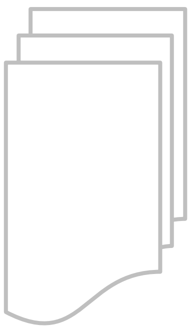
# Knowledge Graph Construction from Text

AAAI 2017

Jay Pujara, Sameer Singh, Bhavana Dalvi

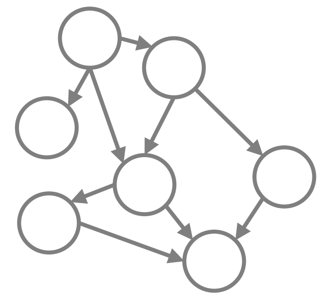
# Tutorial Overview

**Part 1: Knowledge Graphs**



**Part 2: Knowledge Extraction**

**Part 3: Graph Construction**



**Part 4: Critical Analysis**

# Tutorial Outline

1. Knowledge Graph Primer [Jay] 
2. Knowledge Extraction from Text
   a. NLP Fundamentals [Sameer] 
   b. Information Extraction [Bhavana] 

Coffee Break 

3. **Knowledge Graph Construction**
   a. **Probabilistic Models [Jay]** 
   b. **Embedding Techniques [Sameer]** 
4. Critical Overview and Conclusion [Bhavana] 

# Knowledge Graph Construction

TOPICS:

PROBLEM SETTING

PROBABILISTIC MODELS

EMBEDDING TECHNIQUES

# Knowledge Graph Construction

TOPICS:

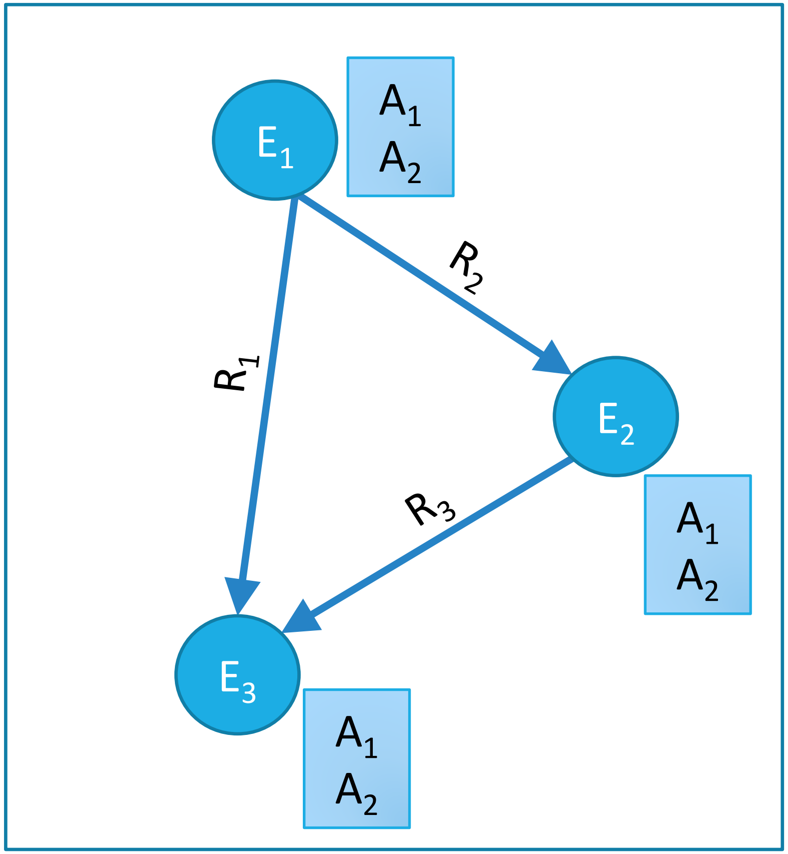
Problem Setting

Probabilistic Models

Embedding Techniques

# Reminder: Basic problems

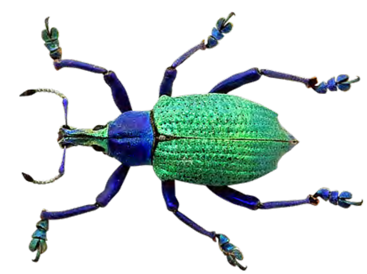
- **Who** are the entities (nodes) in the graph?
- **What** are their attributes and types (labels)?
- **How** are they related (edges)?



# Graph Construction Issues

Extracted knowledge is:

- ambiguous:
  - Ex: Beetles, beetles, Beatles
  - Ex: citizenOf, livedIn, bornIn

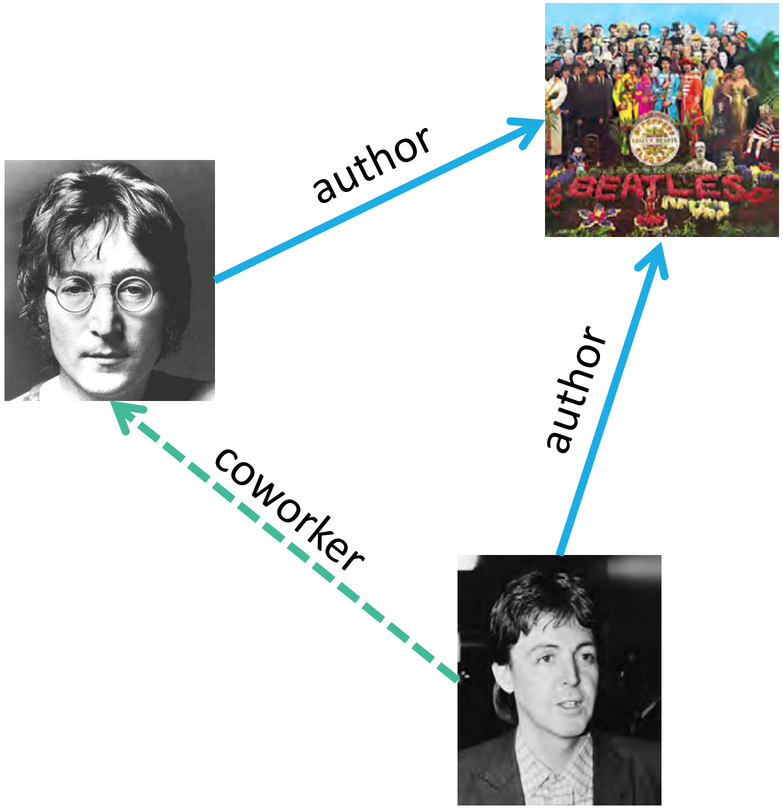


# Graph Construction Issues

Extracted knowledge is:

- ambiguous

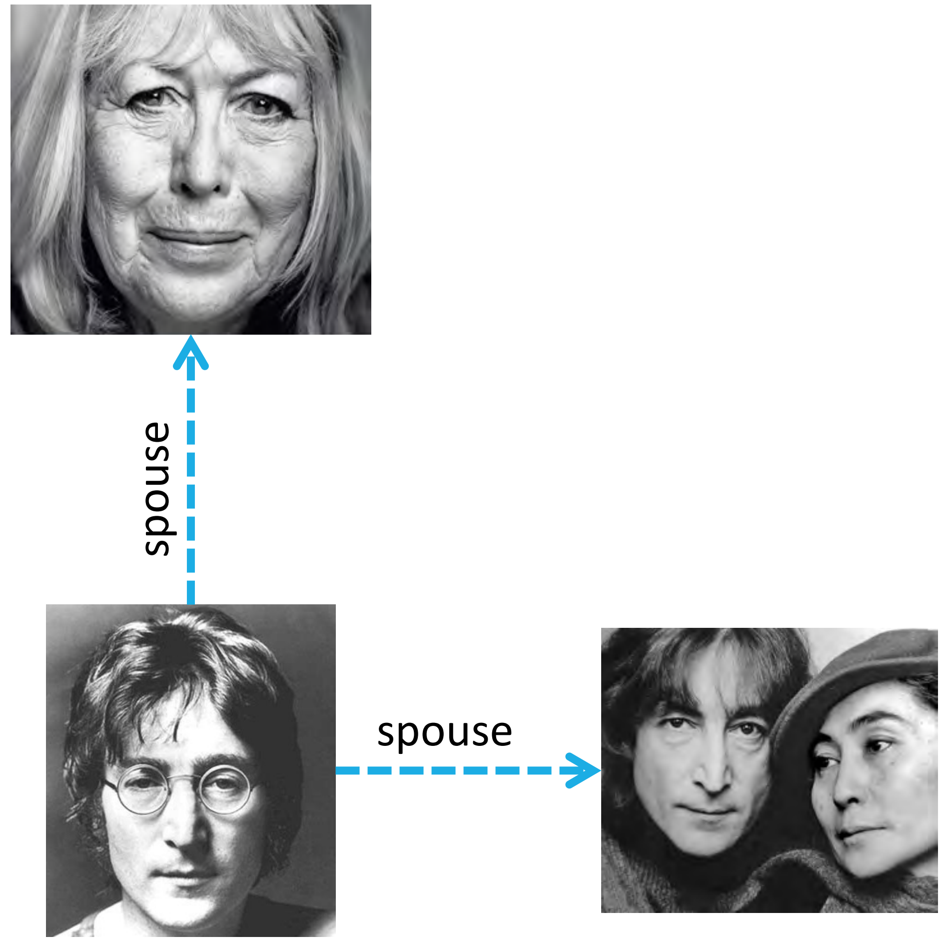
- incomplete
  - Ex: missing relationships
  - Ex: missing labels
  - Ex: missing entities



