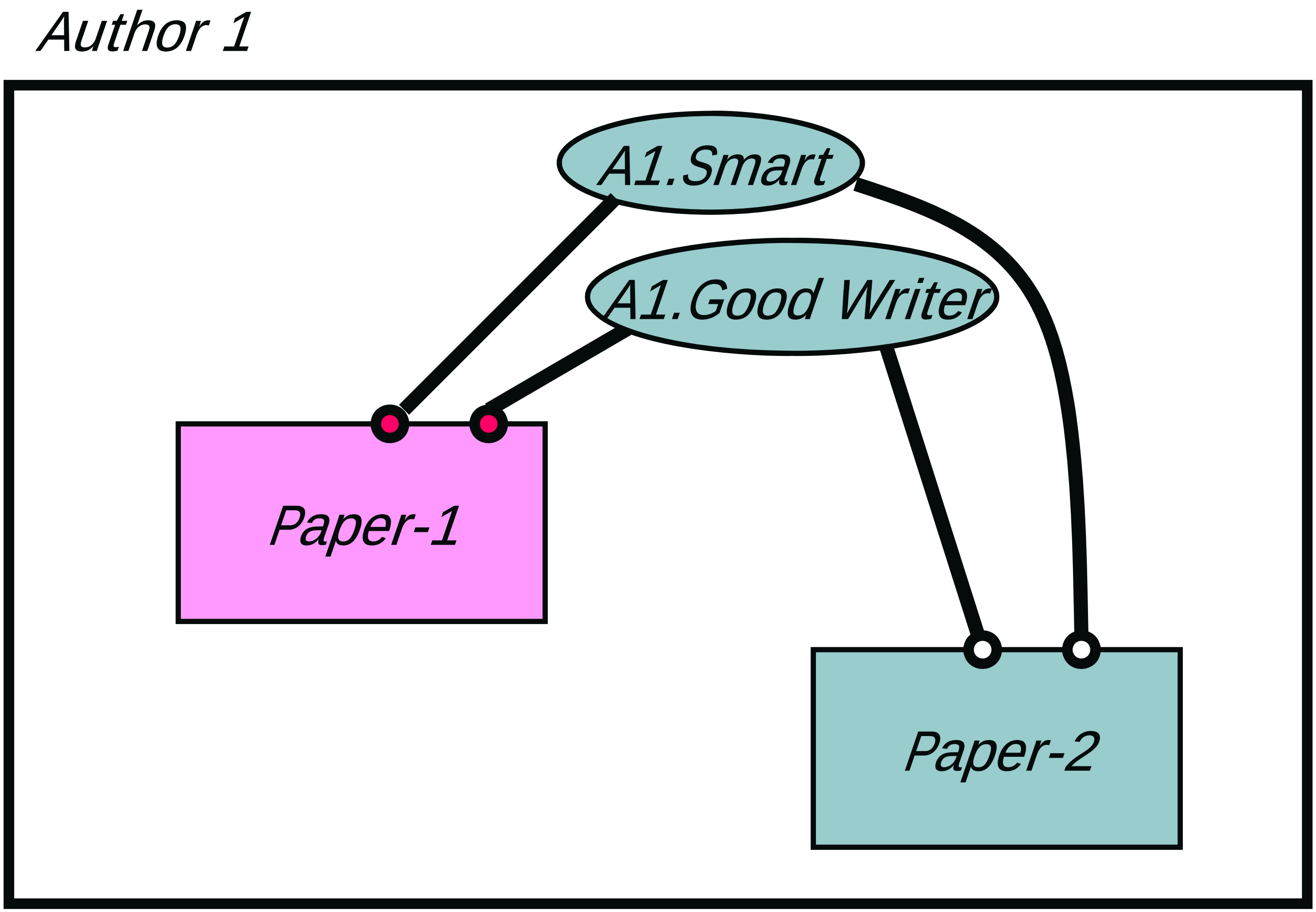


Structured Variable Elimination

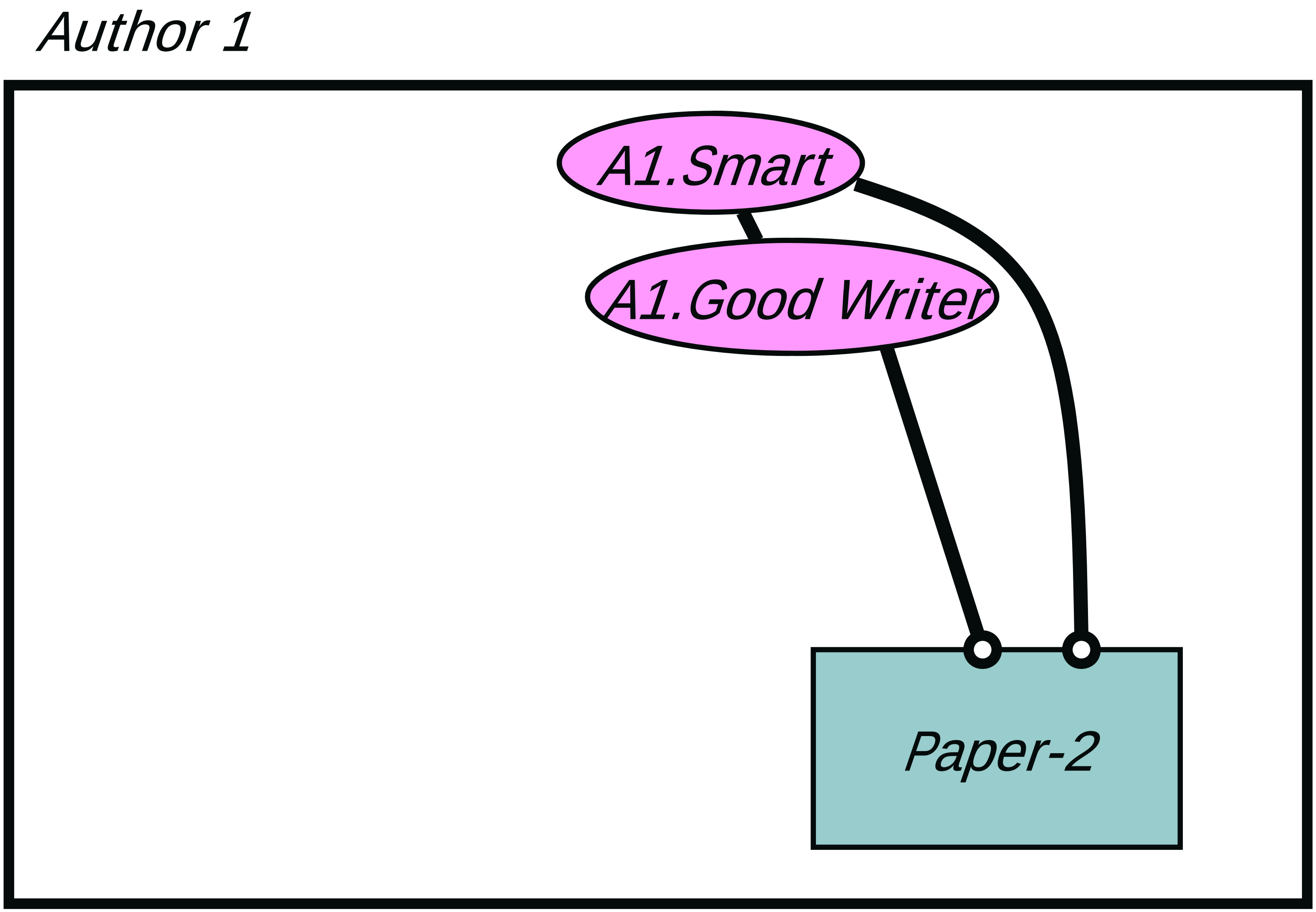
Paper 1



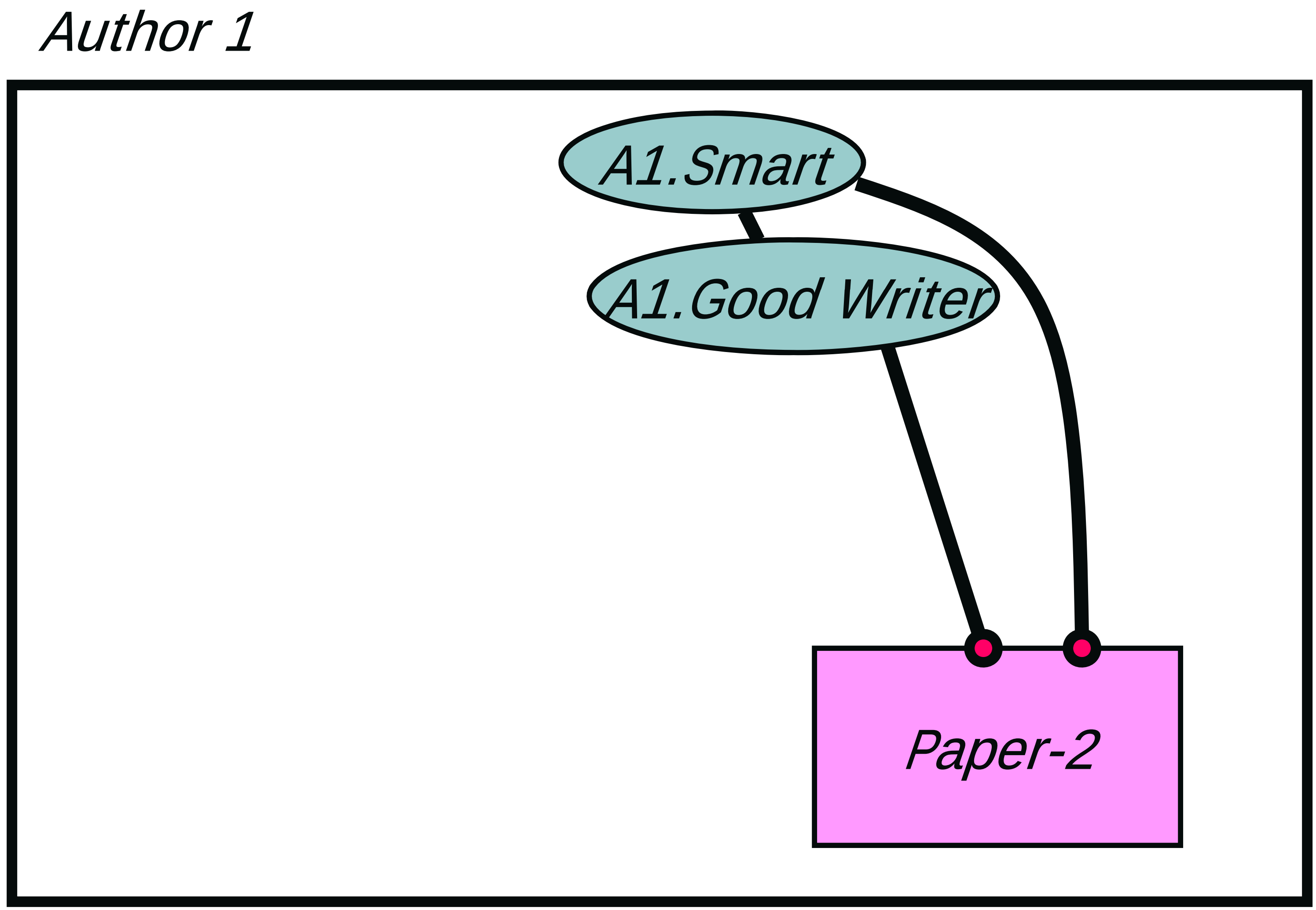
Structured Variable Elimination



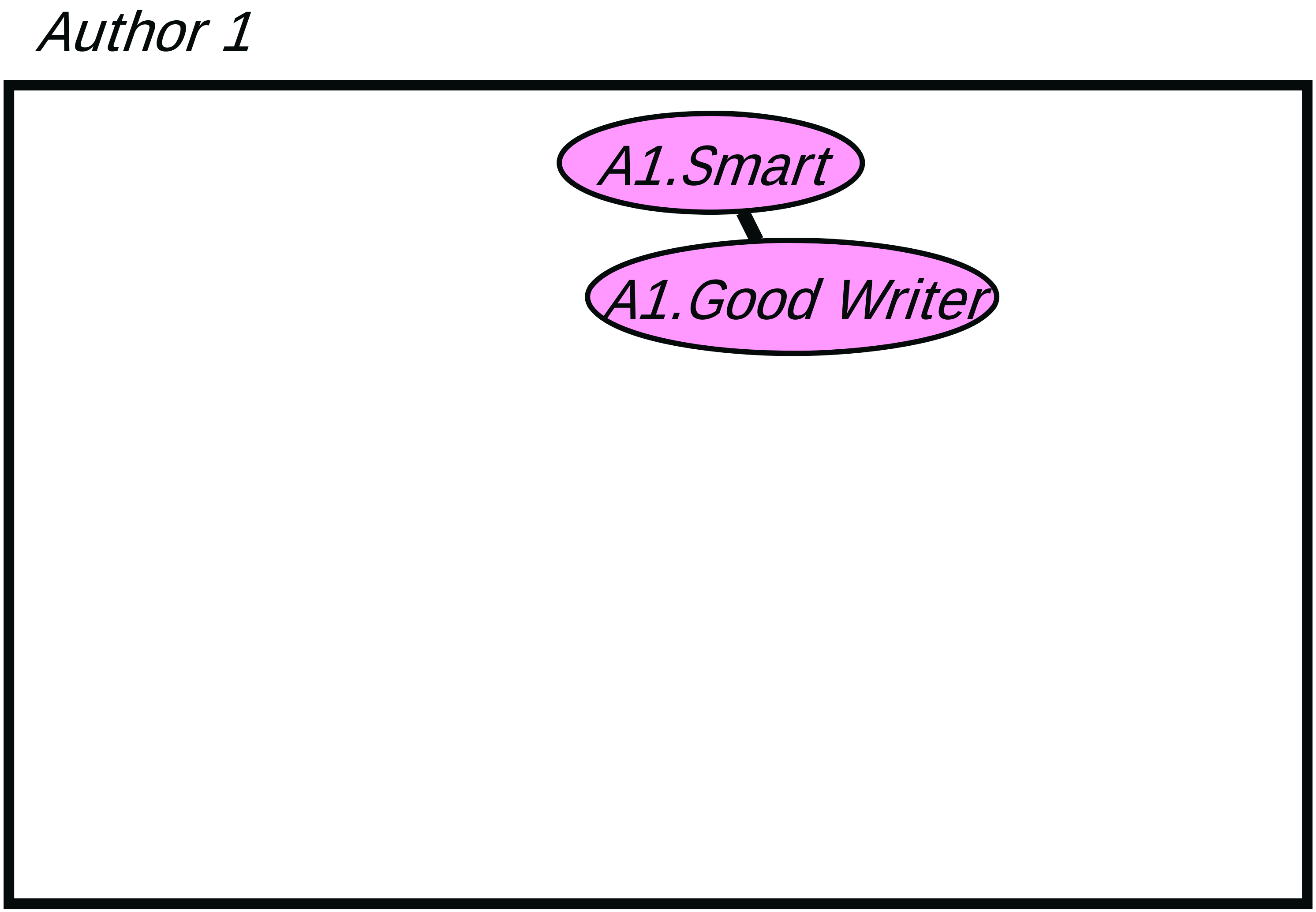
Structured Variable Elimination



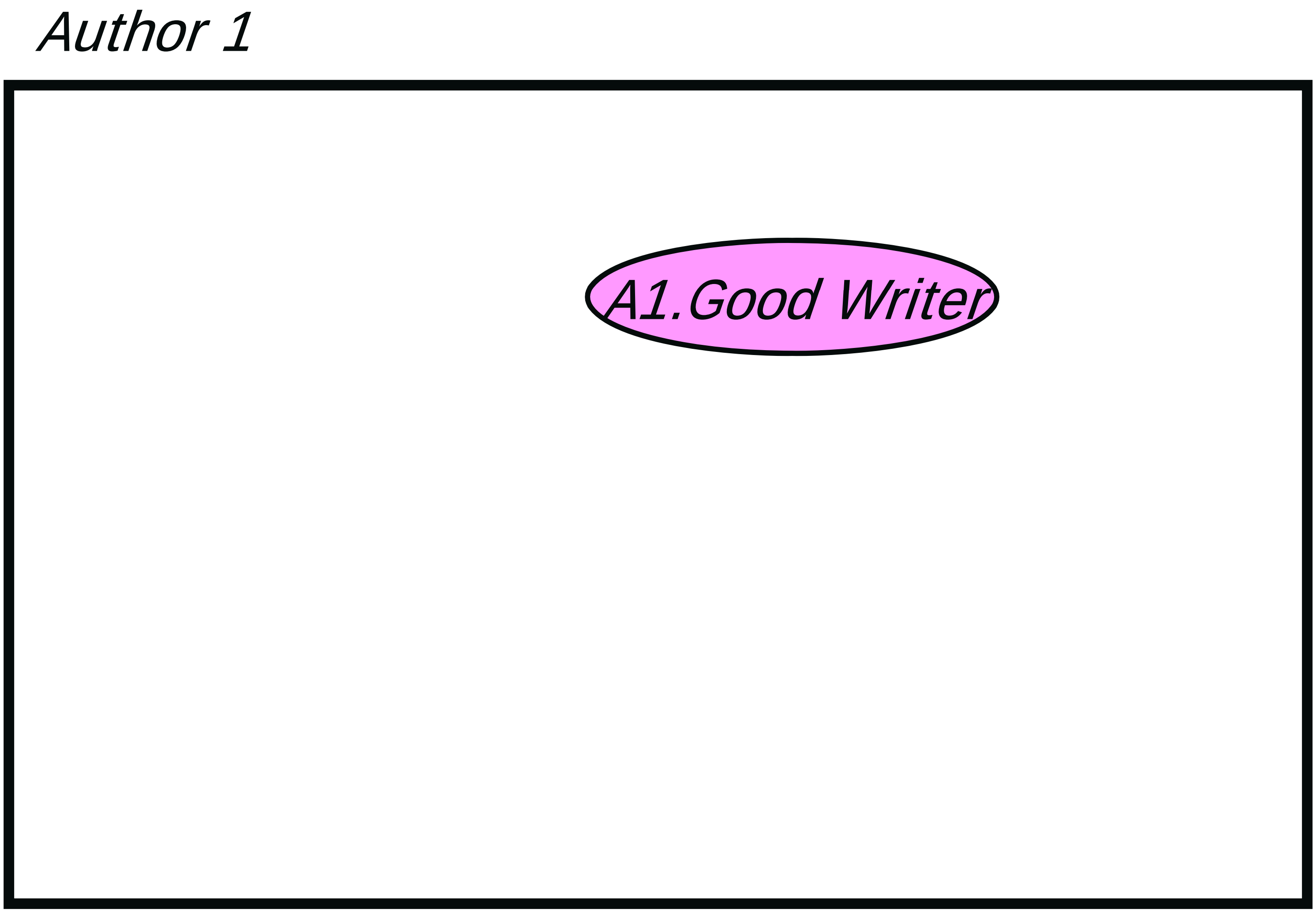
Structured Variable Elimination



Structured Variable Elimination



Structured Variable Elimination

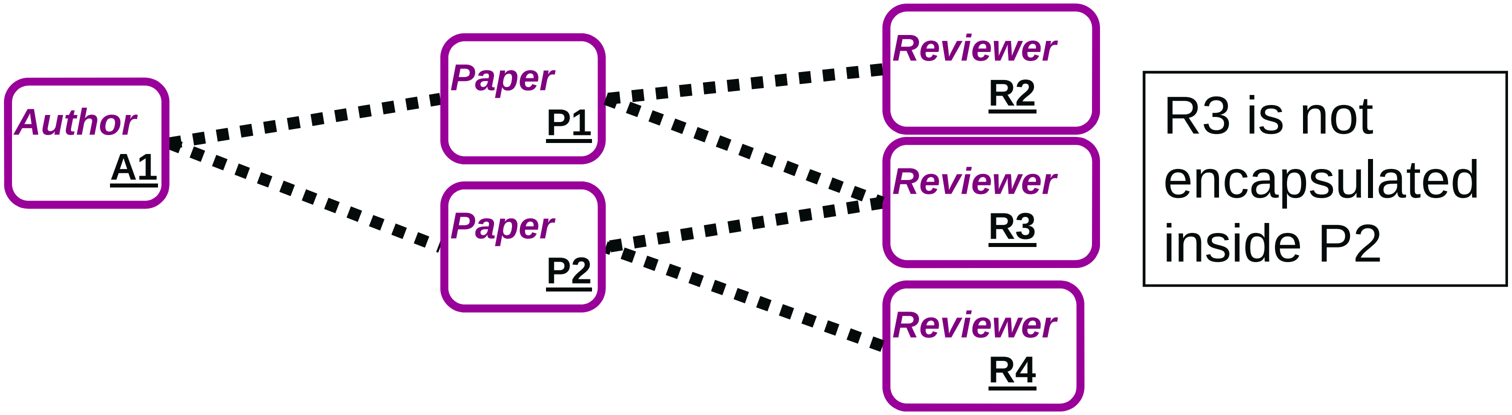


Benefits of SVE

- Structured inference leads to good elimination orderings for VE
  - interfaces are separators
    - finding good separators for large BNs is very hard
  - therefore cheaper BN inference
- Reuses computation wherever possible