# Graph Construction Issues

Extracted knowledge is:

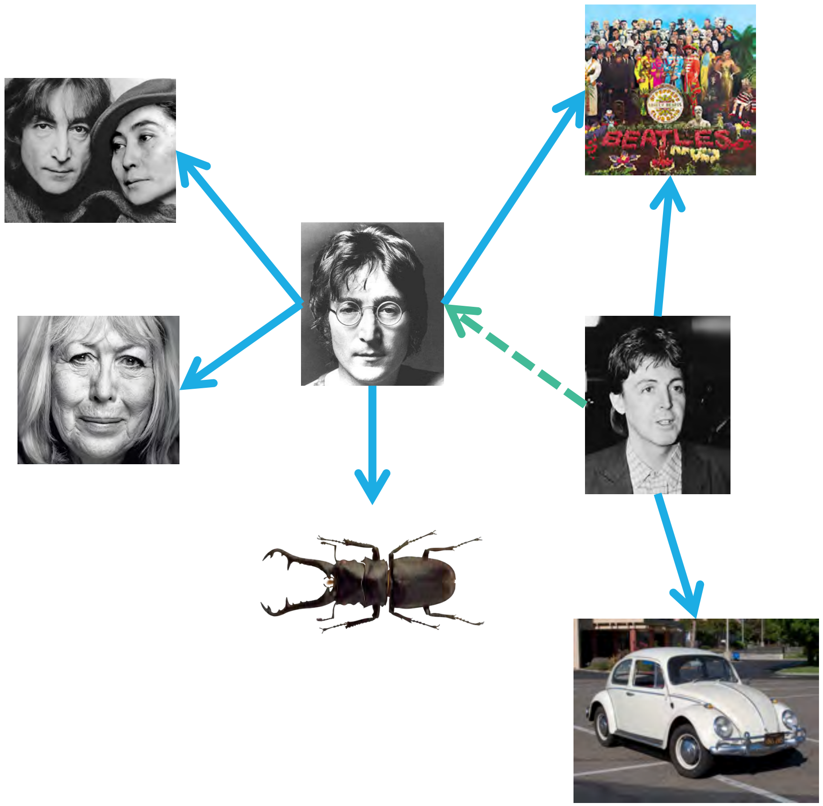
- ambiguous
- incomplete
- inconsistent
  - Ex: Cynthia Lennon, Yoko Ono
  - Ex: exclusive labels (alive, dead)
  - Ex: domain-range constraints



# Graph Construction Issues

Extracted knowledge is:

- ambiguous
- incomplete
- inconsistent



# Graph Construction approach

- Graph construction **cleans** and **completes** extraction graph
- Incorporate ontological constraints and relational patterns
- Discover statistical relationships within knowledge graph

# Knowledge Graph Construction

TOPICS:

PROBLEM SETTING

PROBABILISTIC MODELS

EMBEDDING TECHNIQUES

# Graph Construction
## Probabilistic Models

TOPICS:

- Overview
- Graphical models
- Random Walk Methods

# Graph Construction
## Probabilistic Models

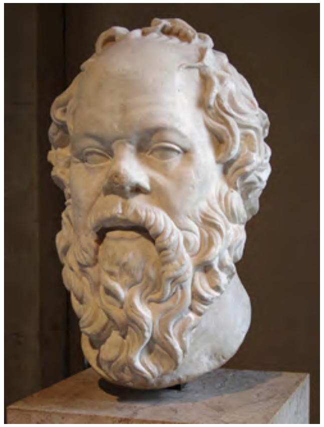
TOPICS:

<u>OVERVIEW</u>

GRAPHICAL MODELS

RANDOM WALK METHODS

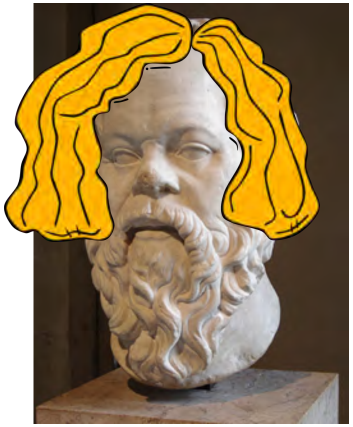
# Beyond Pure Reasoning



- Classical AI approach to knowledge: reasoning

Lbl(Socrates, Man) & Sub(Man, Mortal) -> Lbl(Socrates, Mortal)

# Beyond Pure Reasoning

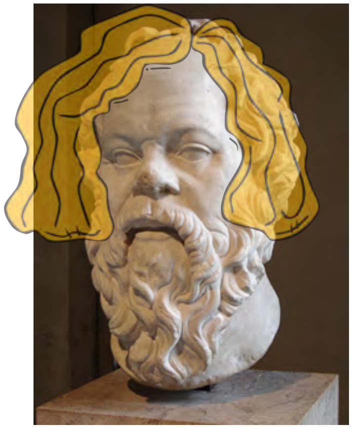


- Classical AI approach to knowledge: reasoning

Lbl(Socrates, Man) & Sub(Man, Mortal) -> Lbl(Socrates, Mortal)

- Reasoning difficult when extracted knowledge has errors

# Beyond Pure Reasoning



- Classical AI approach to knowledge: reasoning

Lbl(Socrates, Man) & Sub(Man, Mortal) -> Lbl(Socrates, Mortal)

- Reasoning difficult when extracted knowledge has errors
- Solution: probabilistic models

P(Lbl(Socrates, Mortal)|Lbl(Socrates,Man)=0.9)

# Graph Construction
## Probabilistic Models

TOPICS:

- Overview
- Graphical models
- Random Walk Methods

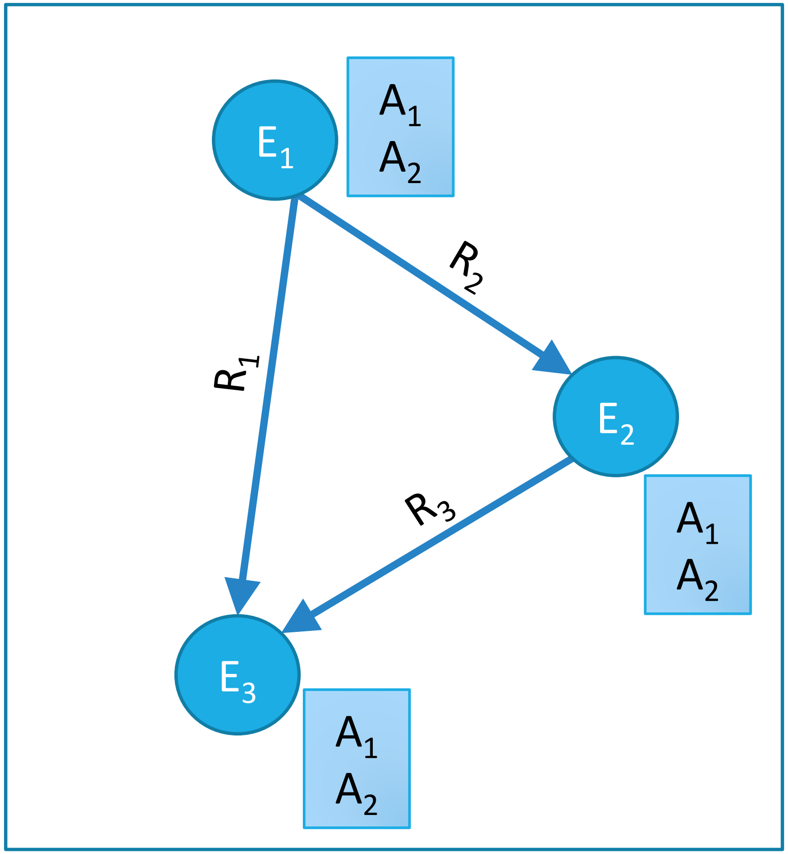
# Graphical Models: Overview

- Define **joint probability distribution** on knowledge graphs
- Each candidate fact in the knowledge graph is a **variable**
- Statistical signals, ontological knowledge and rules parameterize the **dependencies** between variables
- Find most likely knowledge graph by **optimization/sampling**

# Knowledge Graph Identification

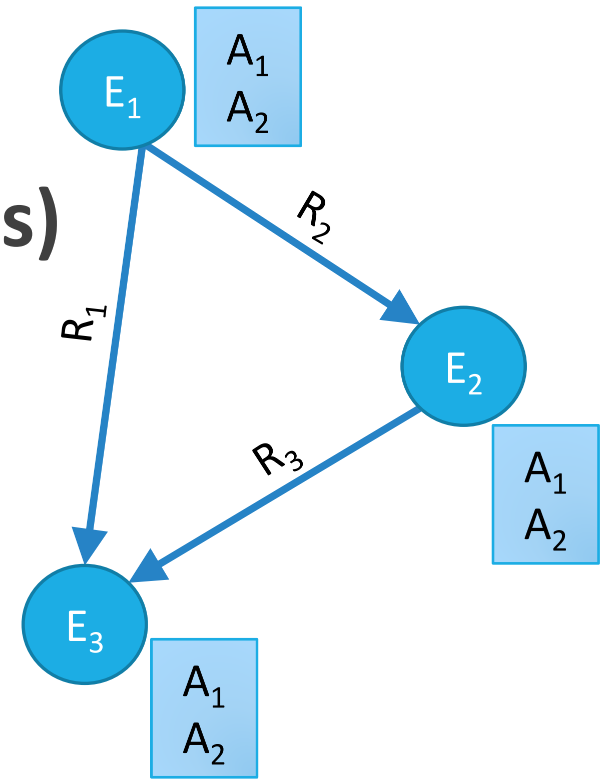
**Define a graphical model to perform all three of these tasks simultaneously!**

- **Who** are the entities (nodes) in the graph?
- **What** are their attributes and types (labels)?
- **How** are they related (edges)?



# Knowledge Graph Identification

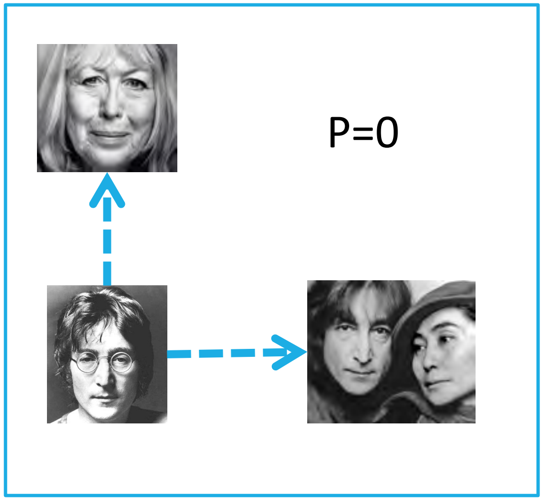
**P(Who, What, How | Extractions)**

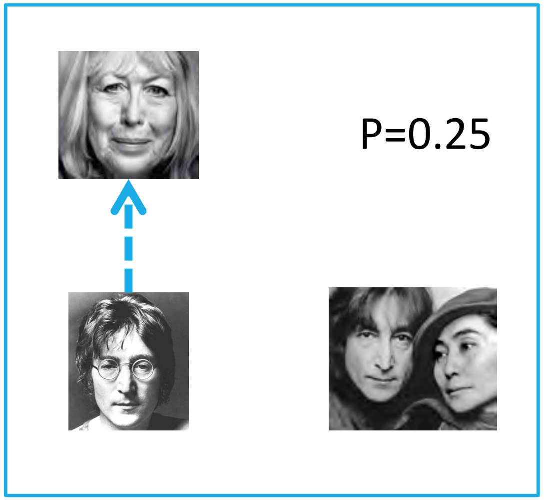


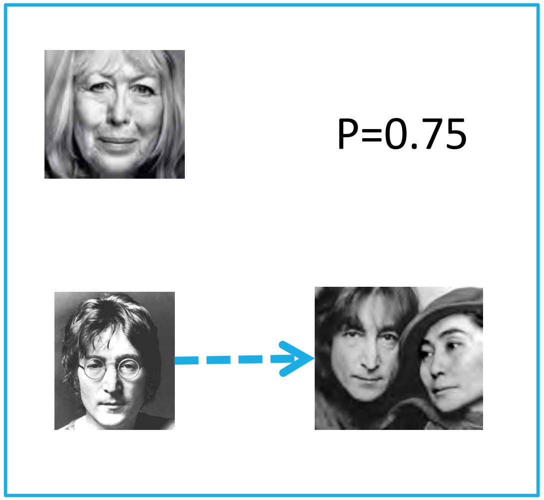
# Probabilistic Models

- Use dependencies between facts in KG

- Probability defined *jointly* over facts



# What determines probability?

- **Statistical signals from text extractors and classifiers**

# What determines probability?

- **Statistical signals from text extractors and classifiers**
  - P(R(John,Spouse,Yoko))=0.75; P(R(John,Spouse,Cynthia))=0.25
  - LevenshteinSimilarity(Beatles, Beetles) = 0.9

# What determines probability?

- Statistical signals from text extractors and classifiers
- **Ontological knowledge about domain**

# What determines probability?

- **Statistical signals from text extractors and classifiers**

- **Ontological knowledge about domain**
  - Functional(Spouse) & R(A,Spouse,B) -> !R(A,Spouse,C)
  - Range(Spouse, Person) & R(A,Spouse,B) -> Type(B, Person)

# What determines probability?

- **Statistical signals from text extractors and classifiers**

- **Ontological knowledge about domain**

- **Rules and patterns mined from data**

# What determines probability?

- **Statistical signals from text extractors and classifiers**

- **Ontological knowledge about domain**

- **Rules and patterns mined from data**
  - R(A, Spouse, B) & R(A, Lives, L) -> R(B, Lives, L)
  - R(A, Spouse, B) & R(A, Child, C) -> R(B, Child, C)

# What determines probability?

- **Statistical signals from text extractors and classifiers**
  - P(R(John,Spouse,Yoko))=0.75; P(R(John,Spouse,Cynthia))=0.25
  - LevenshteinSimilarity(Beatles, Beetles) = 0.9

- **Ontological knowledge about domain**
  - Functional(Spouse) & R(A,Spouse,B) -> !R(A,Spouse,C)
  - Range(Spouse, Person) & R(A,Spouse,B) -> Type(B, Person)

- **Rules and patterns mined from data**
  - R(A, Spouse, B) & R(A, Lives, L) -> R(B, Lives, L)
  - R(A, Spouse, B) & R(A, Child, C) -> R(B, Child, C)