Limitations of SVE

- Does not work when encapsulation breaks down

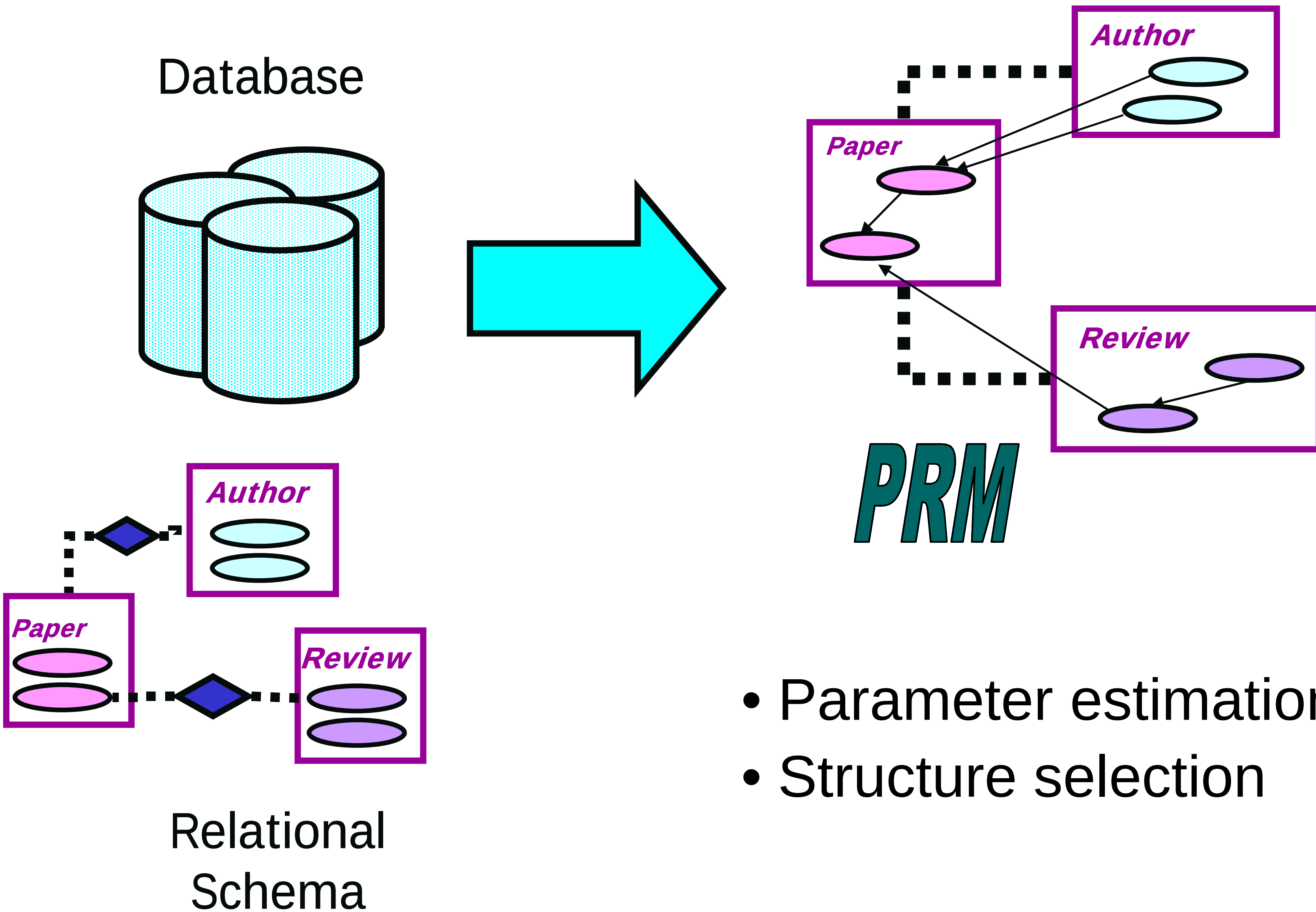


- But when we don't have specific information about the connections between objects, we can assume that encapsulation holds
  - i.e., if we know P1 has two reviewers R1 and R2 but they are not named instances, we assume R1 and R2 are encapsulated
- Cannot reuse computation when different objects have different evidence

Four SRL Approaches

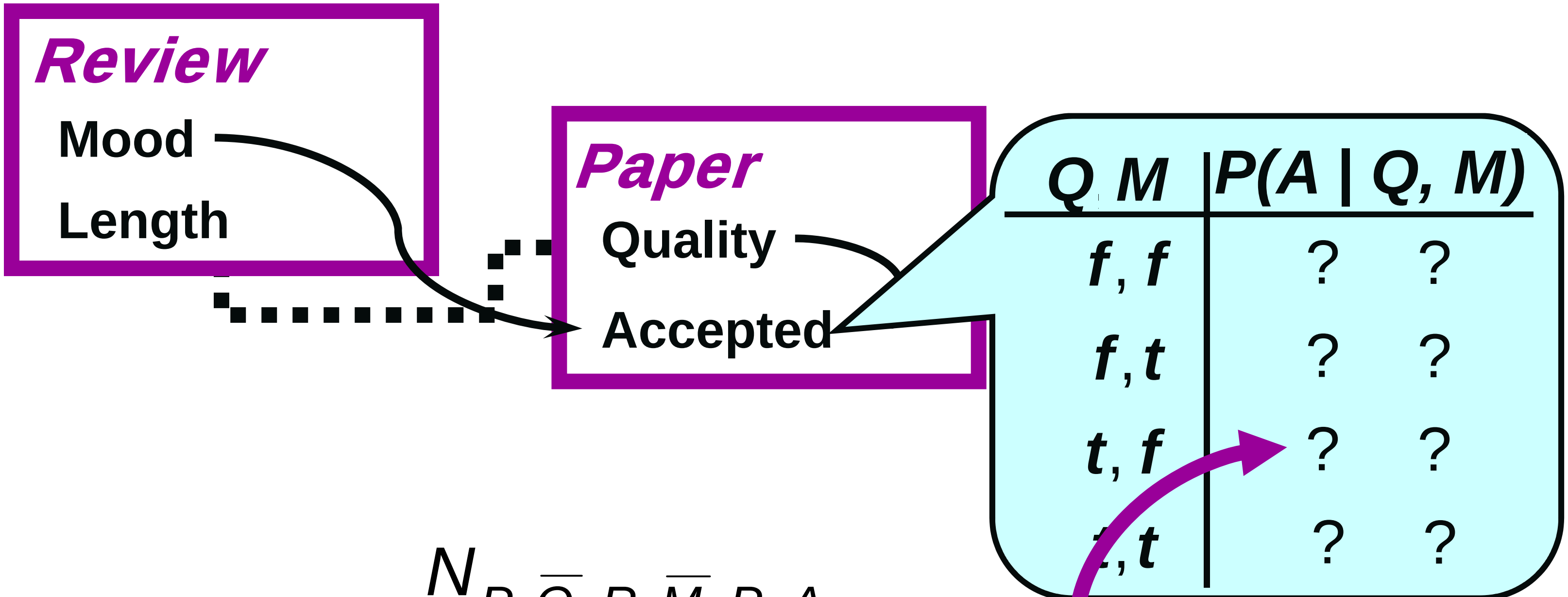
- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**
    - PRMs w/ Attribute Uncertainty
    - Inference in PRMs
    - Learning in PRMs
    - PRMs w/ Structural Uncertainty
    - PRMs w/ Class Hierarchies
- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Learning PRMs w/ AU



- Parameter estimation
- Structure selection

ML Parameter Estimation

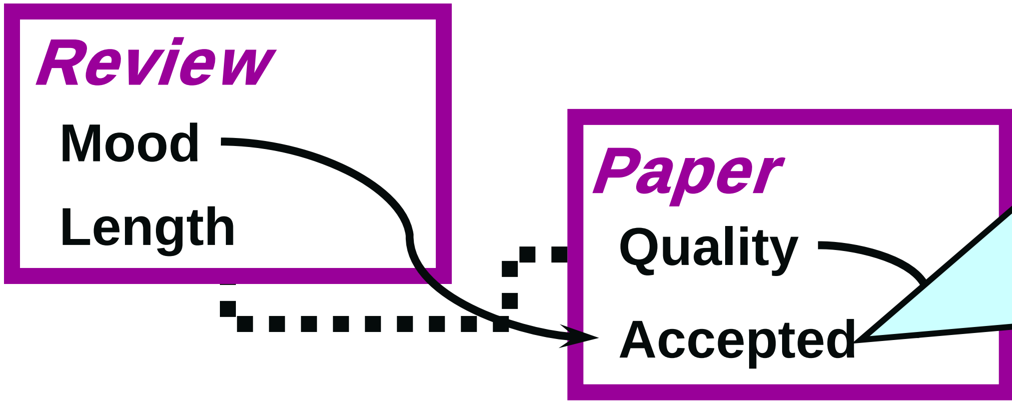


Q, M	P(A | Q, M)	
f, f	?	?
f, t	?	?
t, f	?	?
t, t	?	?

$$\theta^* = \frac{N_{P.\overline{Q}, R.\overline{M}, P.A}}{N_{P.\overline{Q}, R.\overline{M}}}$$

where $N_{P.\overline{Q}, R.\overline{M}, P.A}$ is the number of accepted, low quality papers whose reviewer was in a poor mood

ML Parameter Estimation



Q, M	P(A | Q, M)	
f, f	?	?
f, t	?	?
t, f	?	?
t, t	?	?

$$\theta^* = \frac{N_{P.\overline{Q},R.\overline{M},P.A}}{N_{P.\overline{Q},R.\overline{M}}}$$

Query for counts:

$$Count\left(\pi_{\substack{P.Quality \\ R.Mood \\ P.Accepted}}\left(\boxed{\text{Review table}} \bowtie \boxed{\text{Paper table}}\right)\right)$$

Structure Selection

- Idea:
  - define scoring function
  - do local search over legal structures

- Key Components:
  - legal models
  - scoring models
  - searching model space

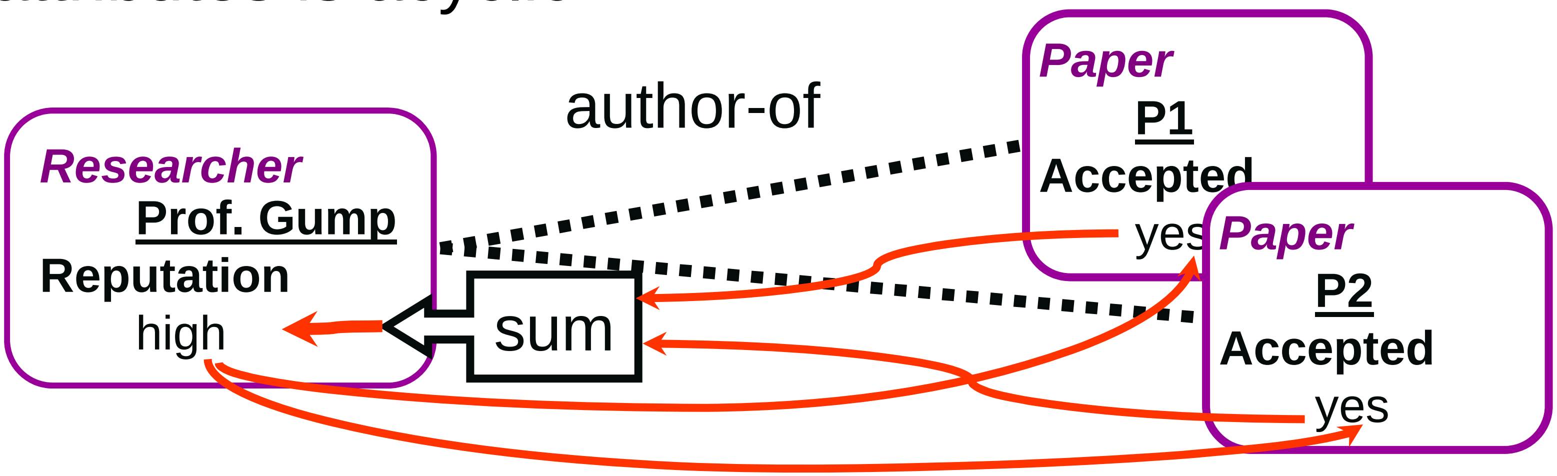
Structure Selection

- Idea:
  - define scoring function
  - do local search over legal structures

- Key Components:
  - » legal models
  - scoring models
  - searching model space

Legal Models

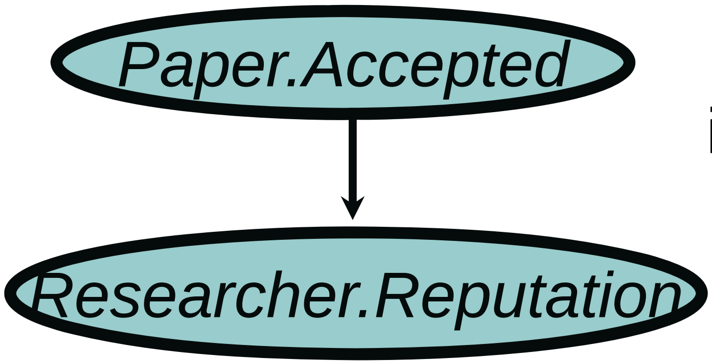
- PRM defines a coherent probability model over a skeleton σ if the dependencies between object attributes is acyclic



How do we guarantee that a PRM is acyclic for ***every*** skeleton?

Attribute Stratification

PRM dependency structure *S* ⇒ dependency graph



if *Researcher.Reputation* depends directly on *Paper.Accepted*

Attribute stratification:

dependency graph acyclic $\Rightarrow$ acyclic for any σ

Algorithm more flexible; allows certain cycles along guaranteed acyclic relations

Structure Selection

- Idea:
  - define scoring function
  - do local search over legal structures

- Key Components:
  - legal models
  - » scoring models – same as BN
  - searching model space

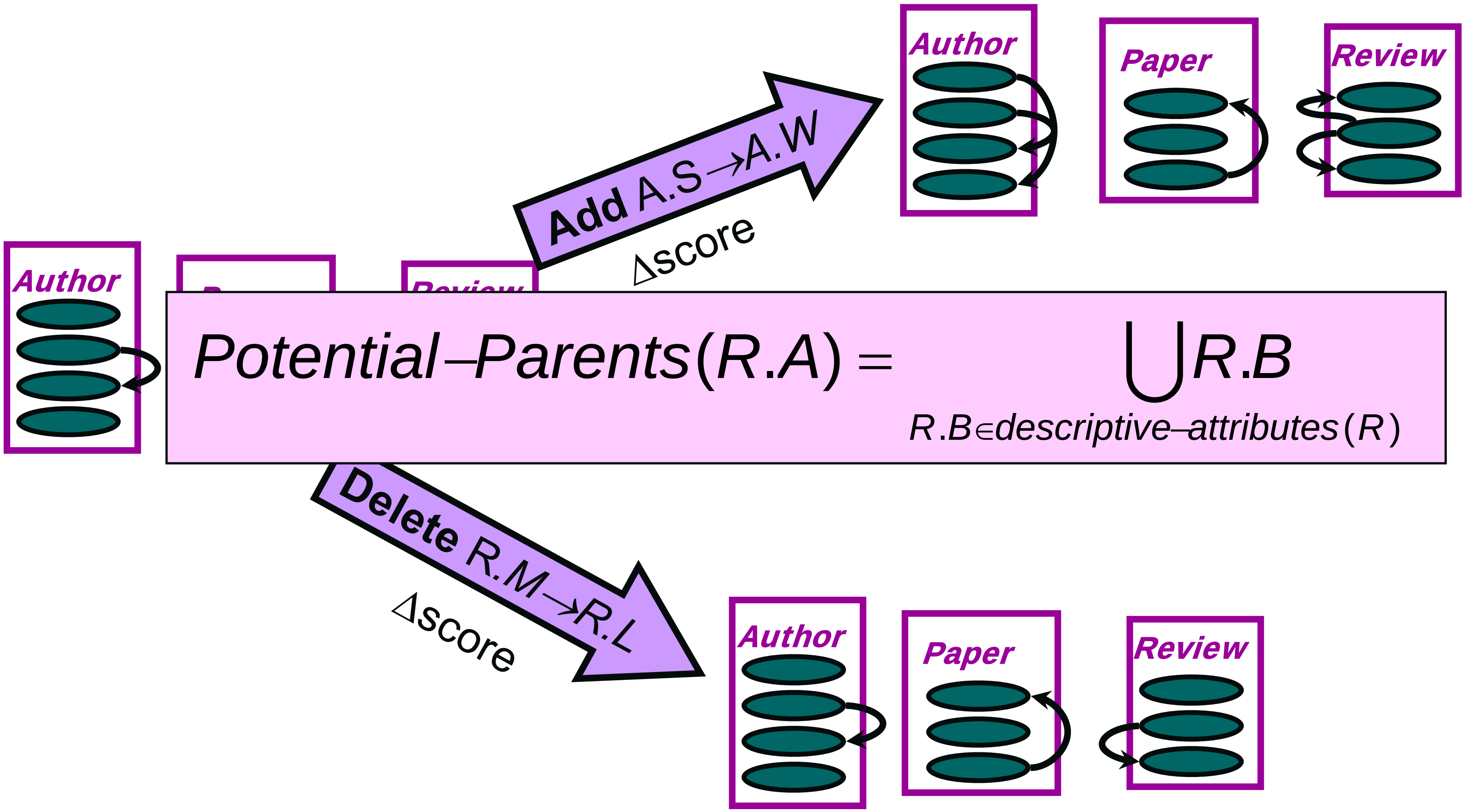
Structure Selection

- Idea:
  - define scoring function
  - do local search over legal structures

- Key Components:
  - legal models
  - scoring models
  - » searching model space

Searching Model Space

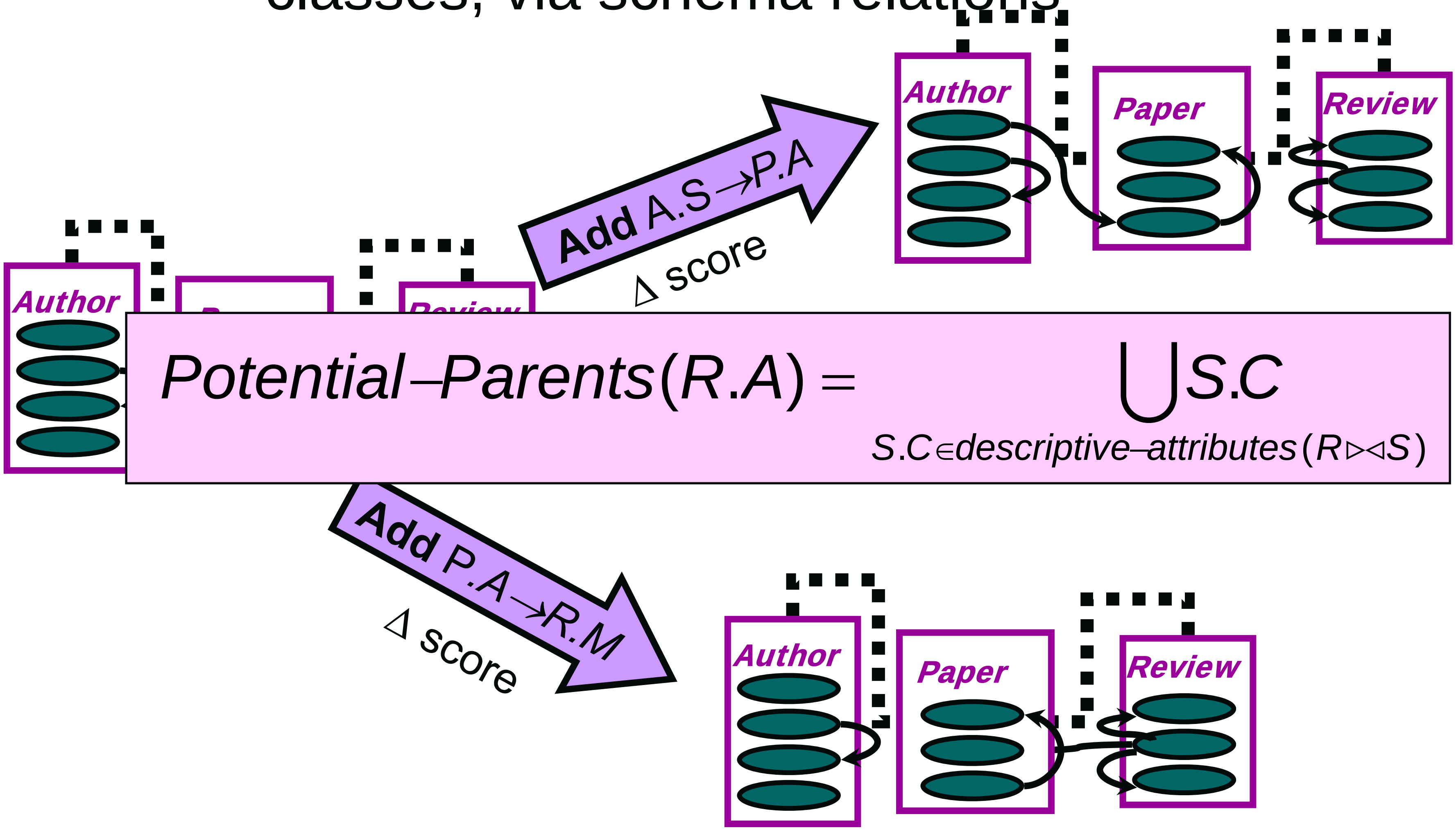
Phase 0: consider only dependencies within a class



$$Potential\text{-}Parents(R.A) = \bigcup_{R.B \in descriptive\text{-}attributes(R)} R.B$$