# Example: The Fab Four

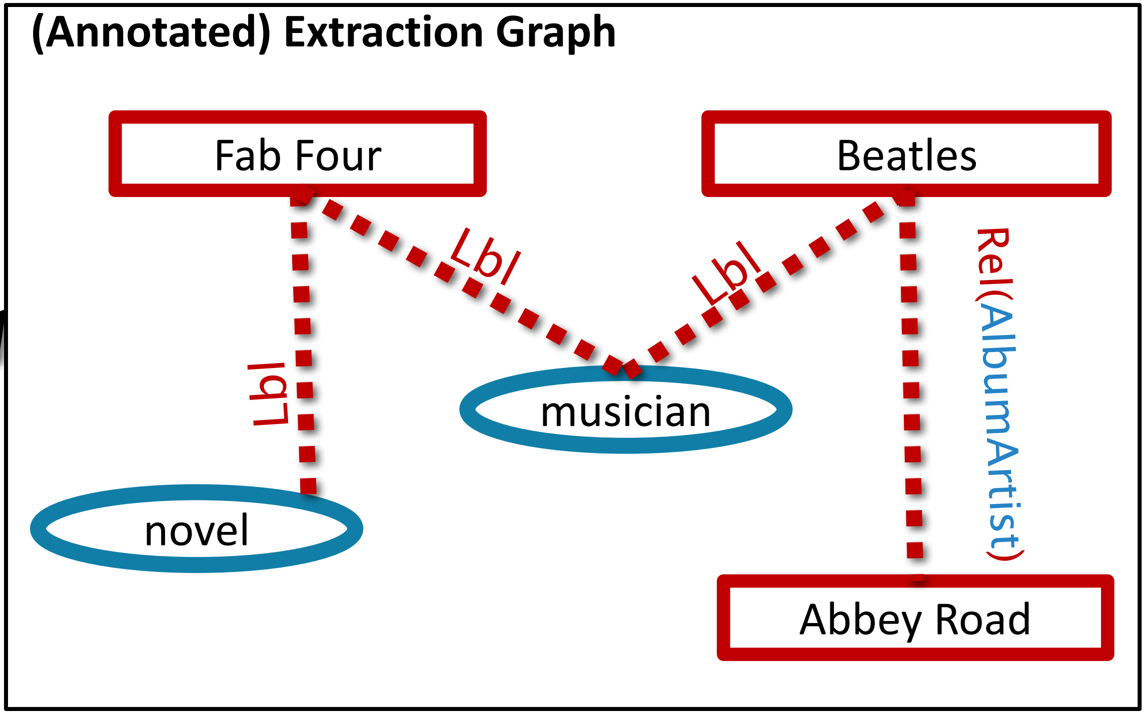
THE BEATLES

# Illustration of KG Identification

**Uncertain Extractions:**
.5: Lbl(Fab Four, novel)
.7: Lbl(Fab Four, musician)
.9: Lbl(Beatles, musician)
.8: Rel(Beatles,AlbumArtist,
Abbey Road)

# Illustration of KG Identification

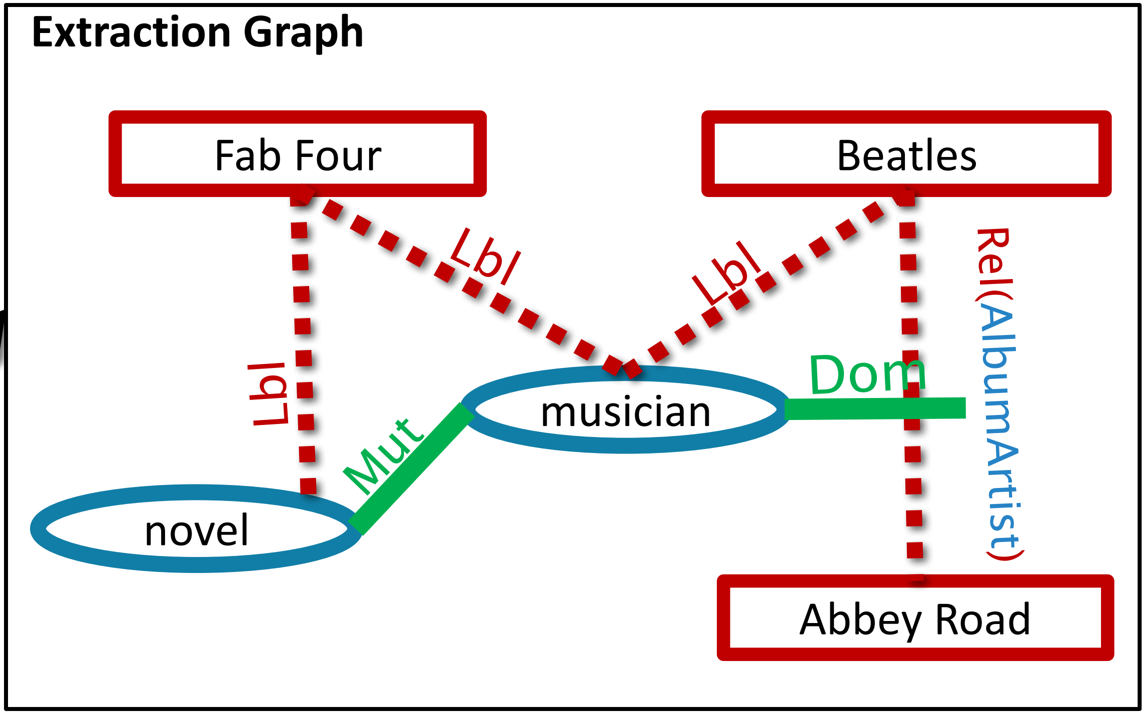
**Uncertain Extractions:**
.5: Lbl(Fab Four, novel)
.7: Lbl(Fab Four, musician)
.9: Lbl(Beatles, musician)
.8: Rel(Beatles,AlbumArtist,
Abbey Road)



# Illustration of KG Identification

**Uncertain Extractions:**
.5: Lbl(Fab Four, novel)
.7: Lbl(Fab Four, musician)
.9: Lbl(Beatles, musician)
.8: Rel(Beatles,AlbumArtist, Abbey Road)

**Ontology:**
Dom(albumArtist, musician)
Mut(novel, musician)

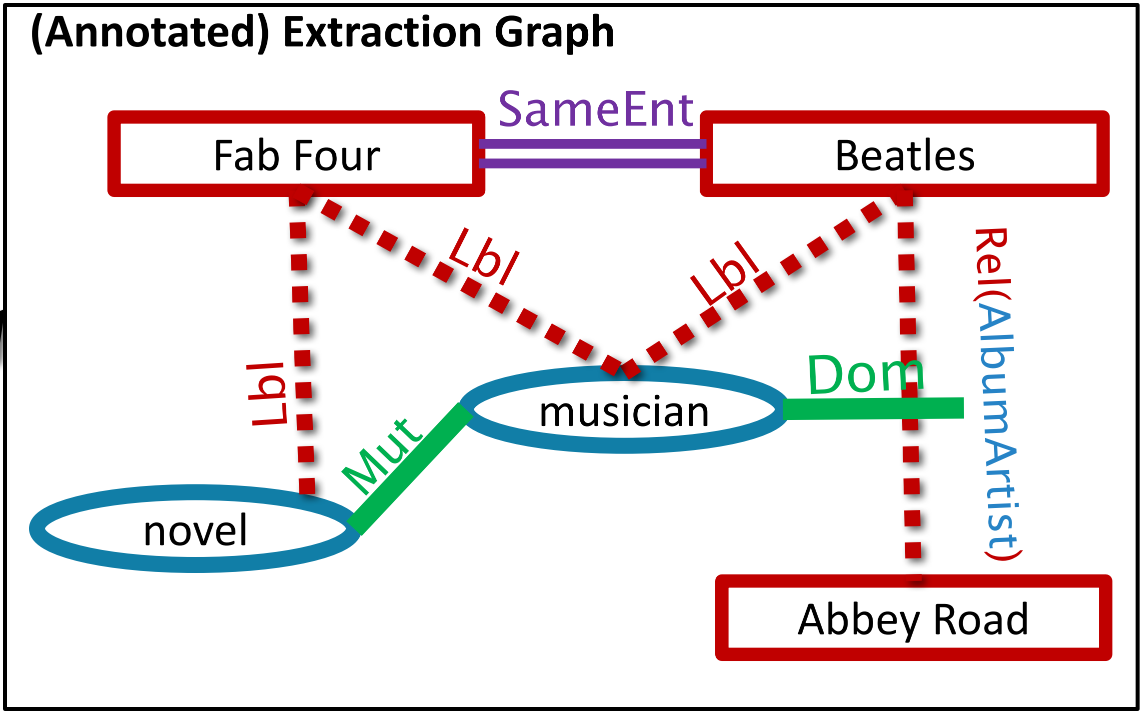


# Illustration of KG Identification

**Uncertain Extractions:**
.5: Lbl(Fab Four, novel)
.7: Lbl(Fab Four, musician)
.9: Lbl(Beatles, musician)
.8: Rel(Beatles,AlbumArtist, Abbey Road)

**Ontology:**
Dom(albumArtist, musician)
Mut(novel, musician)

**Entity Resolution:**
SameEnt(Fab Four, Beatles)



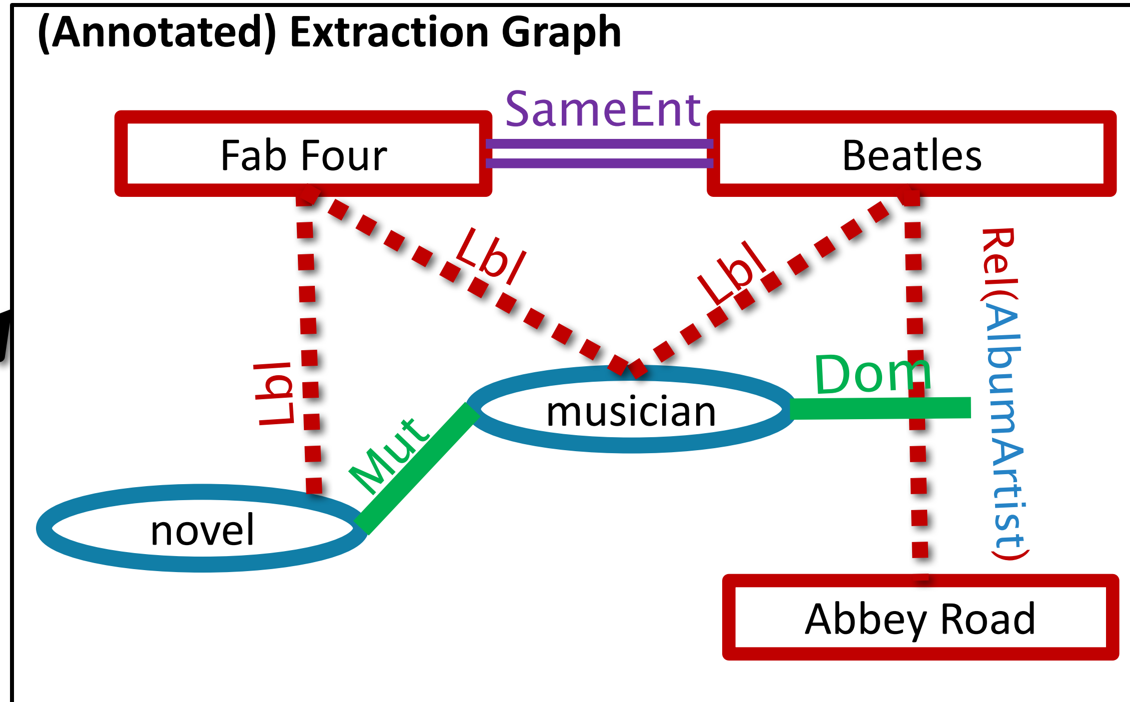
# Illustration of KG Identification

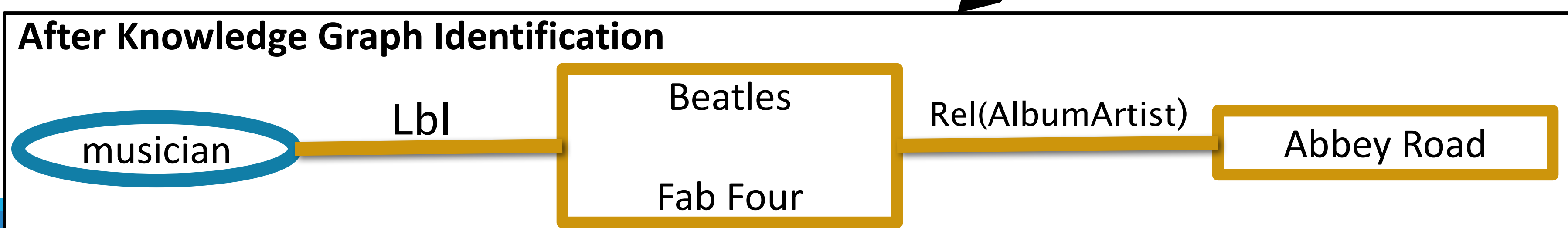
**Uncertain Extractions:**
.5: Lbl(Fab Four, novel)
.7: Lbl(Fab Four, musician)
.9: Lbl(Beatles, musician)

.8: Rel(Beatles,AlbumArtist,
Abbey Road)

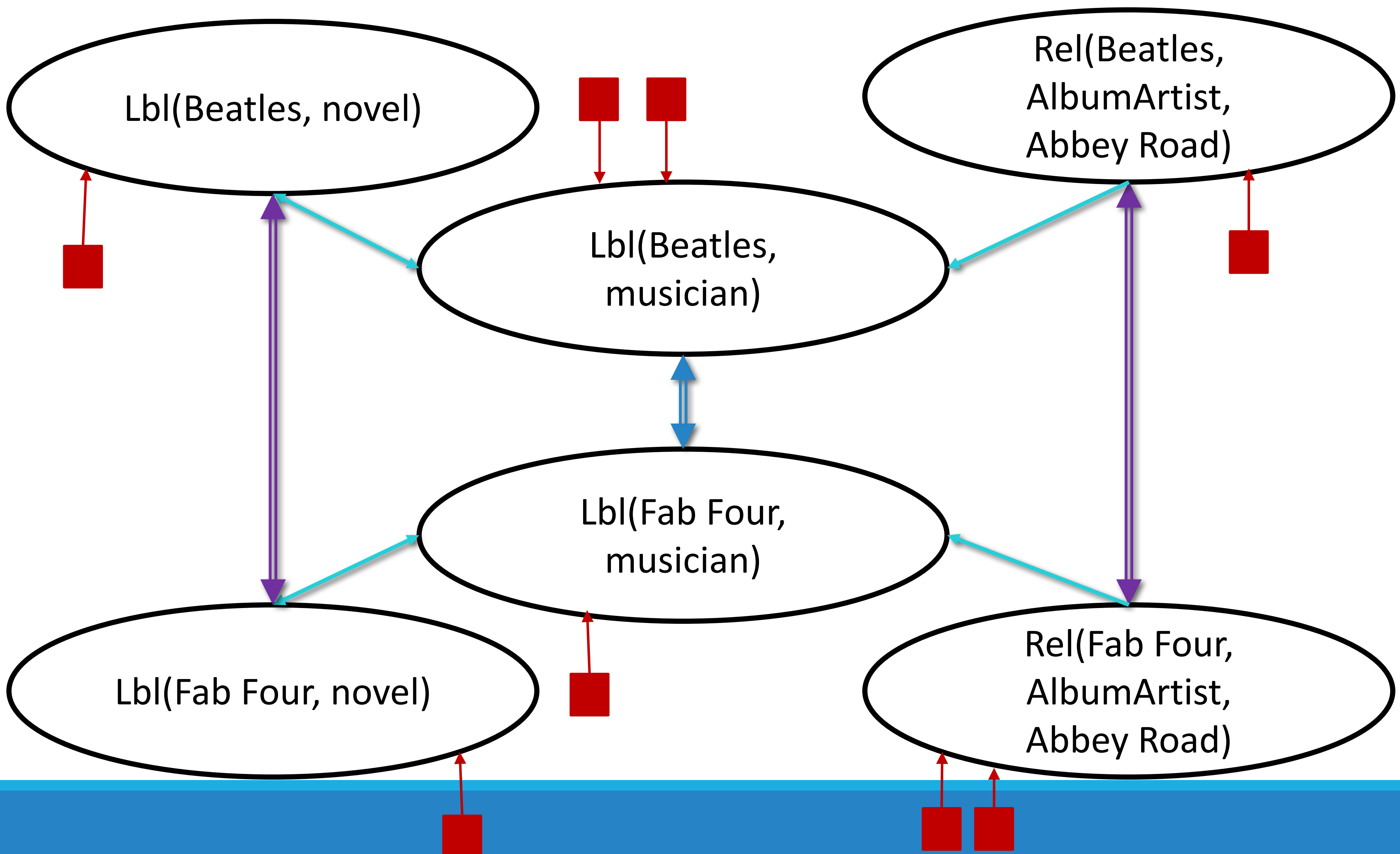
**Ontology:**
Dom(albumArtist, musician)
Mut(novel, musician)