Phased Structure Search

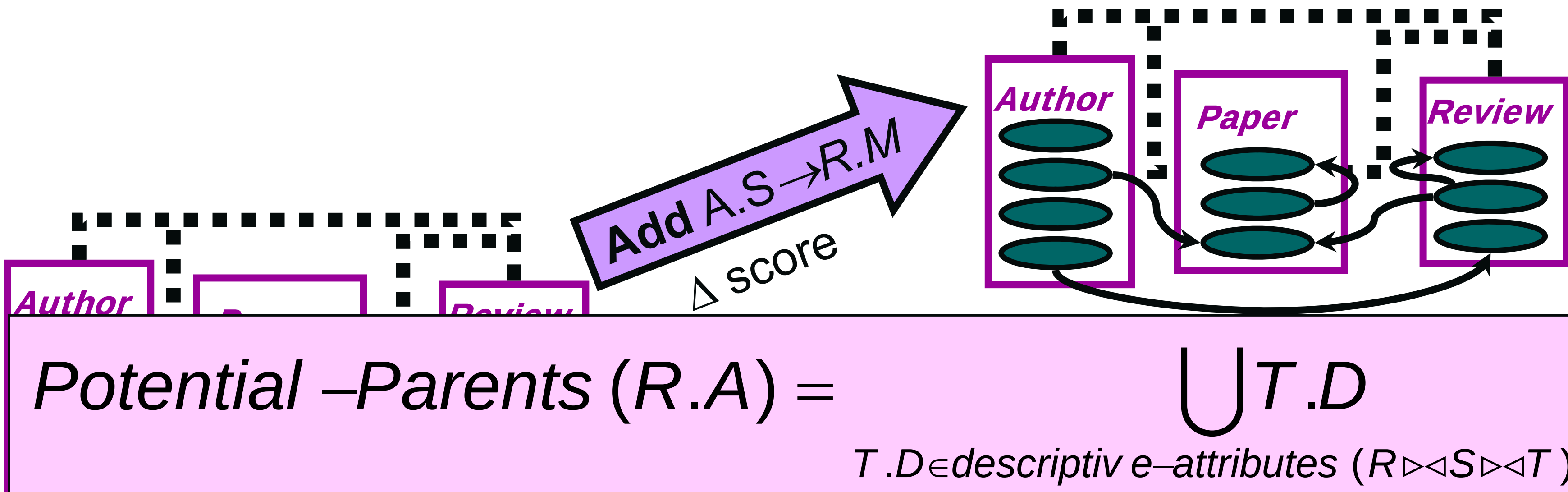
Phase 1: consider dependencies from “neighboring” classes, via schema relations



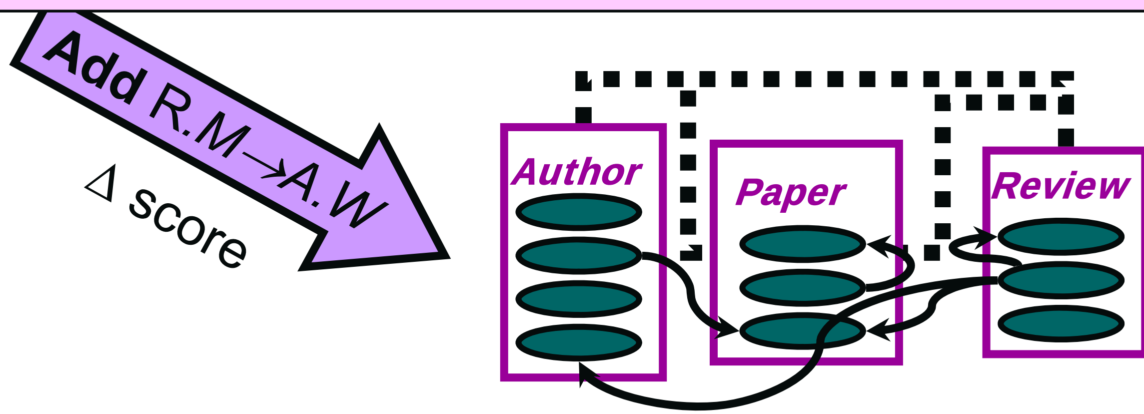
$$\mathit{Potential\text{–}Parents}(R.A) = \bigcup_{S.C \in \mathit{descriptive\text{–}attributes}(R \bowtie S)} S.C$$

Phased Structure Search

Phase 2: consider dependencies from “further” classes, via relation chains



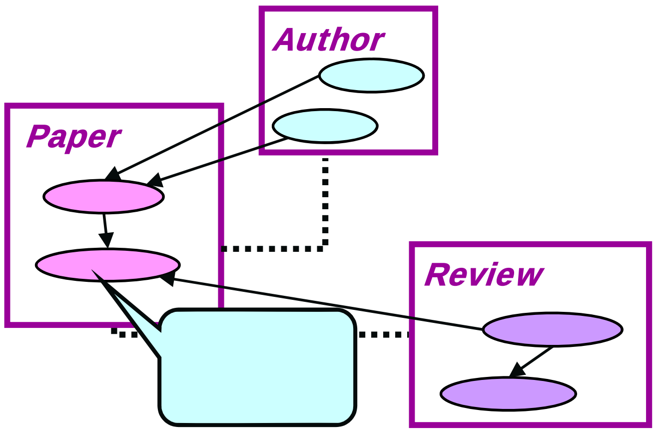
$$Potential\text{–}Parents\,(R.A) = \bigcup_{T.D \in descriptive\text{–}attributes\,(R \bowtie S \bowtie T)} T.D$$

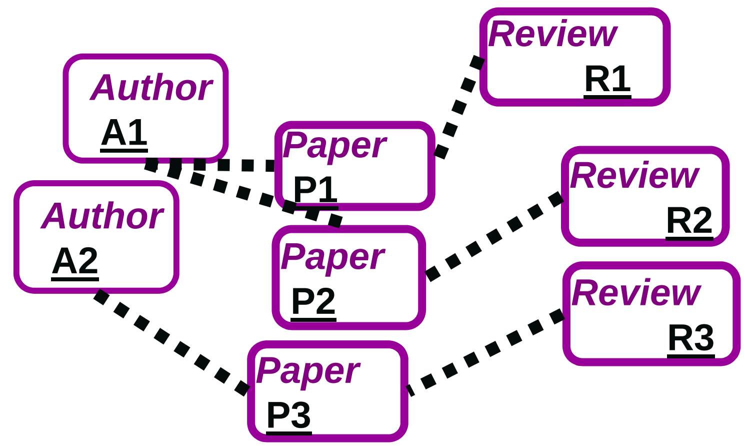


Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**
    - PRMs w/ Attribute Uncertainty
    - Inference in PRMs
    - Learning in PRMs
    - PRMs w/ Structural Uncertainty
    - PRMs w/ Class Hierarchies
- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Reminder: PRM w/ AU Semantics





PRM + relational skeleton σ =

probability distribution over completions I:

$$P(\mathrm{I} \mid \sigma, S, \Theta) = \prod_{x \in \sigma} \prod_{x.A} P(x.A \mid parents_{S,\sigma}(x.A))$$

Objects Attributes

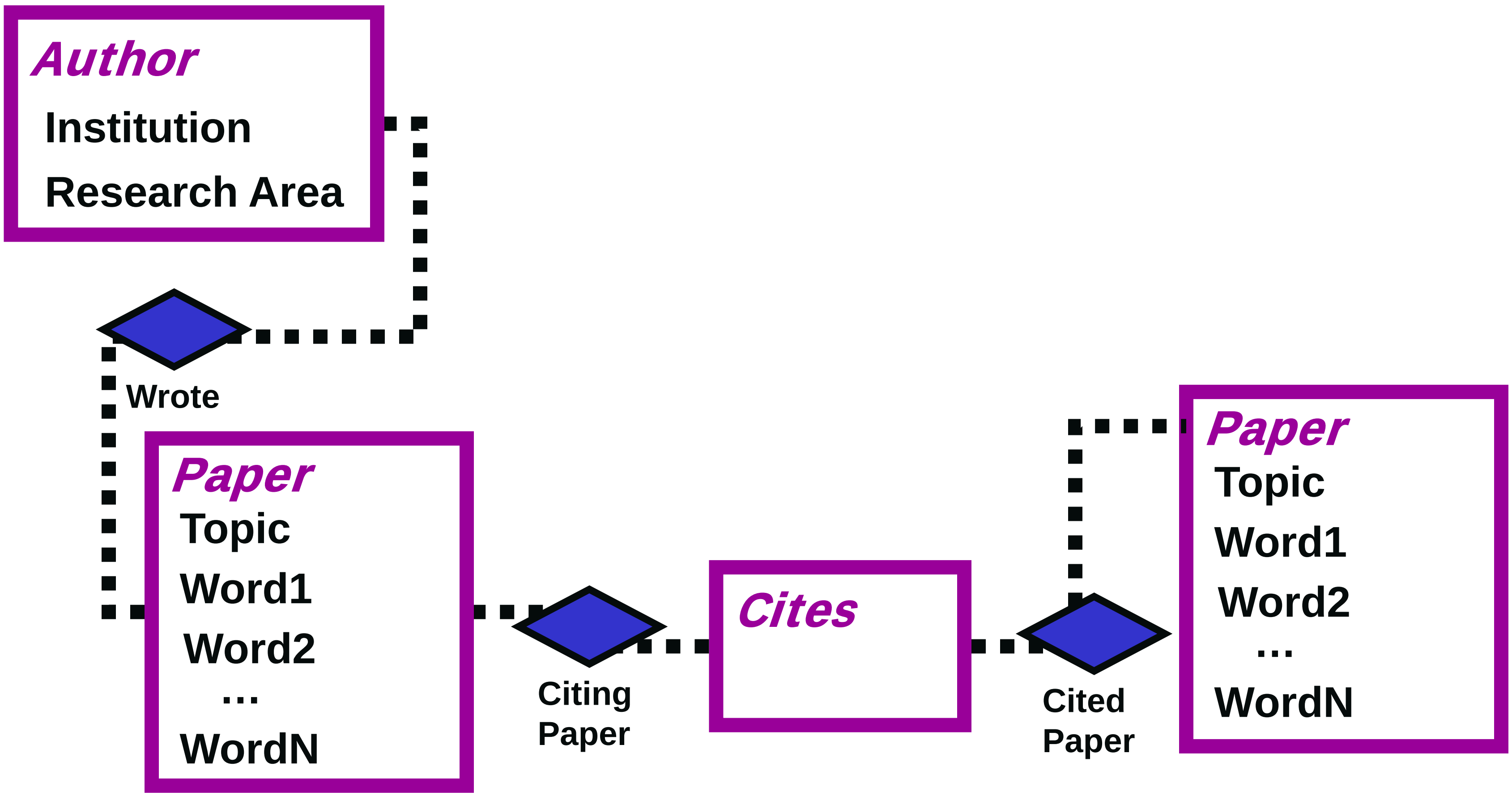
Kinds of structural uncertainty

- How many objects does an object relate to?
  - how many Authors does *Paper1* have?
- Which object is an object related to?
  - does *Paper1* cite *Paper2* or *Paper3*?
- Which class does an object belong to?
  - is *Paper1* a *JournalArticle* or a *ConferencePaper*?
- Does an object actually exist?
- Are two objects identical?

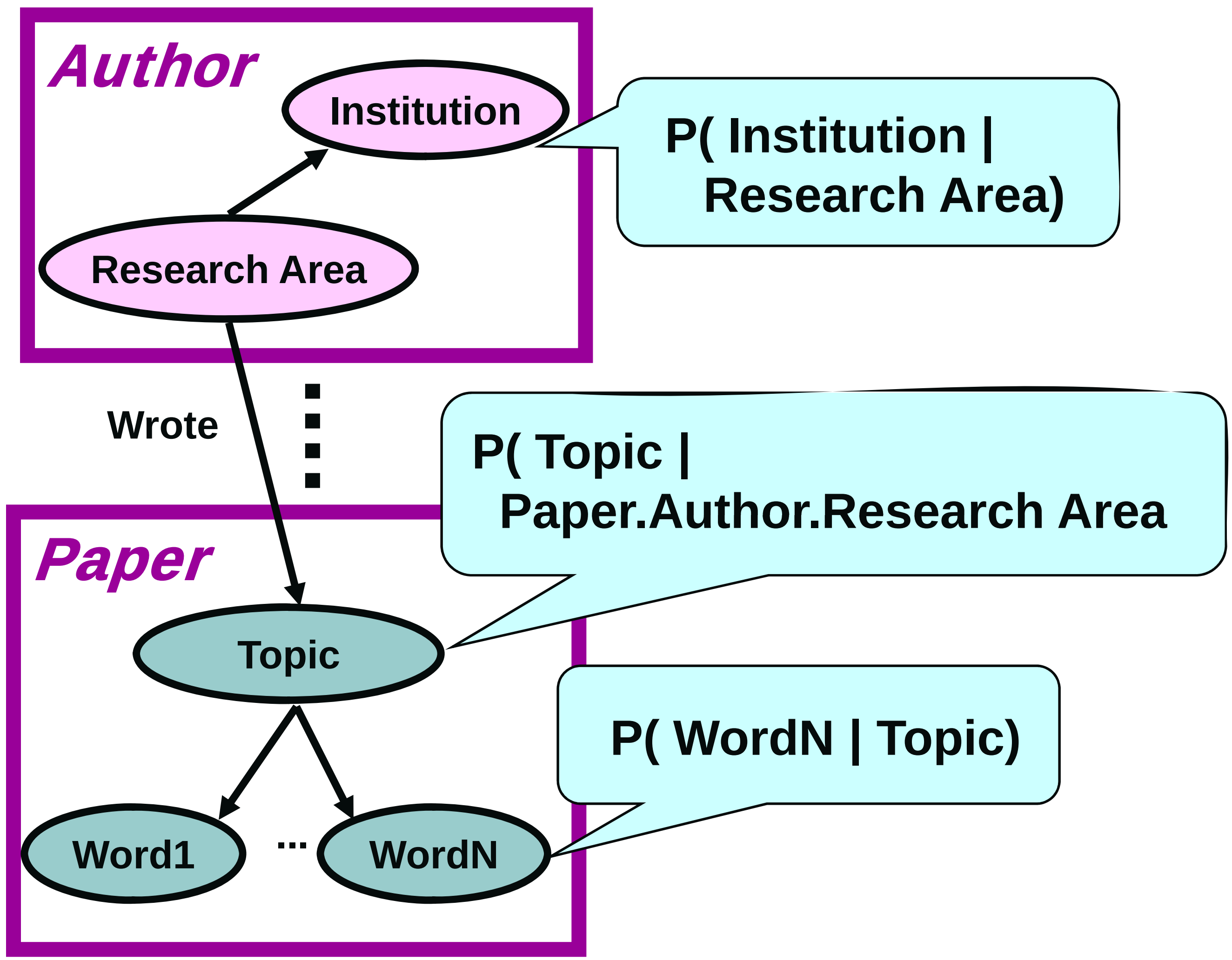
Structural Uncertainty

- Motivation: PRM with AU only well-defined when the skeleton structure is known
- May be uncertain about relational structure itself
- Construct probabilistic models of relational structure that capture structural uncertainty
- Mechanisms:
  - Reference uncertainty
  - Existence uncertainty
  - Number uncertainty
  - Type uncertainty
  - Identity uncertainty

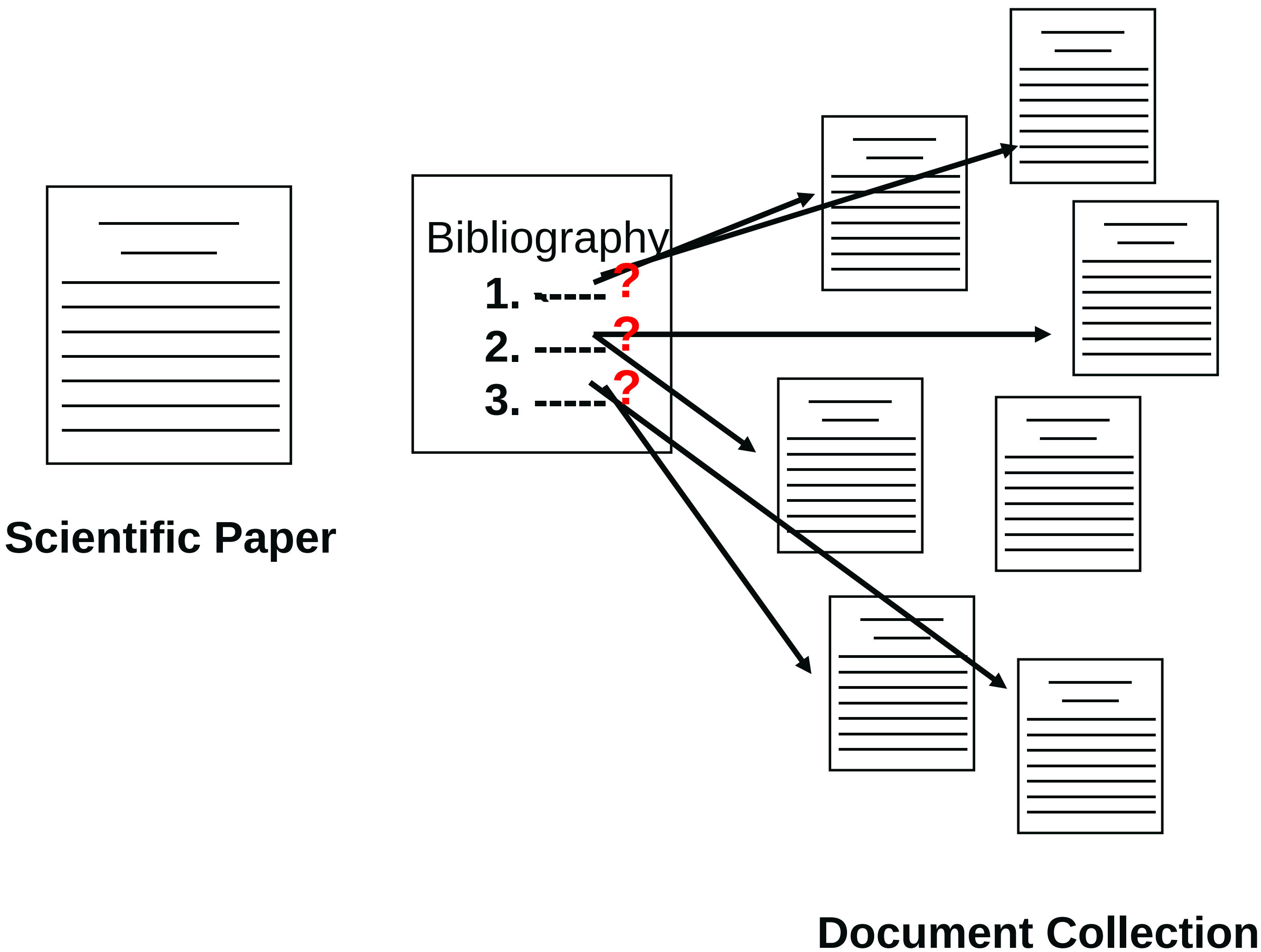
Citation Relational Schema



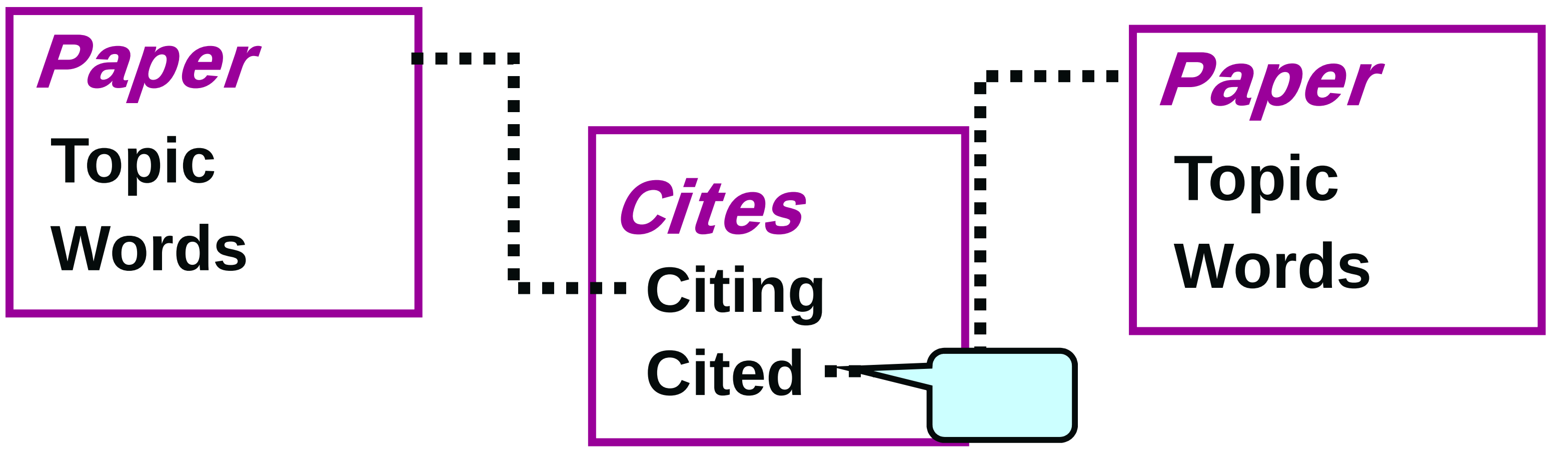
Attribute Uncertainty



Reference Uncertainty



PRM w/ Reference Uncertainty

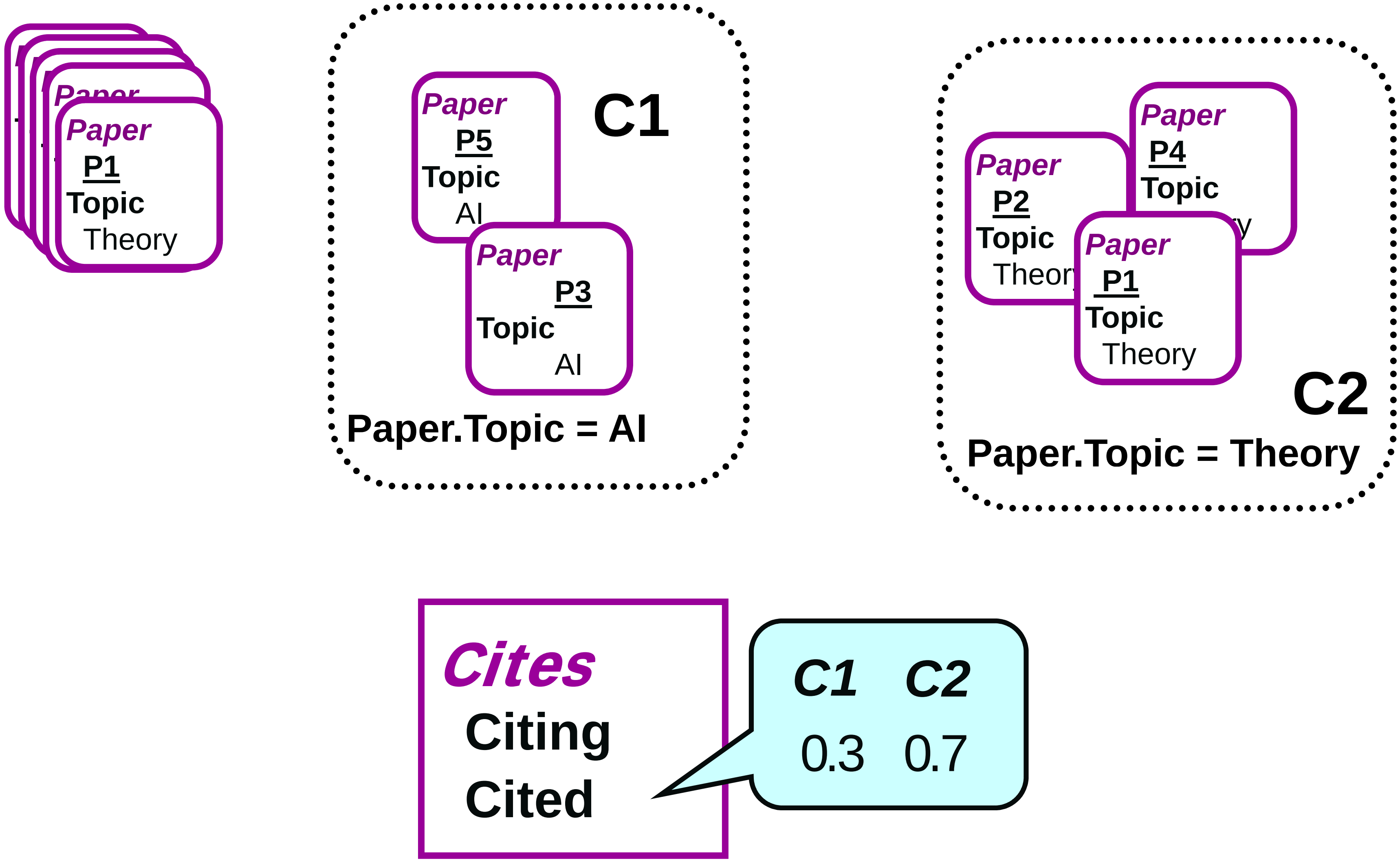


Dependency model for foreign keys

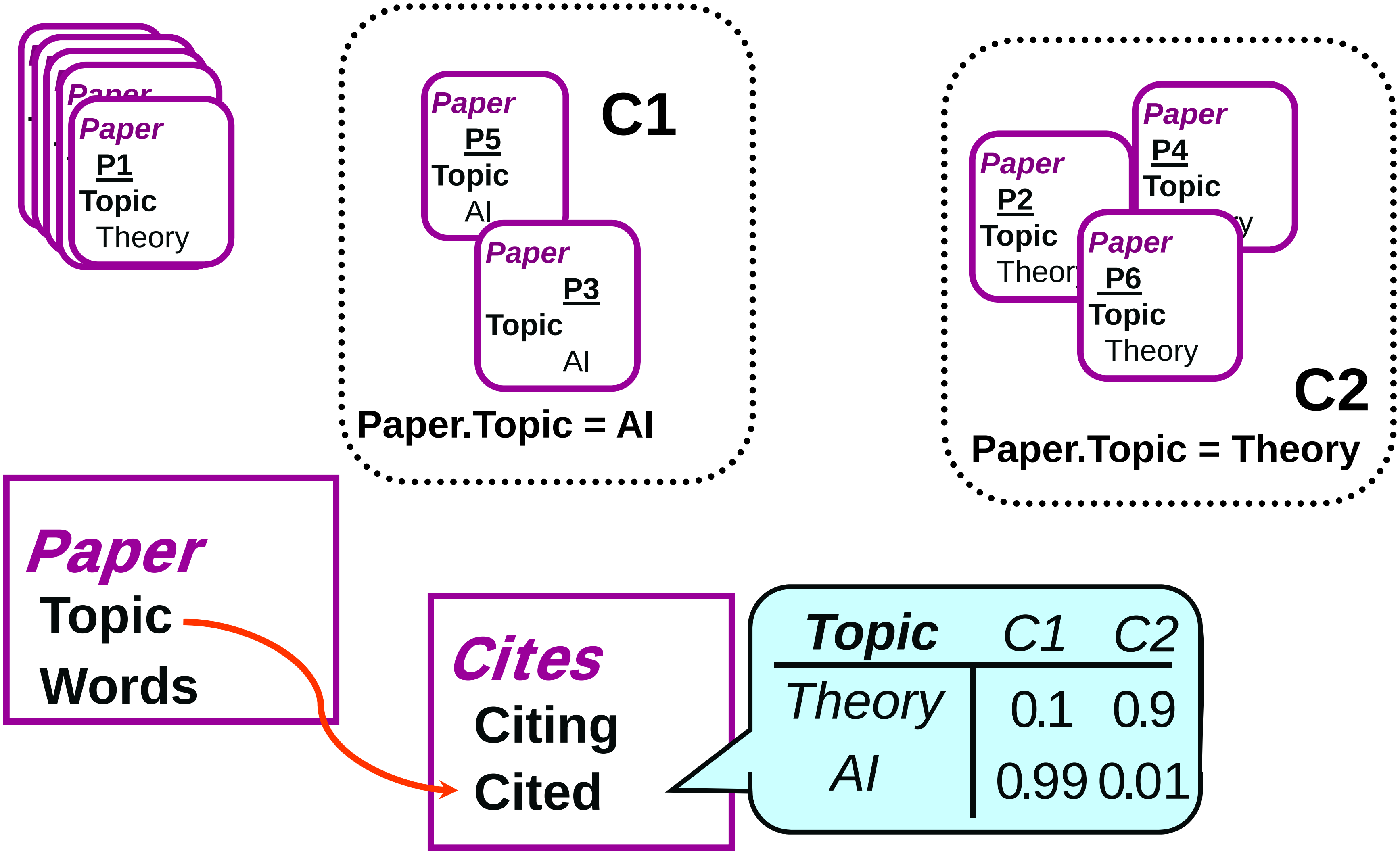
Naïve Approach: multinomial over primary key

- noncompact
- limits ability to generalize

Reference Uncertainty Example

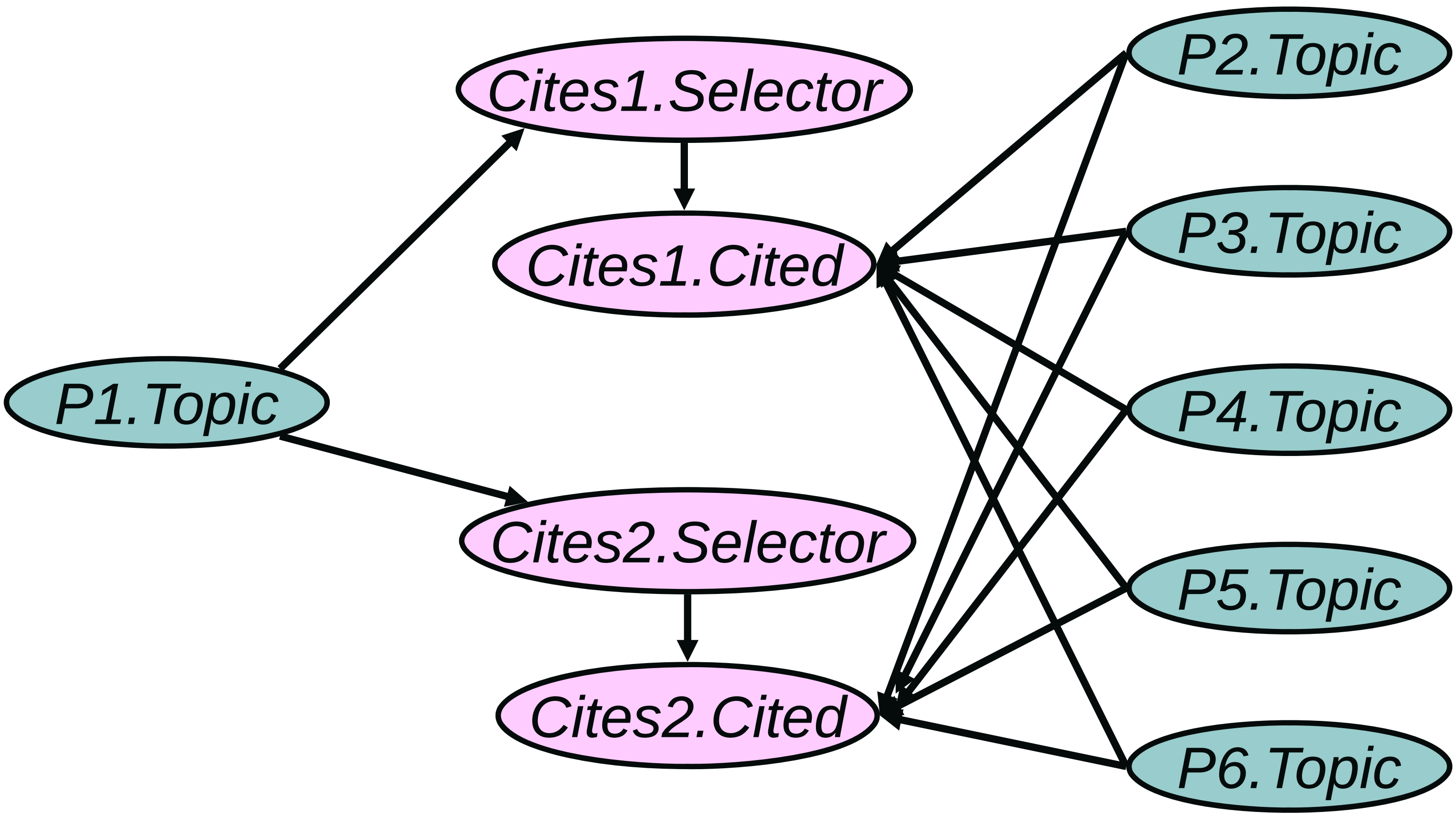


Reference Uncertainty Example



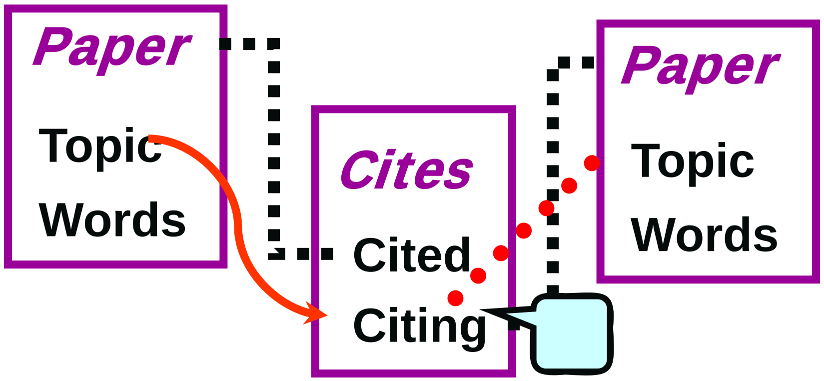
Topic	*C1*	*C2*
Theory	0.1	0.9
AI	0.99	0.01

Introduce Selector RVs

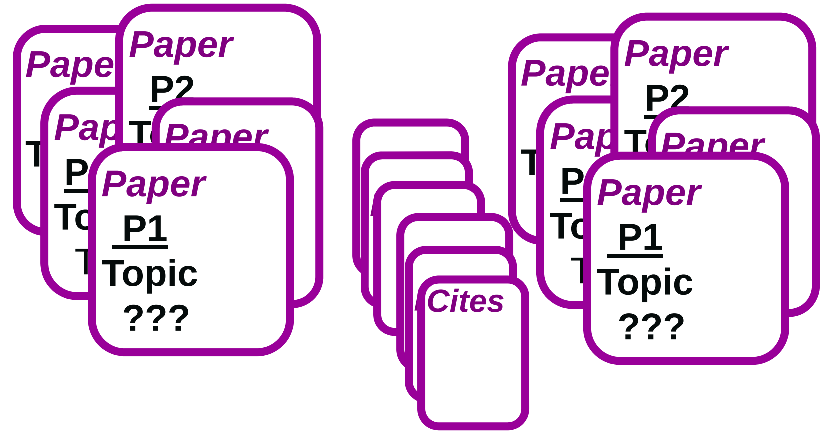


Introduce Selector RV, whose domain is {C1,C2}
The distribution over Cited depends on all of the topics, and the selector

PRMs w/ RU Semantics



PRM RU



entity skeleton σ

PRM-RU + *entity skeleton* σ

$\Rightarrow$ probability distribution over full instantiations I

Learning *PRMs w/ RU*

- **Idea:**
  - **define scoring function**
  - **do phased local search over legal structures**

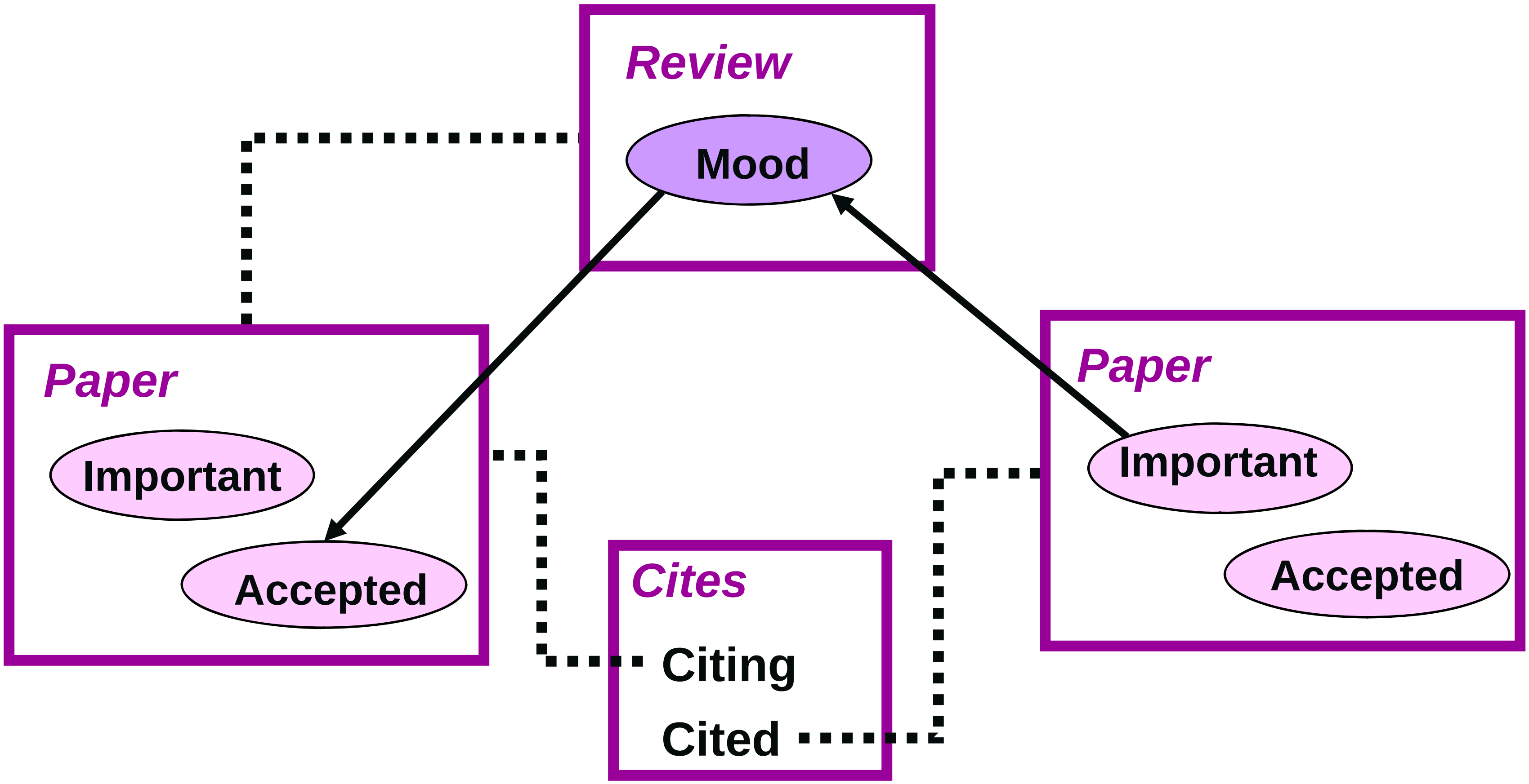
- **Key Components:**
  - **legal models**