**Entity Resolution:**
SameEnt(Fab Four, Beatles)



# Probabilistic graphical model for KG



# Defining graphical models

- Many options for defining a graphical model
- We focus on two approaches, MLNs and PSL, that use **rules**
- **MLNs** treat facts as Boolean, use sampling for satisfaction
- **PSL** infers a “truth value” for each fact via optimization

# Rules for KG Model

```
100:   Subsumes(L1,L2)    & Label(E,L1)        ->  Label(E,L2)
100:   Exclusive(L1,L2)   & Label(E,L1)        -> !Label(E,L2)

100:   Inverse(R1,R2)     & Relation(R1,E,O)   ->  Relation(R2,O,E)
100:   Subsumes(R1,R2)    & Relation(R1,E,O)   ->  Relation(R2,E,O)
100:   Exclusive(R1,R2)   & Relation(R1,E,O)   -> !Relation(R2,E,O)

100:   Domain(R,L)        & Relation(R,E,O)    ->  Label(E,L)
100:   Range(R,L)         & Relation(R,E,O)    ->  Label(O,L)

10:    SameEntity(E1,E2)  & Label(E1,L)        ->  Label(E2,L)
10:    SameEntity(E1,E2)  & Relation(R,E1,O)   ->  Relation(R,E2,O)

1:     Label_NYT(E,L)                          ->  Label(E,L)
1:     Label_YouTube(E,L)                      ->  Label(E,L)
1:     Relation_LATimes(R,E,O)                 ->  Relation(R,E,O)
1:                                                !Relation(R,E,O)
1:                                                !Label(E,L)
```

# Rules to Distributions

- Rules are *grounded* by substituting literals into formulas

$$\mathbf{w_r} : \textsc{SameEnt}(\text{Fab Four}, \text{Beatles}) \;\wedge$$
$$\textsc{Lbl}(\text{Beatles}, \text{musician}) \Rightarrow \textsc{Lbl}(\text{Fab Four}, \text{musician})$$

- Each ground rule has a weighted *satisfaction* derived from the formula's truth value

$$\boxed{P(G|E)} = \frac{1}{Z}\exp\left[\sum_{r \in R} w_r \phi_r(G, E)\right]$$

- Together, the ground rules provide a joint probability distribution over knowledge graph facts, conditioned on the extractions

# Probability Distribution over KGs

$$P(G \mid E) = \frac{1}{Z} \exp\left[ -\sum_{r \in R} w_r \, \varphi_r(G) \right]$$

$$\textsc{CandLbl}_T(\texttt{FabFour}, \texttt{novel}) \Rightarrow \textsc{Lbl}(\texttt{FabFour}, \texttt{novel})$$

$$\textsc{Mut}(\texttt{novel}, \texttt{musician}) \wedge \textsc{Lbl}(\texttt{Beatles}, \texttt{novel}) \Rightarrow \neg\textsc{Lbl}(\texttt{Beatles}, \texttt{musician})$$

$$\textsc{SameEnt}(\texttt{Beatles}, \texttt{FabFour}) \wedge \textsc{Lbl}(\texttt{Beatles}, \texttt{musician}) \Rightarrow \textsc{Lbl}(\texttt{FabFour}, \texttt{musician})$$

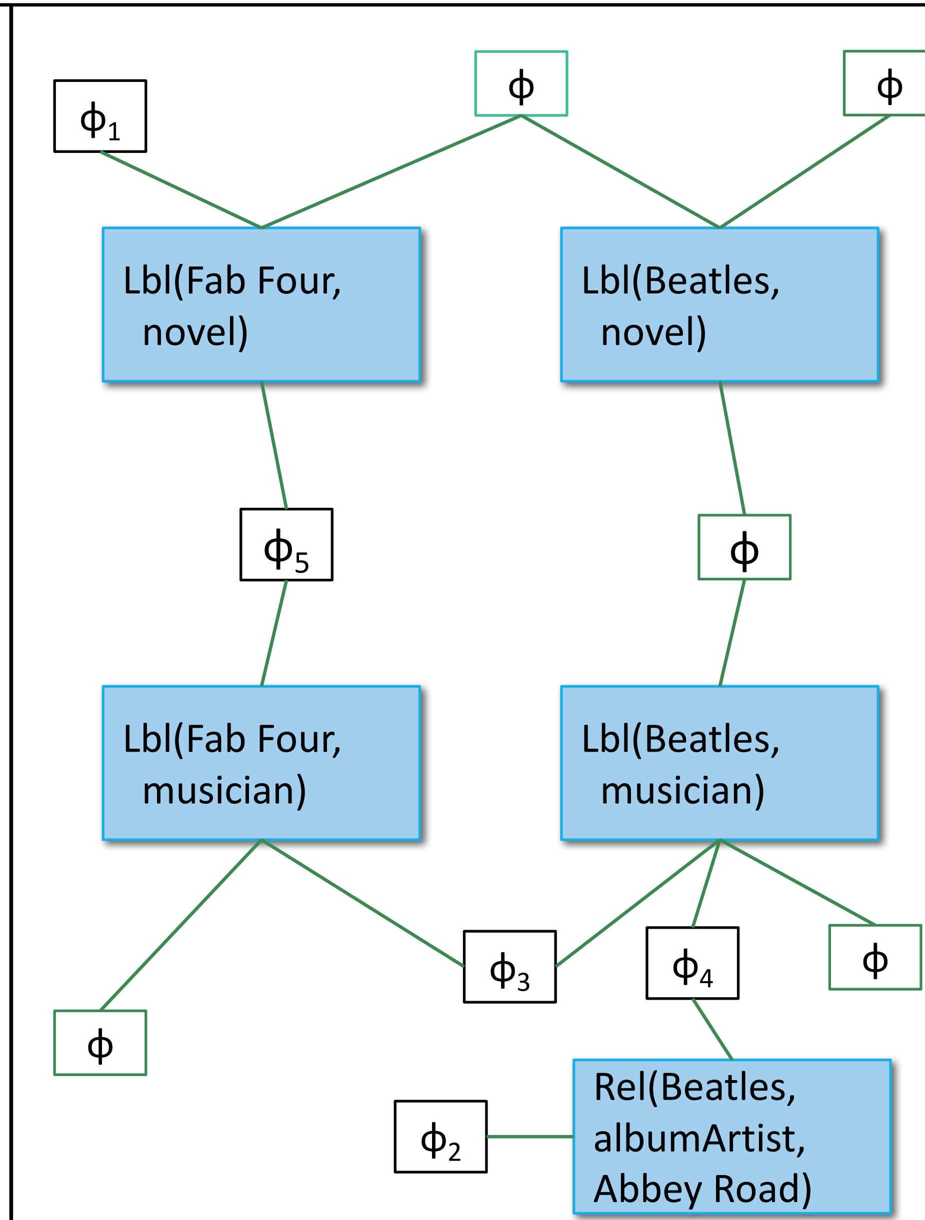
$[\phi_1]$ $\textsc{CandLbl}_{\text{struct}}(\texttt{FabFour}, \texttt{novel})$
$\Rightarrow$ $\textsc{Lbl}(\texttt{FabFour}, \texttt{novel})$

$[\phi_2]$ $\textsc{CandRel}_{\text{pat}}(\texttt{Beatles}, \texttt{AlbumArtist}, \texttt{AbbeyRoad})$
$\Rightarrow$ $\textsc{Rel}(\texttt{Beatles}, \texttt{AlbumArtist}, \texttt{AbbeyRoad})$