    model new dependencies
  - **scoring models**

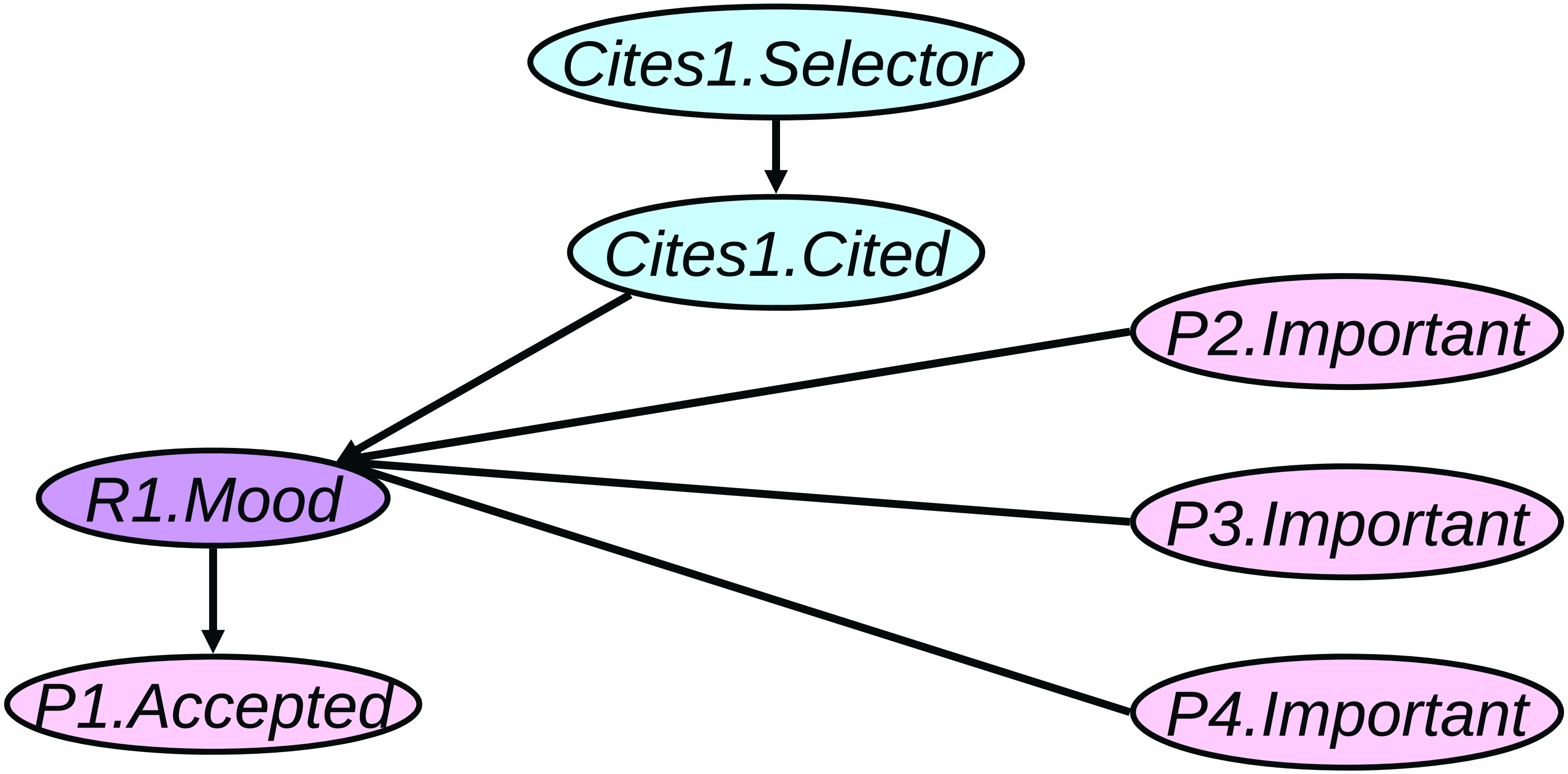
    unchanged
  - **searching model space**

    new operators

Legal Models

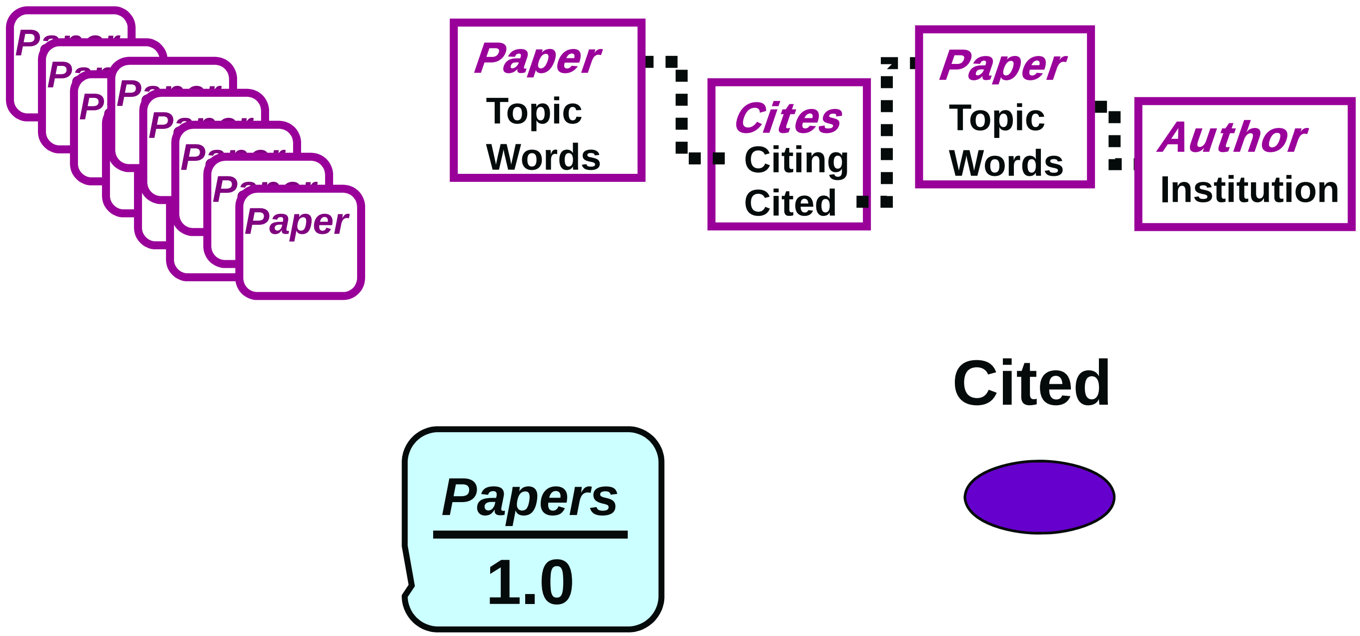


Legal Models

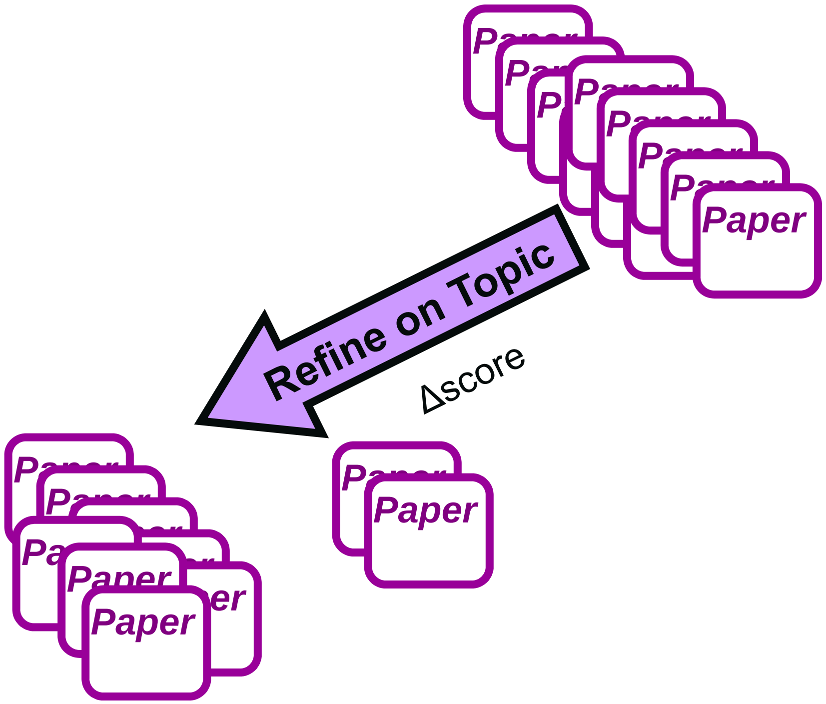


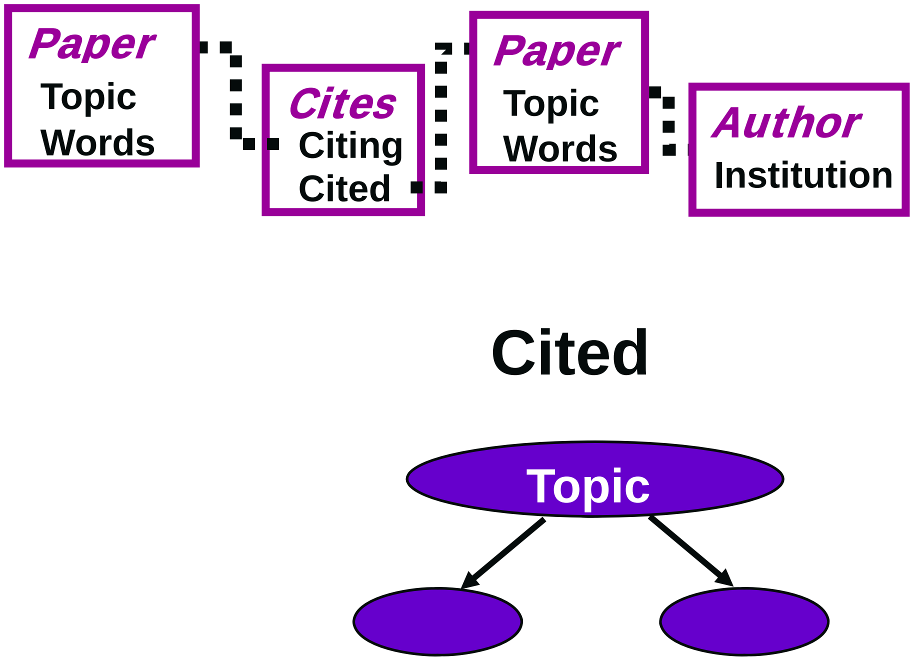
When a node's parent is defined using an uncertain relation, the reference RV must be a parent of the node as well.

Structure Search

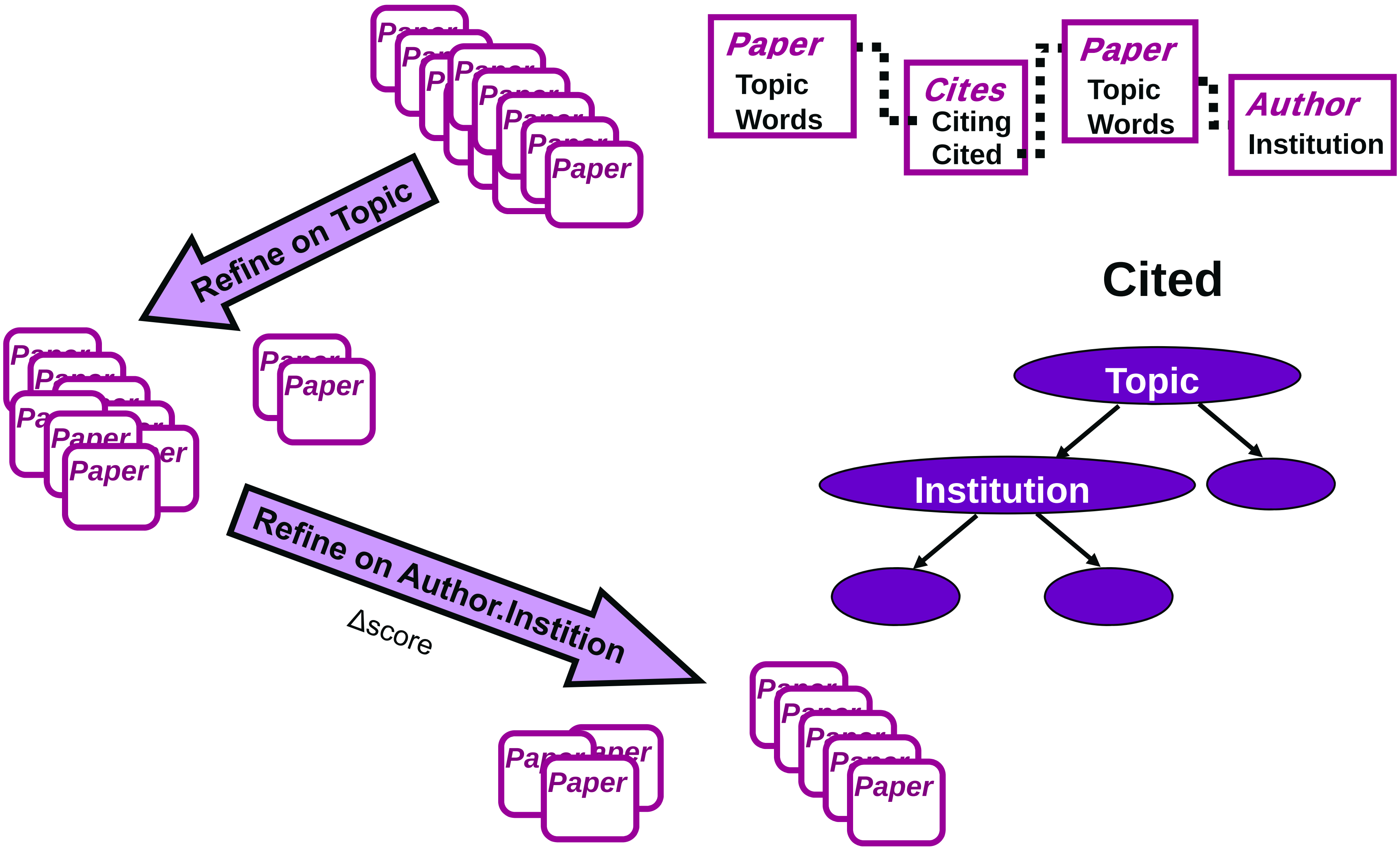


Structure Search: New Operators





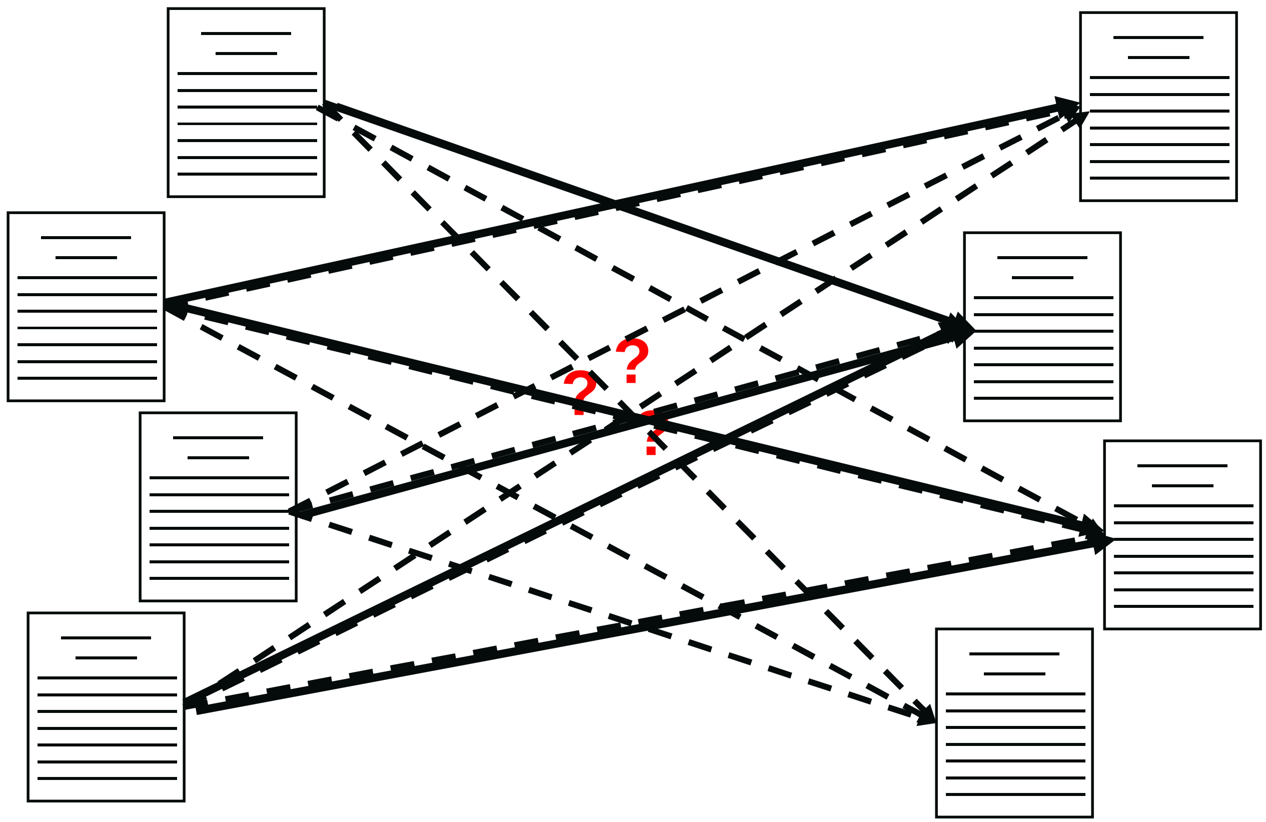
Structure Search: New Operators



PRMs w/ RU Summary

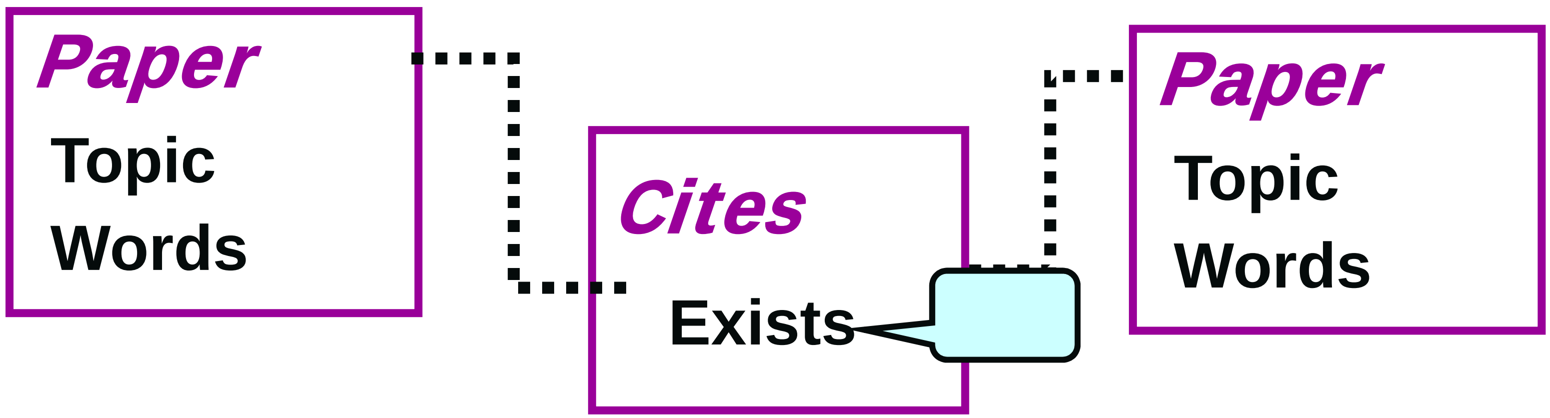
- Define semantics for uncertainty over which entities are related to each other
- Search now includes operators **Refine** and **Abstract** for constructing foreign-key dependency model
- Provides one simple mechanism for link uncertainty

Existence Uncertainty



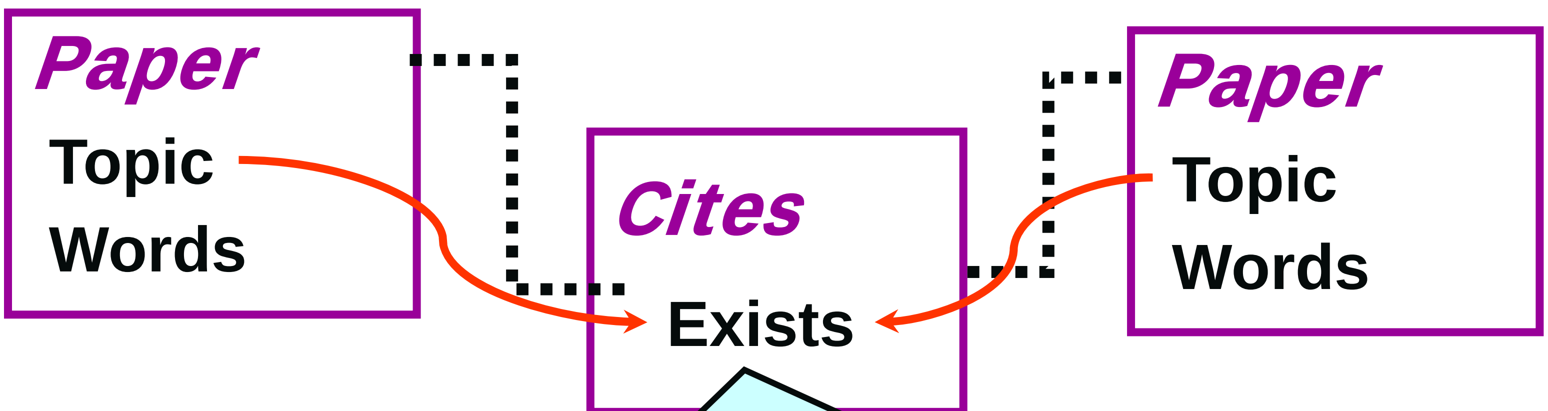
Document Collection **Document Collection**

PRM w/ Exists Uncertainty



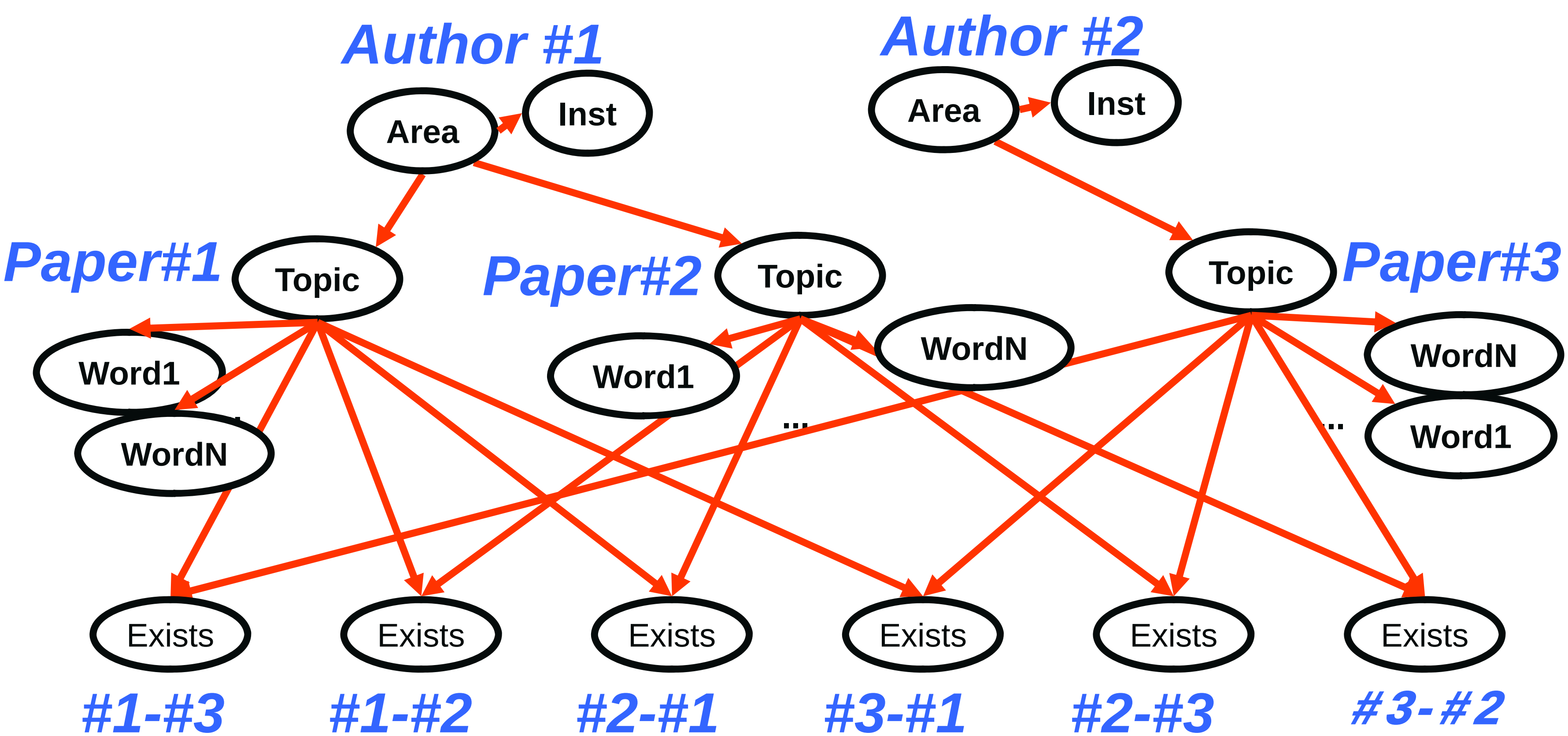
Dependency model for existence of relationship

Exists Uncertainty Example

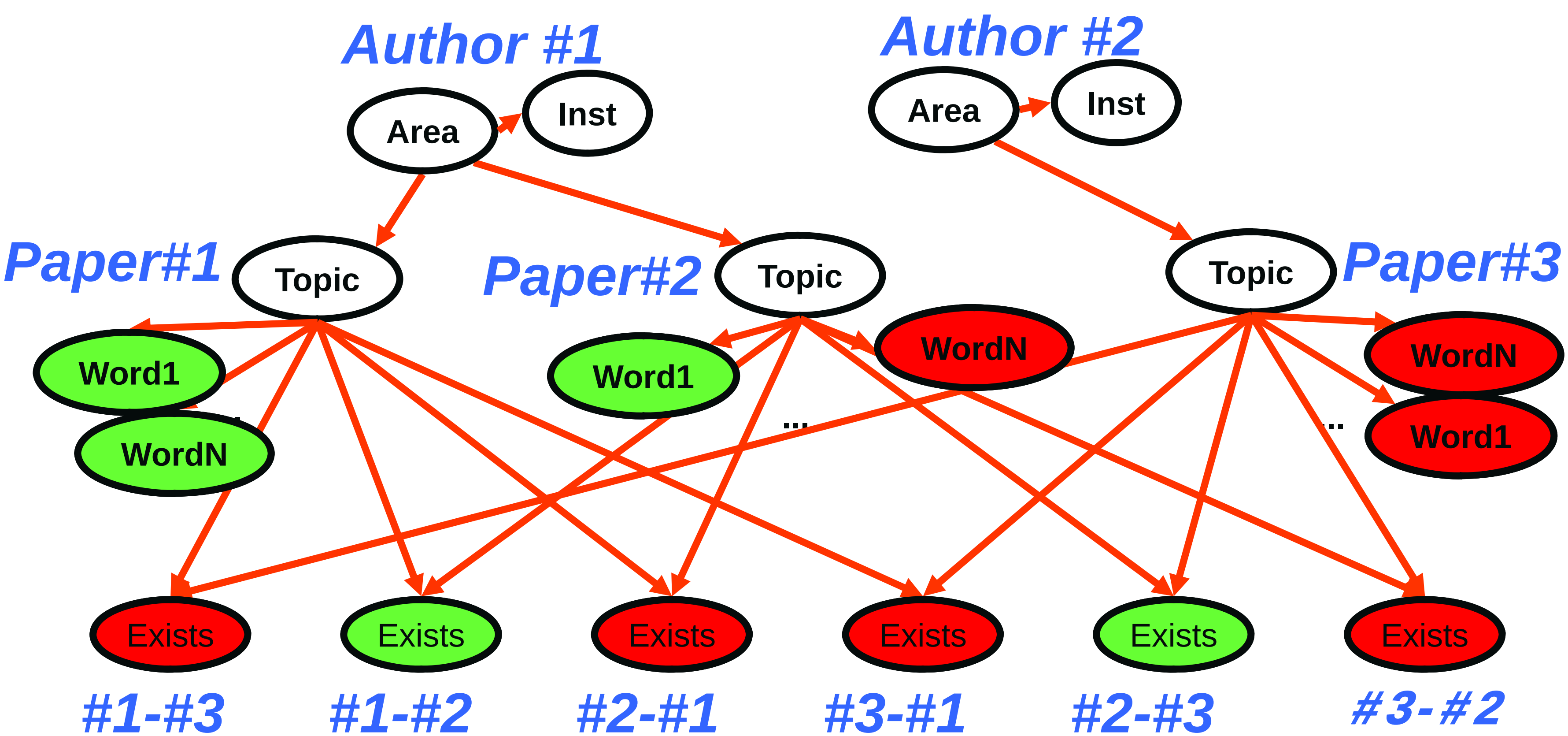


Citer.Topic	Cited.Topic	*False*	*True*
Theory	*Theory*	0.995	0005
Theory	*AI*	0.999	0001
AI	*Theory*	0.997	0003
AI	*AI*	0.993	0008

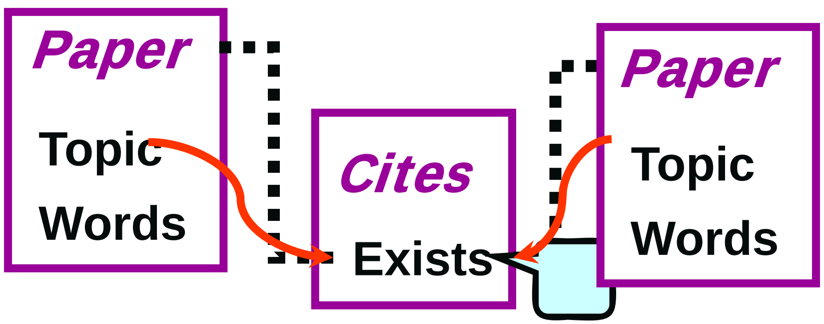
Introduce Exists RVs



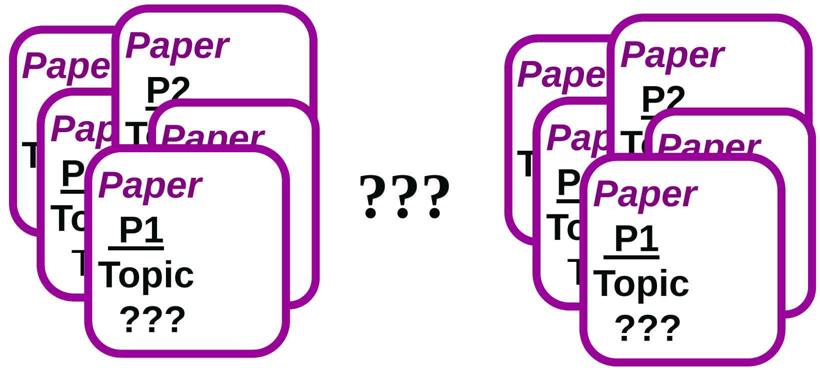
Introduce Exists RVs



PRMs w/ EU Semantics



PRM EU



object skeleton σ

PRM-EU + *object skeleton* σ

$\Rightarrow$ probability distribution over full instantiations $\mathcal{I}$

Learning ***PRMs w/ EU***

- **Idea:**
  - **define scoring function**
  - **do phased local search over legal structures**

- **Key Components:**
  - **legal models**

    model new dependencies
  - **scoring models**

    unchanged
  - **searching model space**

    unchanged

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - **Frame-based Directed Models**
    - PRMs w/ Attribute Uncertainty
    - Inference in PRMs
    - Learning in PRMs
    - PRMs w/ Structural Uncertainty
    - PRMs w/ Class Hierarchies
- Undirected Approaches
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - Rule-based Undirected Models

PRMs with classes

- Relations organized in a class hierarchy

Venue

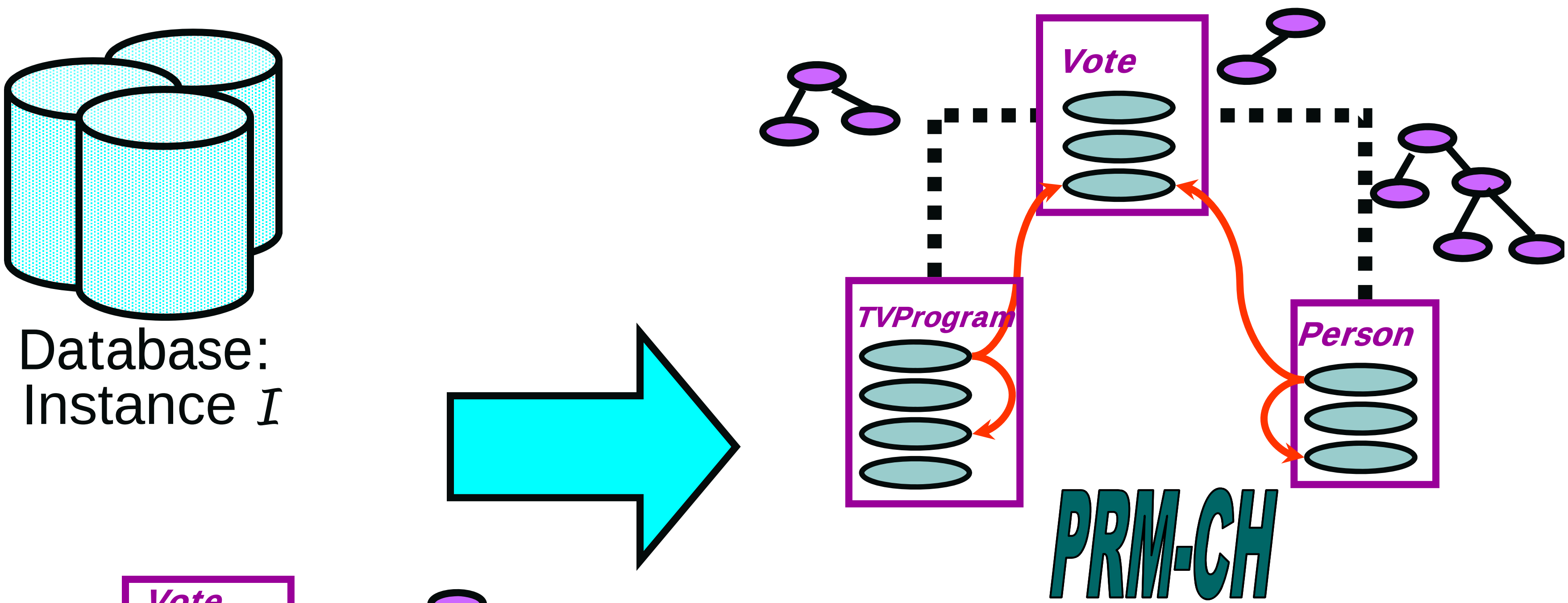
Journal   Conference

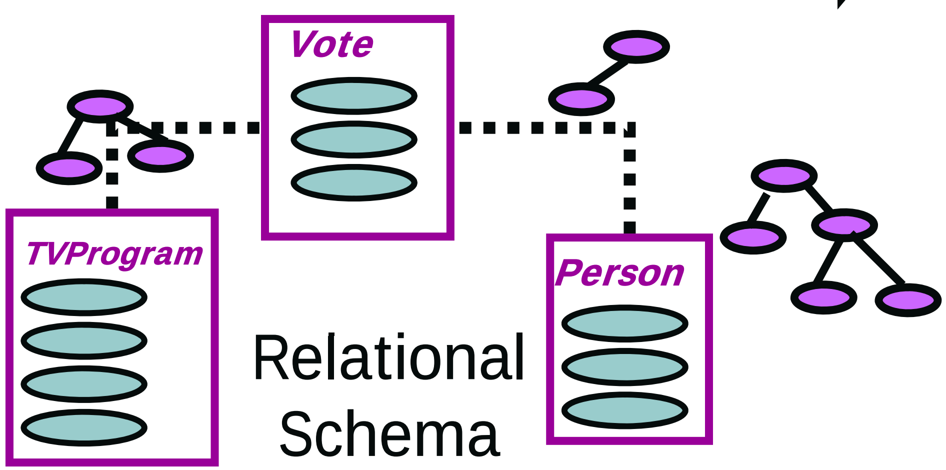
- Subclasses inherit their probability model from superclasses
- Instances are a special case of subclasses of size 1
- As you descend through the class hierarchy, you can have richer dependency models
  - e.g. cannot say **Accepted(P1) <- Accepted(P2)** (cyclic)
  - but can say **Accepted(JournalP1) <- Accepted(ConfP2)**

Type Uncertainty

- Is *1st-Venue* a *Journal* or *Conference* ?
- Create *1st-Journal* and *1st-Conference* objects
- Introduce *Type(1st-Venue)* variable with possible values *Journal* and *Conference*
- Make *1st-Venue* equal to *1st-Journal* or *1st-Conference* according to value of *Type(1st-Venue)*

Learning PRM-CHs





- Class hierarchy provided
- Learn class hierarchy

Learning *PRMs w/ CH*

- **Idea:**
  - **define scoring function**
  - **do phased local search over legal structures**

- **Key Components:**
  - **legal models**

    model new dependencies

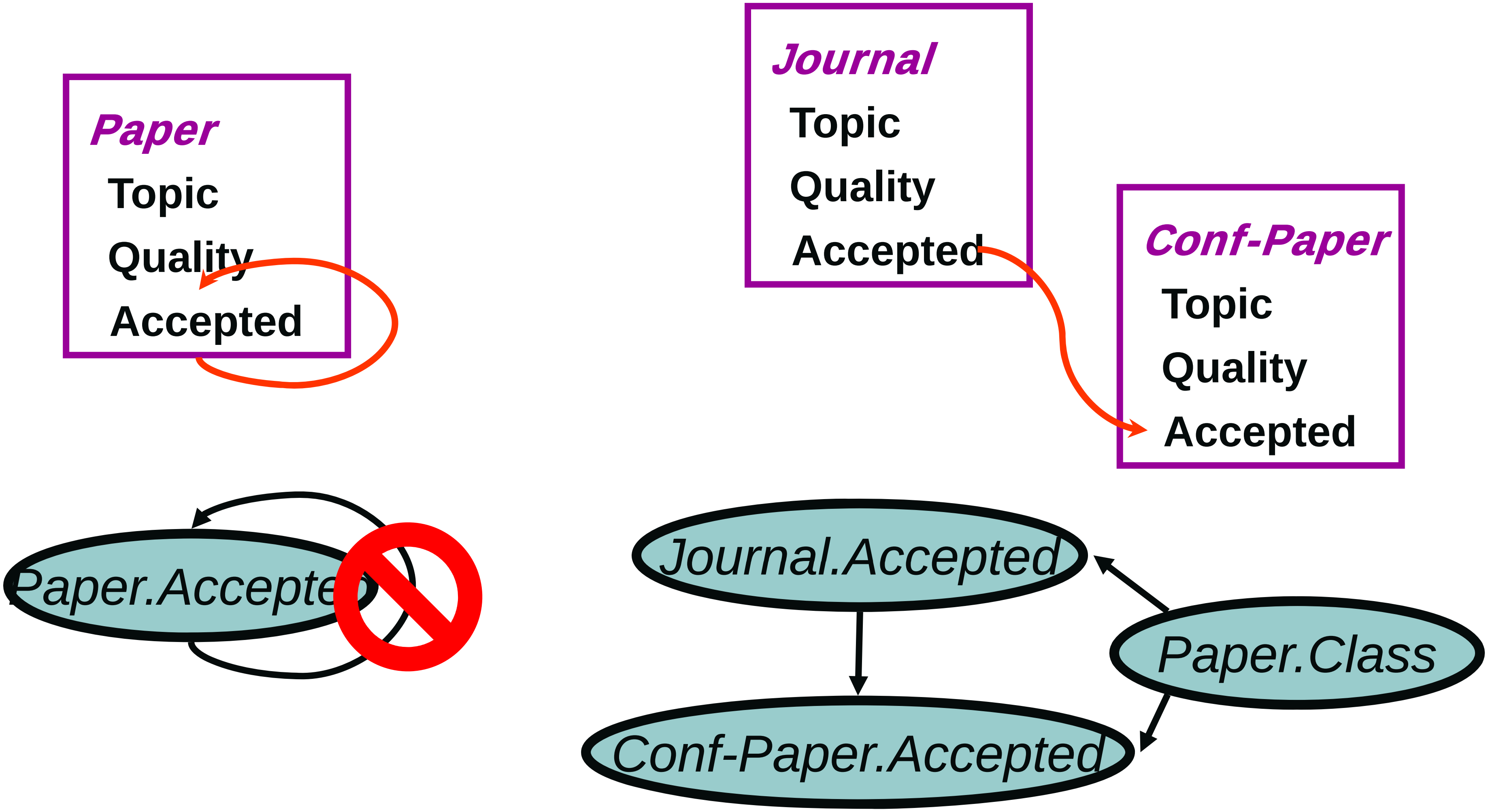
  - **scoring models**

    unchanged

  - **searching model space**

    new operators

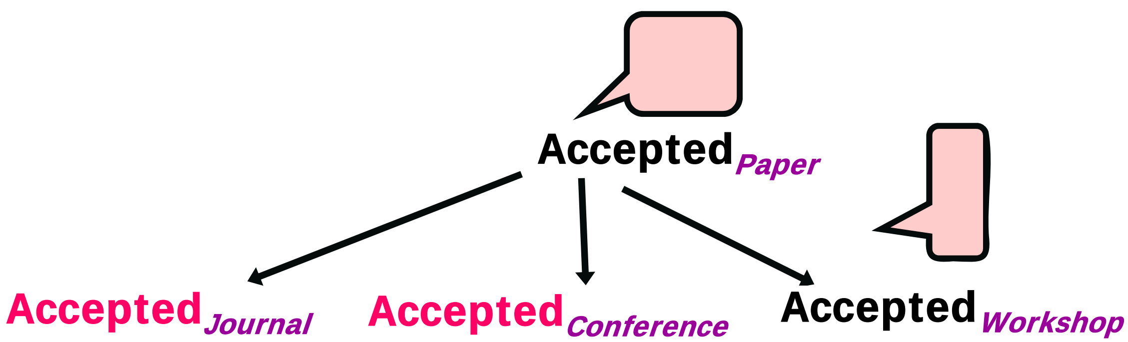
Guaranteeing Acyclicity w/ Subclasses



Learning PRM-CH

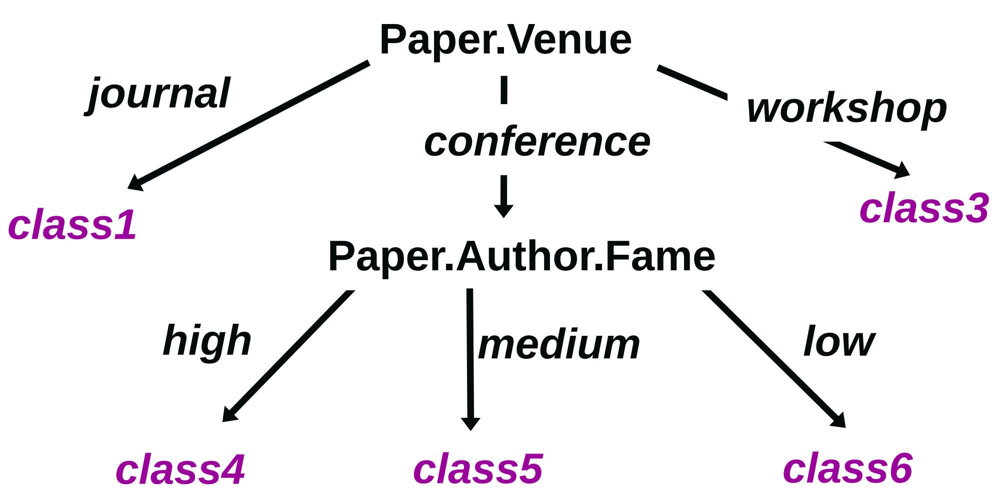
- Scenario 1: Class hierarchy is provided