$[\phi_3]$ $\textsc{SameEnt}(\texttt{Beatles}, \texttt{FabFour})$
$\wedge\ \textsc{Lbl}(\texttt{Beatles}, \texttt{musician})$
$\Rightarrow \textsc{Lbl}(\texttt{FabFour}, \texttt{musician})$

$[\phi_4]$ $\textsc{Dom}(\texttt{AlbumArtist}, \texttt{musician})$
$\wedge\ \textsc{Rel}(\texttt{Beatles}, \texttt{AlbumArtist}, \texttt{AbbeyRoad})$
$\Rightarrow \textsc{Lbl}(\texttt{Beatles}, \texttt{musician})$

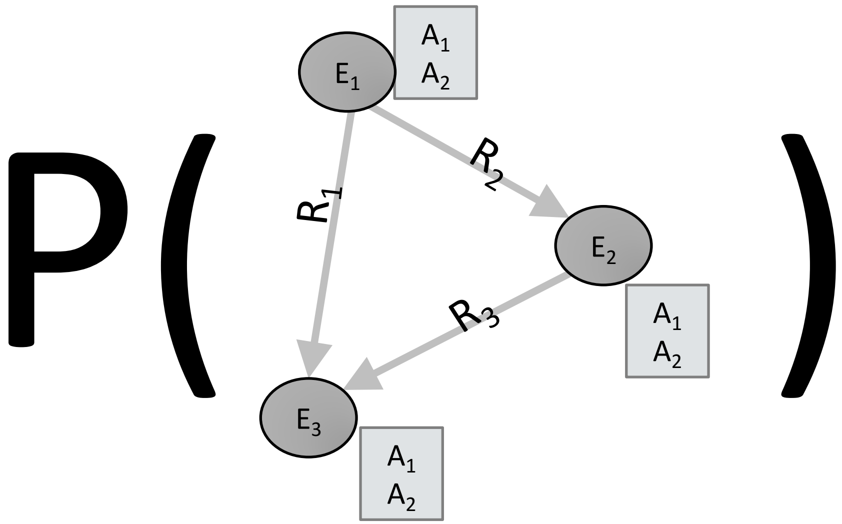
$[\phi_5]$ $\textsc{Mut}(\texttt{musician}, \texttt{novel})$
$\wedge\ \textsc{Lbl}(\texttt{FabFour}, \texttt{musican})$
$\Rightarrow \neg\textsc{Lbl}(\texttt{FabFour}, \texttt{novel})$



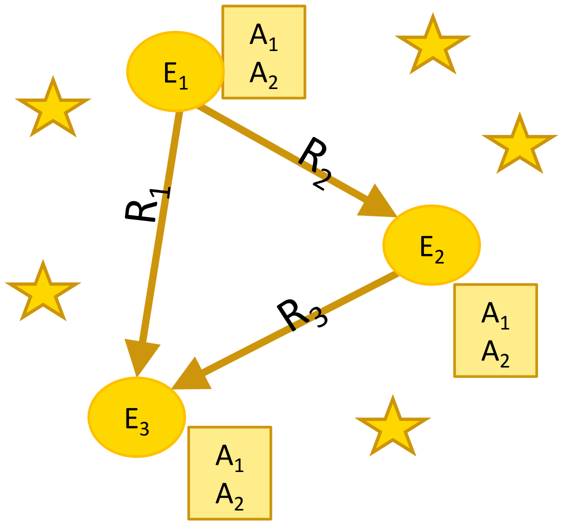

PUJARA+ISWC13; PUJARA+AIMAG15

# How do we get a knowledge graph?

Have: P(KG) forall KGs




Need: best KG




MAP inference: optimizing over distribution to find the best knowledge graph

# Inference and KG optimization

- Finding the best KG satisfying weighed rules: NP Hard

- MLNs [discrete]: Monte Carlo sampling methods
  - Solution quality dependent on burn-in time, iterations, etc.

- PSL [continuous]: optimize convex linear surrogate
  - Fast optimization, ¾-optimal MAX SAT lower bound

# Graphical Models Experiments

**Data:** ~1.5M extractions, ~70K ontological relations, ~500 relation/label types
**Task:** Collectively construct a KG and evaluate on 25K target facts

**Comparisons:**

| | |
|---|---|
| **Extract** | Average confidences of extractors for each fact in the NELL candidates |
| **Rules** | Default, rule-based heuristic strategy used by the NELL project |
| **MLN** | Jiang+, ICDM12 – estimates marginal probabilities with MC-SAT |
| **PSL** | Pujara+, ISWC13 – convex optimization of continuous truth values with ADMM |

**Running Time:** Inference completes in 10 seconds, values for 25K facts

| | **AUC** | **F1** |
|---|---|---|
| Extract | .873 | .828 |
| Rules | .765 | .673 |
| MLN (Jiang, 12) | .899 | .836 |
| PSL (Pujara, 13) | .904 | .853 |

# Graphical Models: Pros/Cons

## BENEFITS

- Define probability distribution over KGs
- Easily specified via rules
- Fuse knowledge from many different sources

## DRAWBACKS

- Requires optimization over all KG facts - overkill
- Dependent on rules from ontology/expert
- Require probabilistic semantics - unavailable

# Graph Construction
## Probabilistic Models

TOPICS:

- Overview
- Graphical models
- Random Walk Methods

# Random Walk Overview

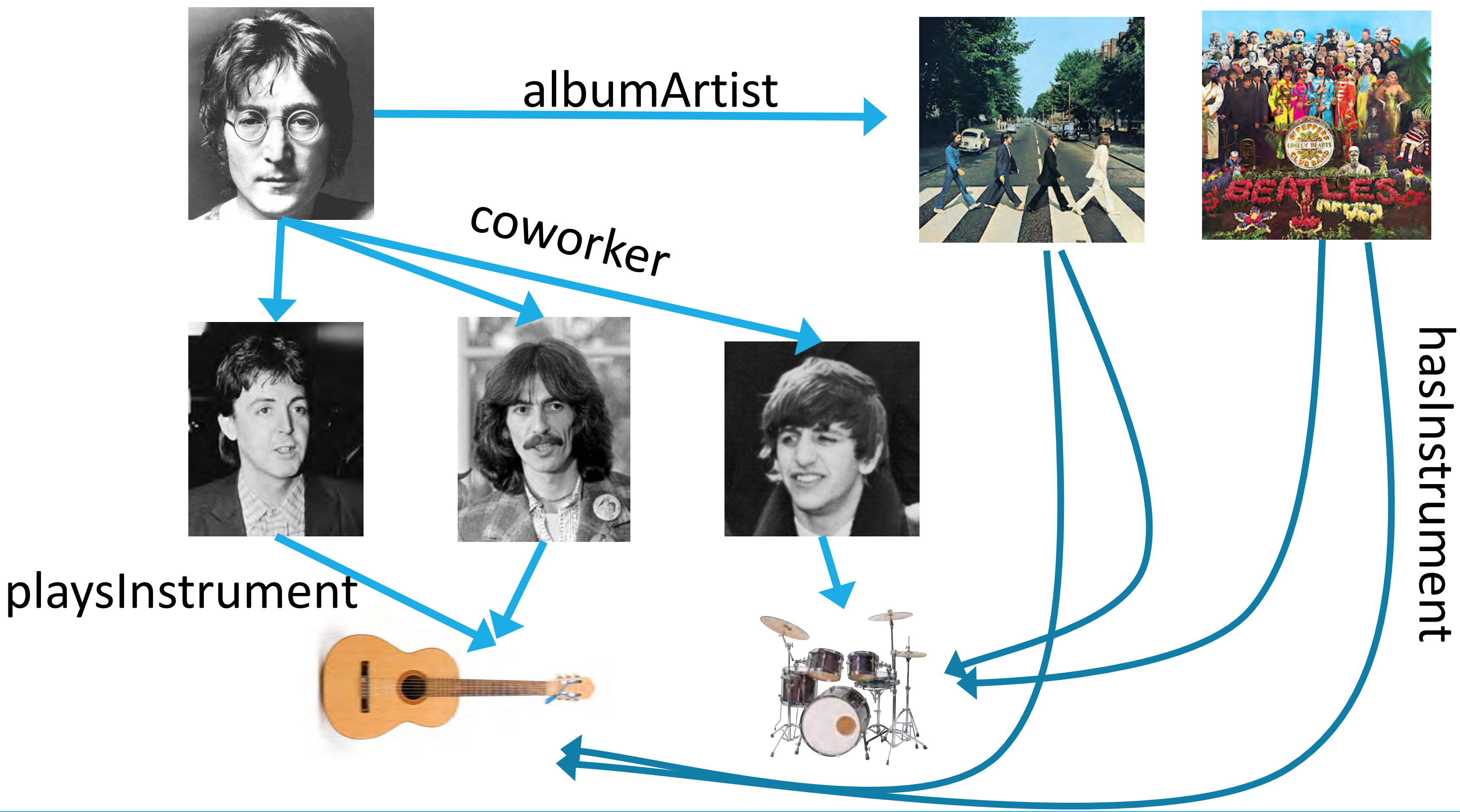
- Given: a query of an **entity** and **relation**
- Starting at the entity, **randomly walk** the KG
- Random walk ends when reaching an appropriate **goal**
- Learned **parameters** bias choices in the random walk
- Output **relative probabilities** of goal states