- New Operators
  - Specialize/Inherit



Learning Class Hierarchy

- Issue: partially observable data set
- Construct decision tree for class defined over attributes *observed in training set*
- New operator
  - Split on class attribute
  - Related class attribute



PRMs w/ Class Hierarchies

Allow us to:

- Refine a “heterogenous” class into more coherent subclasses
- Refine probabilistic model along class hierarchy
  - Can specialize/inherit CPDs
  - Construct new dependencies that were originally “acyclic”

Provides bridge from class-based model to instance-based model

Summary: Directed Frame-based Approaches

- Focus on objects and relationships
  - what types of objects are there, and how are they related to each other?
  - how does a property of an object depend on other properties (of the same or other objects)?

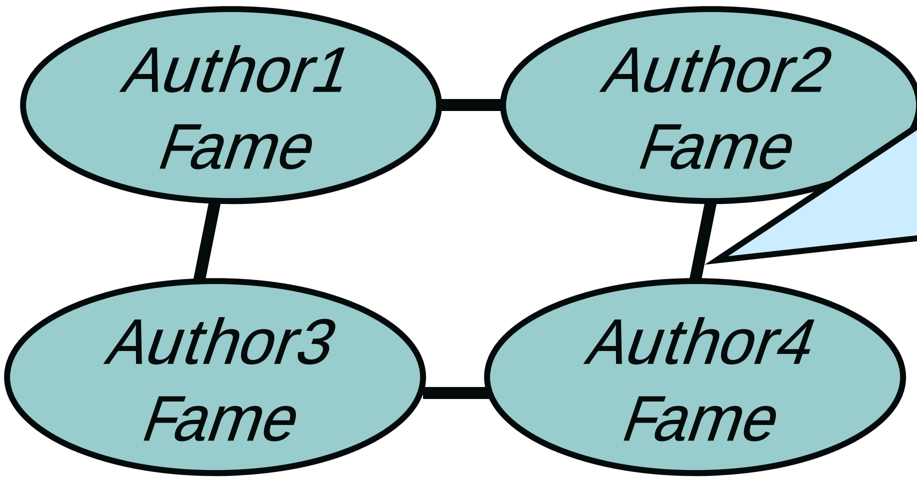
- Representation support
  - Attribute uncertainty
  - Structural uncertainty
  - Class Hierarchies

- Efficient Inference and Learning Algorithms

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - Frame-based Directed Models
- **Undirected Approaches**
  - **Markov Network Tutorial**
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Markov Networks



clique potential

F2	F4	ϕ(F2,F4)
f2	f4	0.6
f2	$\overline{f4}$	0.3
$\overline{f2}$	f4	0.3
$\overline{f2}$	$\overline{f4}$	1.5

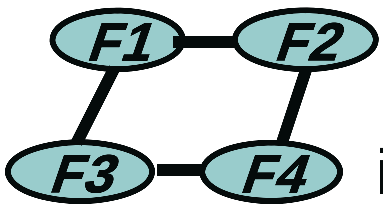
nodes = domain variables
edges = mutual influence

parameters measure compatibility of values

Network structure encodes conditional independencies:

I(*A1 Fame, A4 Fame* | *A2 Fame, A3 Fame*)

Markov Network Semantics



conditional independencies in MN structure + local clique potentials = full joint distribution over domain

$$P(f1,\bar{f}2,f3,\bar{f}4) = \frac{1}{Z}\phi_{12}(f1,\bar{f}2)\phi_{13}(f1,f3)\phi_{24}(\bar{f}2,\bar{f}4)\phi_{34}(f3,\bar{f}4)$$

where Z is a normalizing factor that ensures that the probabilities sum to 1

Good news: no acyclicity constraints

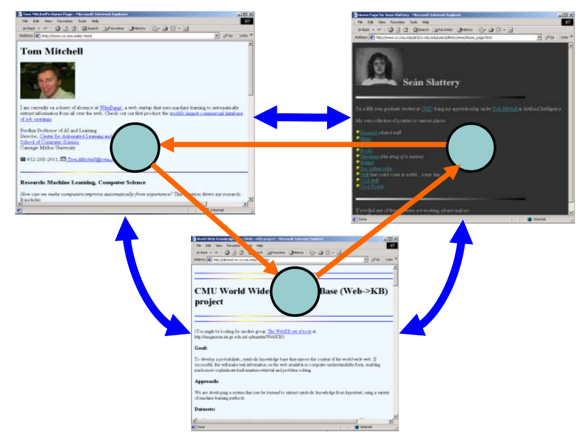
Bad news: global normalization (1/Z)

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - Frame-based Directed Models
- **Undirected Approaches**
  - Markov Network Tutorial
  - **Frame-based Undirected Models**
  - Rule-based Undirected Models

Advantages of Undirected Models

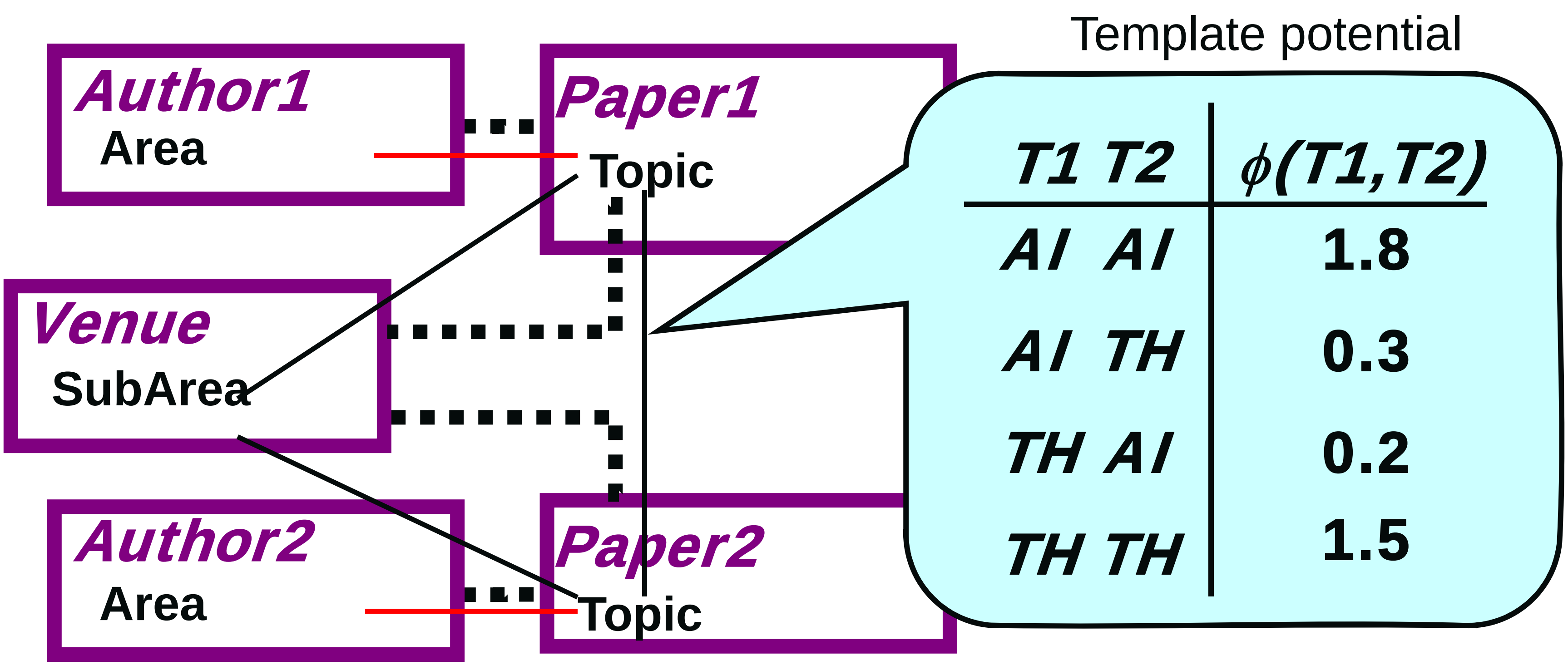
- Symmetric, non-causal interactions
  - Web: categories of linked pages are correlated
  - Social nets: individual correlated with peers
  - Cannot introduce direct edges because of cycles



- Patterns involving multiple entities
  - Web: “triangle” patterns
  - Social nets: transitive relations

Relational Markov Networks

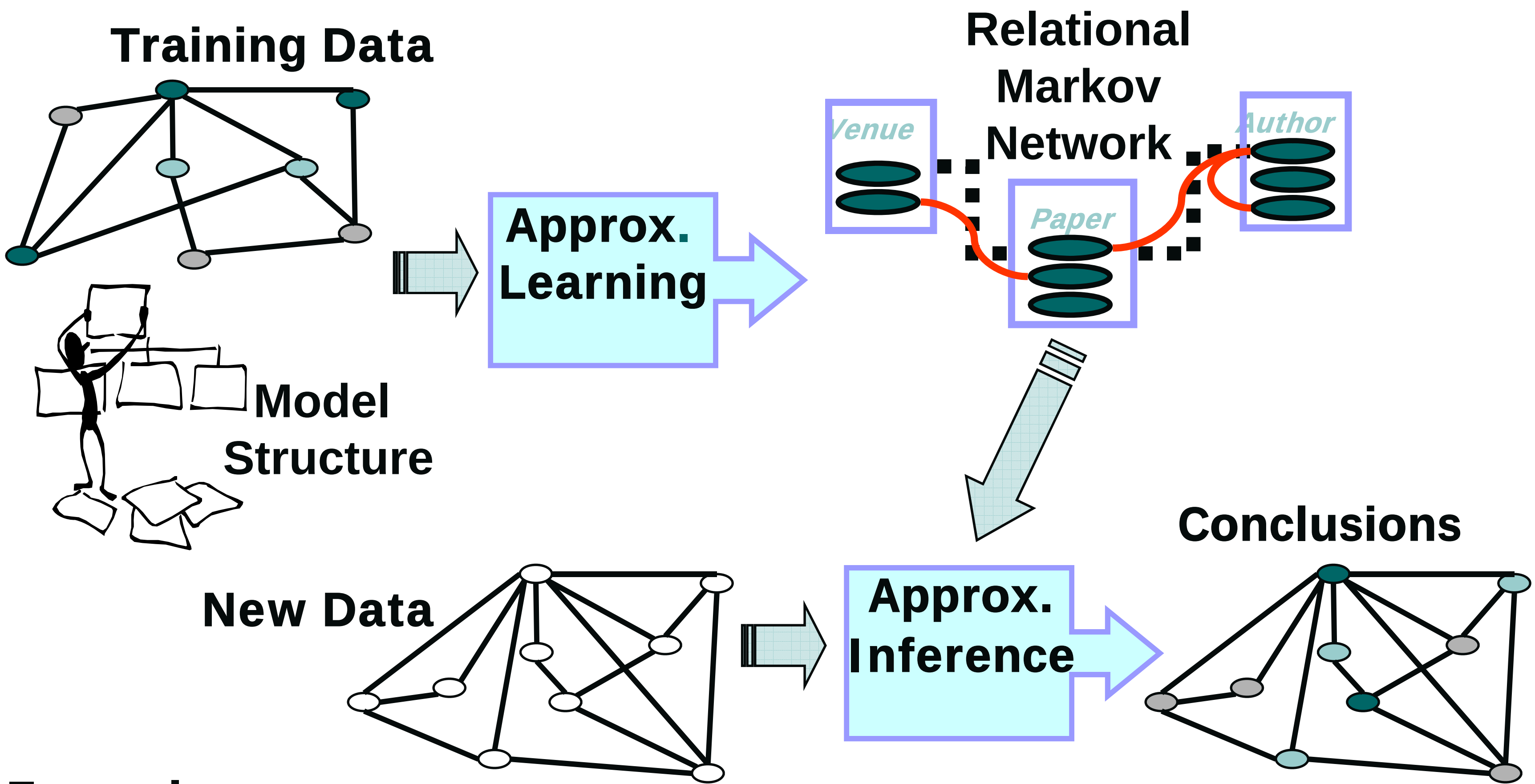
- Locality:
  - Local probabilistic dependencies given by relational links
- Universals:
  - Same dependencies hold for all objects linked in a particular pattern



RMNs

- Semantics
  - Instantiated RMN → MN
    - variables: attributes of all objects
    - dependencies: determined by links & RMN
- Learning
  - Discriminative training
  - Max-margin Markov networks
  - Associative Markov networks
- Collective classification:
  - Classifies multiple entities and links simultaneously
  - Exploits links & correlations between related entities

Collective Classification Overview



Example:

- Train on one labeled conference/venue
- Predict labels on a new conference given papers and links

Maximum Likelihood

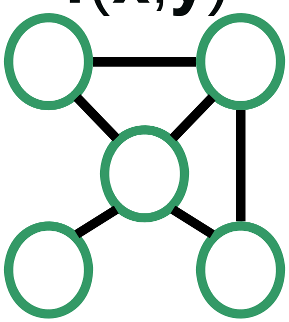
D

$o_1.\mathbf{x},\ o_1.\mathbf{y^*}$

...

$o_m.\mathbf{x},\ o_m.\mathbf{y^*}$

$\mathbf{f}(\mathbf{x},\mathbf{y})$



Estimation

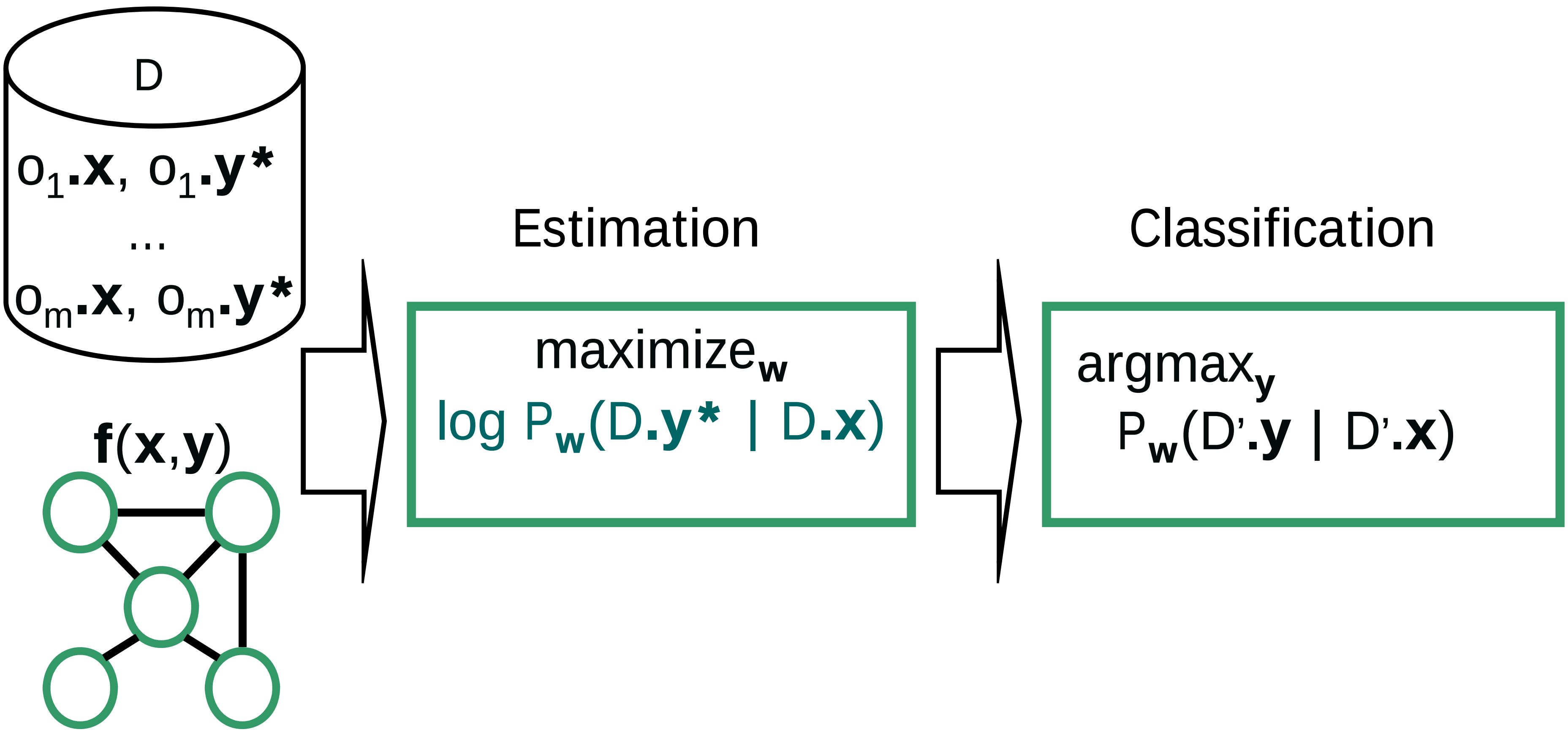
$\text{maximize}_{\mathbf{w}}$
$\log P_{\mathbf{w}}(D.\mathbf{y^*},\ D.\mathbf{x})$

Classification

$\text{argmax}_{\mathbf{y}}$
$P_{\mathbf{w}}(D'.\mathbf{y} \mid D'.\mathbf{x})$

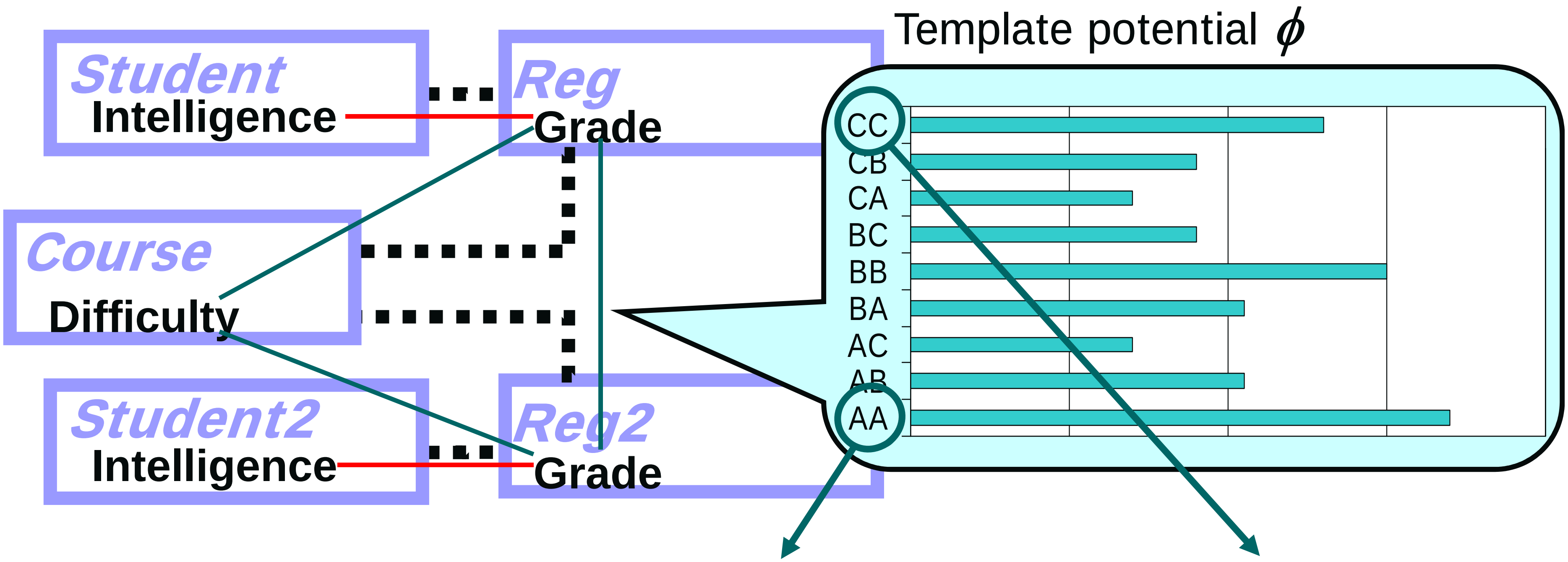
We don't care about the *joint* distribution P(D.X=x, D.Y=y*)

Maximum Conditional Likelihood



[**Lafferty et al '01**]

Learning RMNs



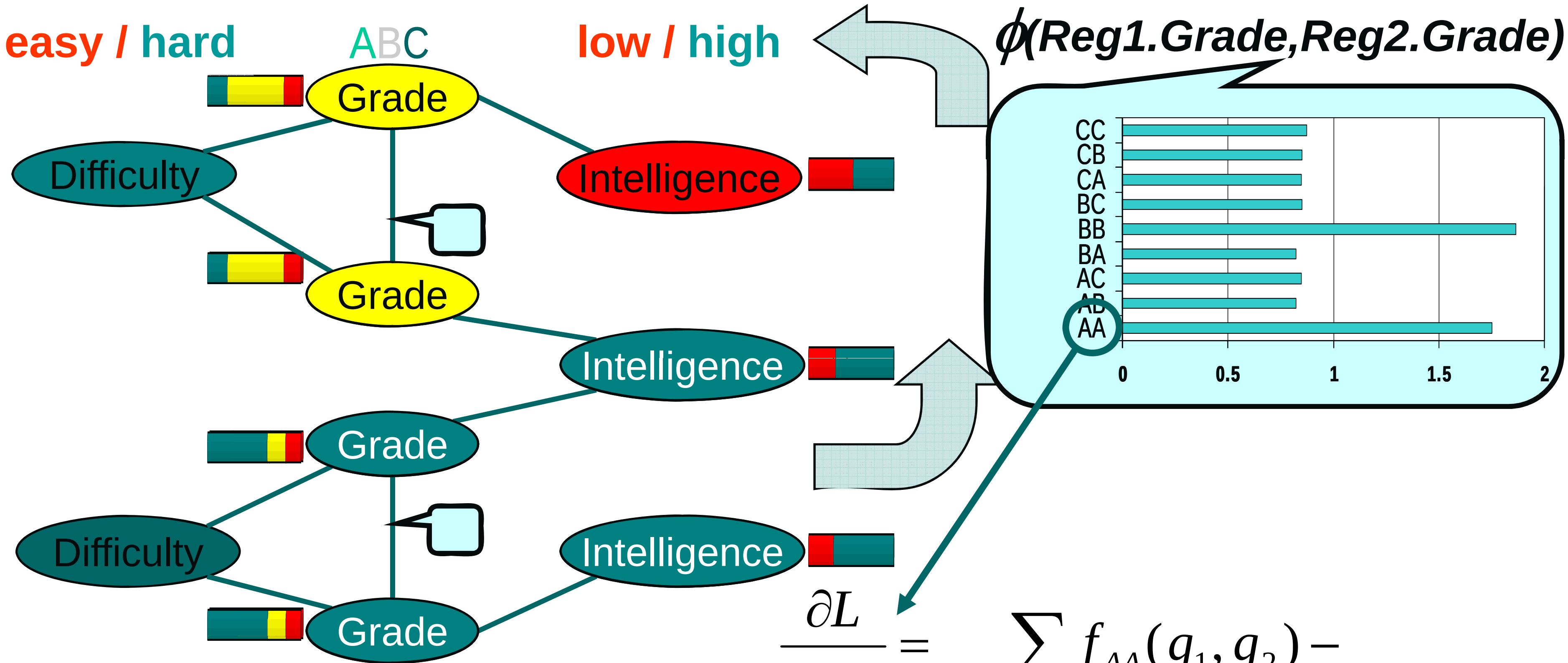
$$\phi(y_1, y_2) = \exp\{w_{AA} \cdot f_{AA}(y_1, y_2) + \ldots + w_{CC} \cdot f_{CC}(y_1, y_2)\}$$

$$\log P_{\mathbf{w}}(D.Y \mid D.X) = \mathbf{w} \cdot \mathbf{f}(D.Y, D.X) - \log Z(D.X)$$

$$\nabla_{\mathbf{w}} \log P_{\mathbf{w}}(D.Y \mid D.X) = \mathbf{f}(D.Y, D.X) - E_{P_{\mathbf{w}}(D.Y \mid D.X)} \mathbf{f}(D.Y, D.X)$$

Learning RMNs

Maximize L = log P(D.Grade,D.Intelligence|D.Difficulty)



$$\frac{\partial L}{\partial w_{AA}} = \sum_{g_1, g_2 \in D} f_{AA}(g_1, g_2) - \sum_{g_1, g_2 \in D} f_{AA}(g_1, g_2) P(g_1, g_2 \mid D.Difficulty)$$

Four SRL Approaches

- Directed Approaches
  - BN Tutorial
  - Rule-based Directed Models
  - Frame-based Directed Models
- **Undirected Approaches**
  - Markov Network Tutorial
  - Frame-based Undirected Models
  - **Rule-based Undirected Models**

Markov Logic Networks

- [Richardson and Domingos 03,Singla & Domingos 05, Kok & Domingos 05]

- 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

- Together with a finite set of constants,
  it 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

Example of an MLN

1.5	$\forall x\ author(x, p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x\ high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x, y\ co \quad author(x, y) \Rightarrow \left(smart(x) \Leftrightarrow smart(y)\right)$
∞	$\forall x, y\ \exists p\ author(x, p) \wedge author(y, p) \Rightarrow co\_author(x, y)$

Example of an MLN

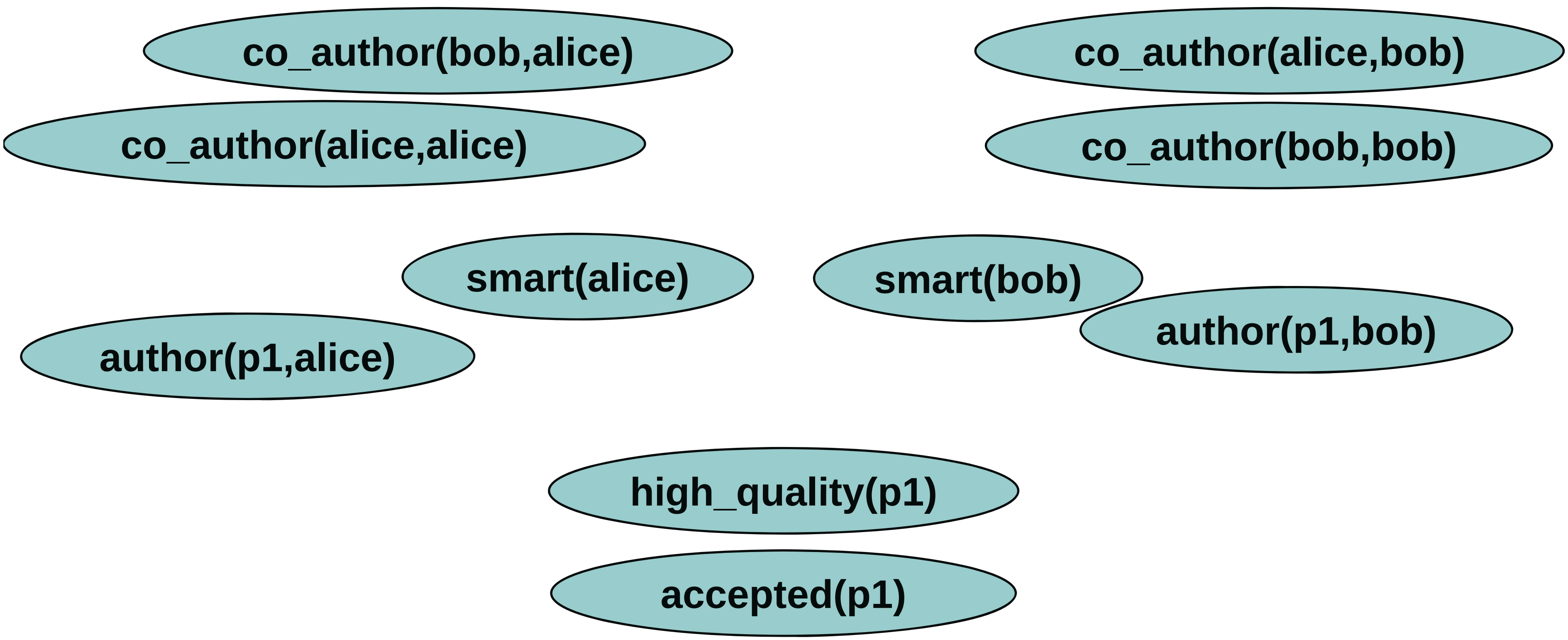
1.5	$\forall x \; author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x \; high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x,y \; co \quad author(x,y) \Rightarrow \left(smart(x) \Leftrightarrow smart(y)\right)$
∞	$\forall x,y \; \exists p \; author(x,p) \wedge author(y,p) \Rightarrow co\_author(x,y)$

Suppose we have constants: **alice**, **bob** and **p1**

Example of an MLN

1.5	$\forall x\ author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x\ high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x,y\ co\quad author(x,y) \Rightarrow \left(smart(x) \Leftrightarrow smart(y)\right)$
∞	$\forall x,y\ \exists p\ author(x,p) \wedge author(y,p) \Rightarrow co\_author(x,y)$

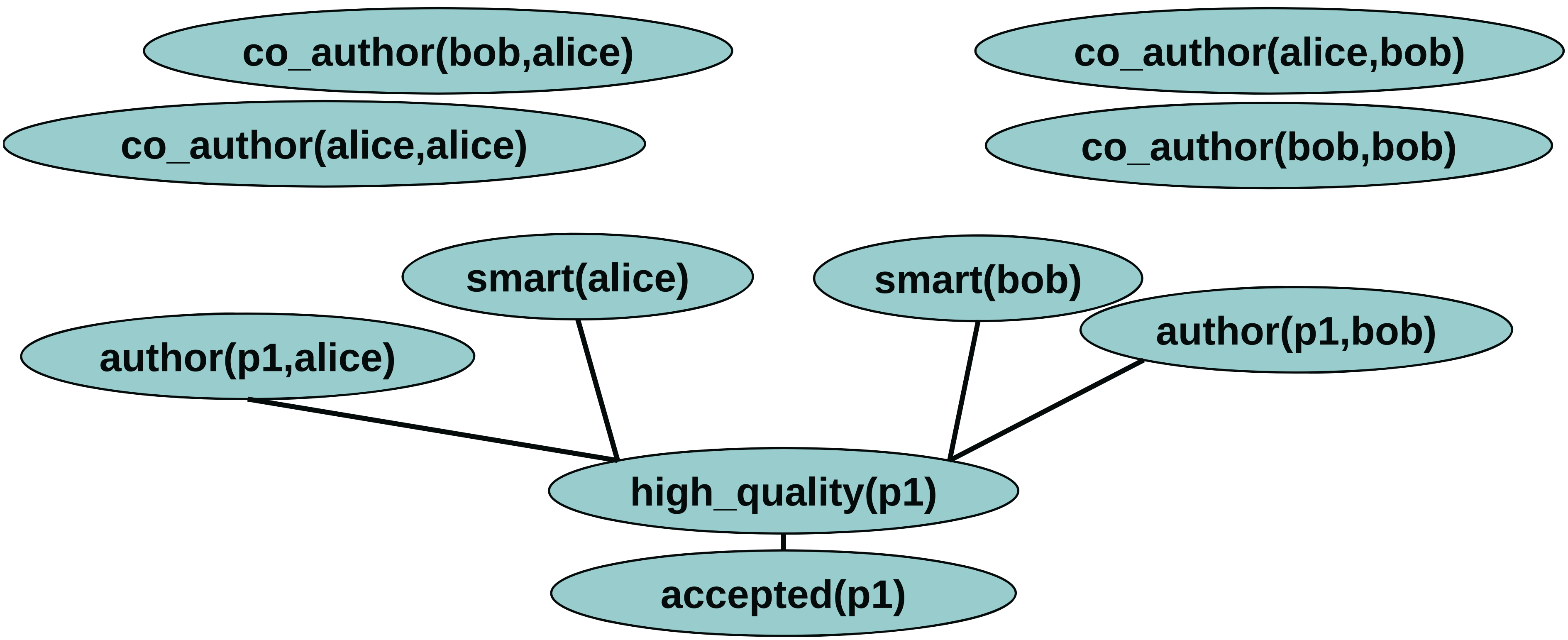
Suppose we have constants: **alice**, **bob** and **p1**



Example of an MLN

1.5	$\forall x\ author(x, p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x\ high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x, y\ co \quad author(x, y) \Rightarrow \left(smart(x) \Leftrightarrow smart(y)\right)$
∞	$\forall x, y\ \exists p\ author(x, p) \wedge author(y, p) \Rightarrow co\_author(x, y)$

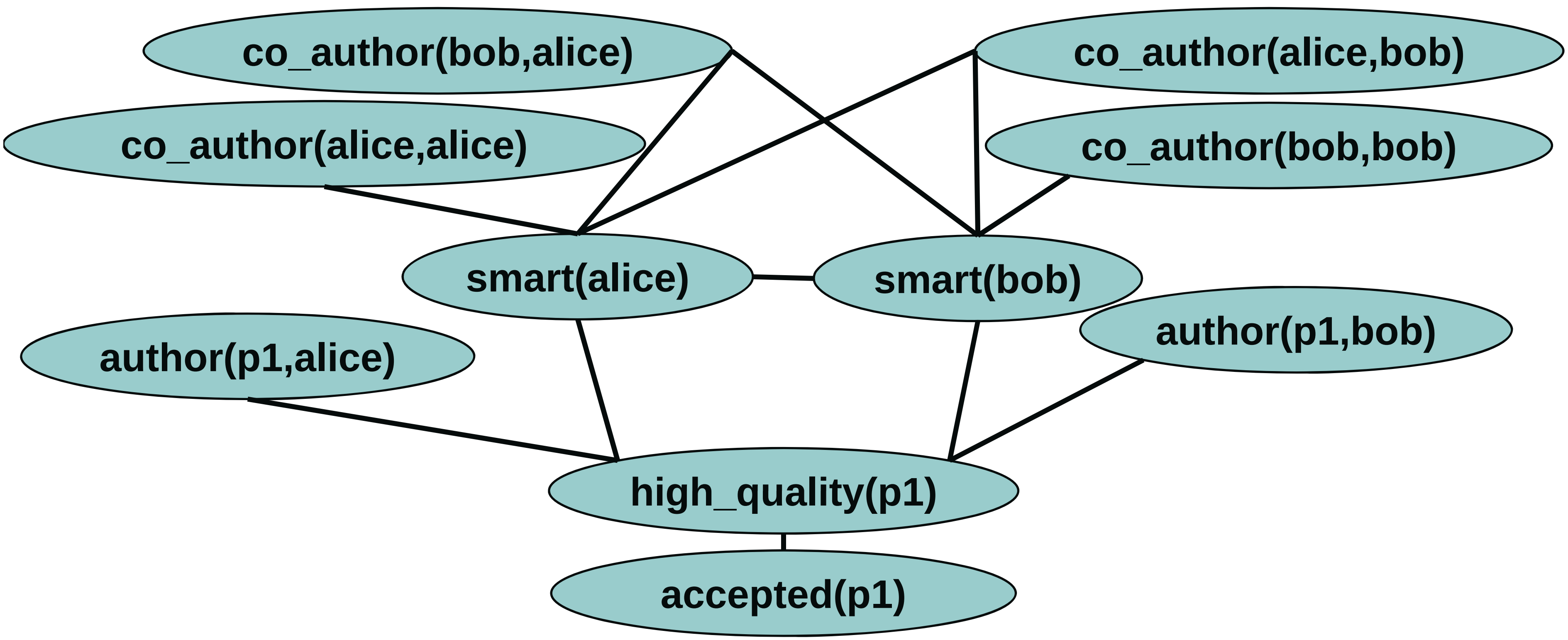
Suppose we have constants: **alice**, **bob** and **p1**



Example of an MLN

1.5	$\forall x\ author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x\ high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x, y\ co \quad author(x,y) \Rightarrow \left(smart(x) \Leftrightarrow smart(y)\right)$
∞	$\forall x, y\ \exists p\ author(x,p) \wedge author(y,p) \Rightarrow co\_author(x,y)$

Suppose we have constants: **alice**, **bob** and **p1**



Markov Logic Networks

- Combine first-order logic and Markov networks
  - **Syntax:** First-order logic + Weights
  - **Semantics:** Templates for Markov networks
- **Inference:** KBMC + MaxWalkSat + MCMC
- **Learning:** ILP + Pseudo-likelihood / discriminitive training

Summary: Undirected Approaches

- Focus on symmetric, non-causal relationships
  - Like directed approaches, support collective classification

Four SRL Approaches

- Directed Approaches
  - Rule-based Directed Models
  - Frame-based Directed Models
- **Undirected Approaches**
  - Frame-based Undirected Models
  - Rule-based Undirected Models

Themes: Representation

- Basic representational elements and focus
  - rules: facts
  - frames: objects
  - programs: processes
- Representing domain structure
  - context
  - relational skeleton
  - non-probabilistic language constructs
- Representing local probabilities
  - noise factors
  - conditional probability tables
  - clique potentials

Themes: Representational Issues

- Structural uncertainty
  - reference, exists, type, number, identity
- Combining probabilities from multiple sources
  - combining rules
  - aggregation
- Cyclicity
  - ruling out (stratification)
  - introducing time
  - guaranteed acyclic relations
  - undirected models
- Functions and infinite chains
  - iterative approximation

Themes: Inference

- Inference on ground random variables
  - knowledge based model construction
- Inference at the first-order object level
  - first-order variable elimination
    - unification
    - quantification over populations
  - structured variable elimination
  - memoization
- Utilizing entity-relations structure
- Query-directed inference
  - backward chaining on query and evidence
  - lazy evaluation

Themes: Learning

- Learning parameters
  - Parameter sharing
    - rules apply many times
    - same type of object appears many times
    - same function is called many times
  - Expectation-Maximization
- Learning structure
  - structure search
  - legal models
  - scoring function
  - search operators

Goals

- By the end of this tutorial, hopefully, you will be:
  1. able to distinguish among different SRL tasks
  2. able to represent a problem in one of several SRL representations
  3. excited about SRL research problems and practical applications

- Many other interesting topics that I didn't have time to cover…

Conclusion

- Statistical Relational Learning
  - Supports multi-relational, heterogeneous domains
  - Supports noisy, uncertain, non-IID data
  - aka, real-world data!
- Differences in approaches:
  - rule-based vs. frame-based
  - directed vs. undirected
- Many common issues:
  - Need for collective classification and consolidation
  - Need for aggregation and combining rules
  - Need to handle labeled and unlabeled data
  - Need to handle structural uncertainty
  - etc.
- Great opportunity for combining rich logical representation and inference and learning with hierarchical statistical models!!