# Random Walk Illustration

**Query: R(Lennon, PlaysInstrument, ?)**

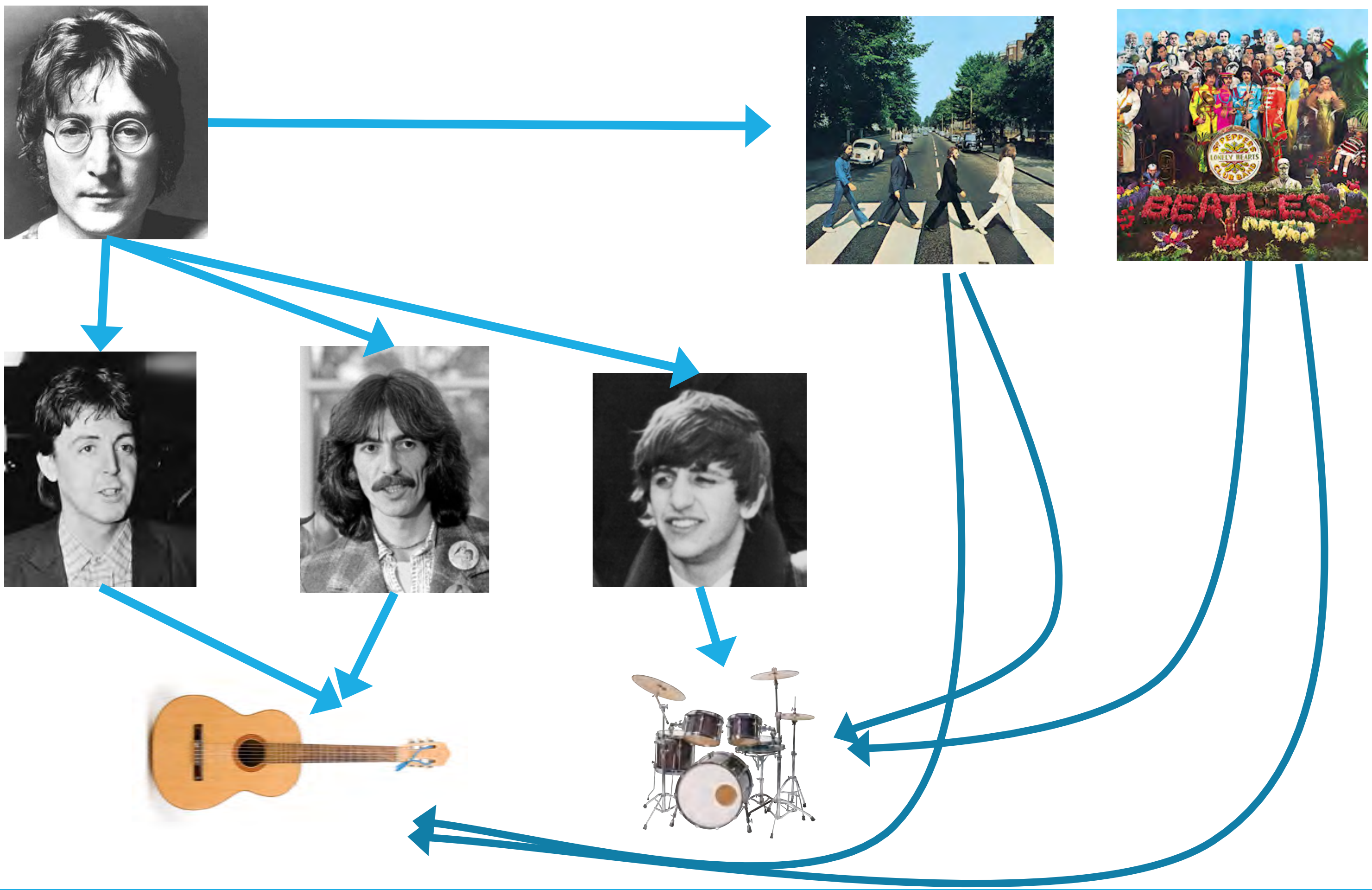
# Random Walk Illustration

**Query: R(Lennon, PlaysInstrument, ?)**



# Random Walk Illustration

**Query: R(Lennon, PlaysInstrument, ?)**



# Random Walk Illustration

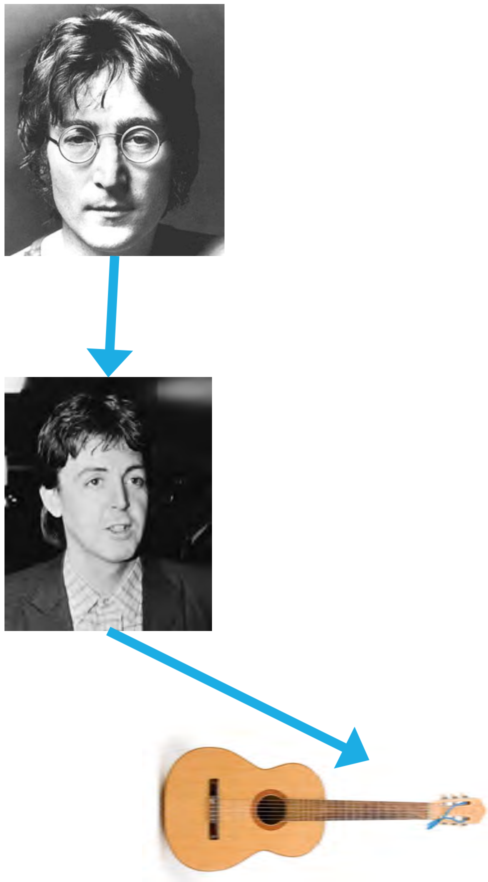
**Query Q: R(Lennon, PlaysInstrument, ?)**

# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

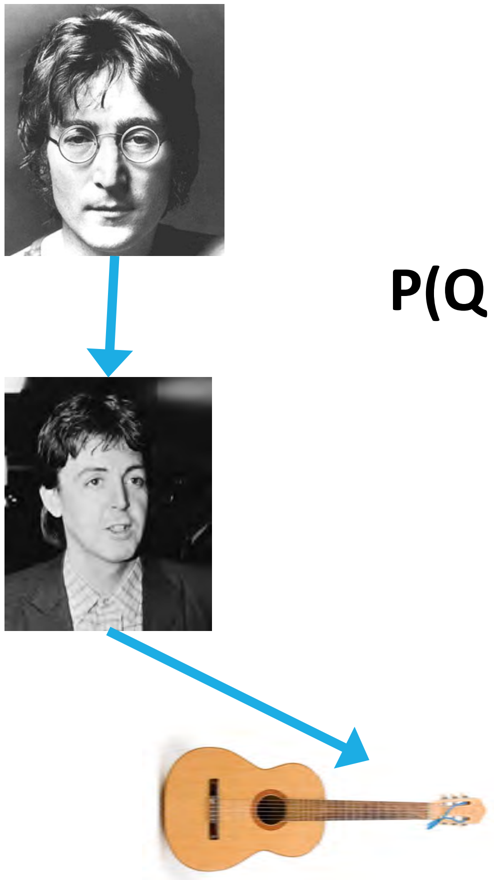
# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**



# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

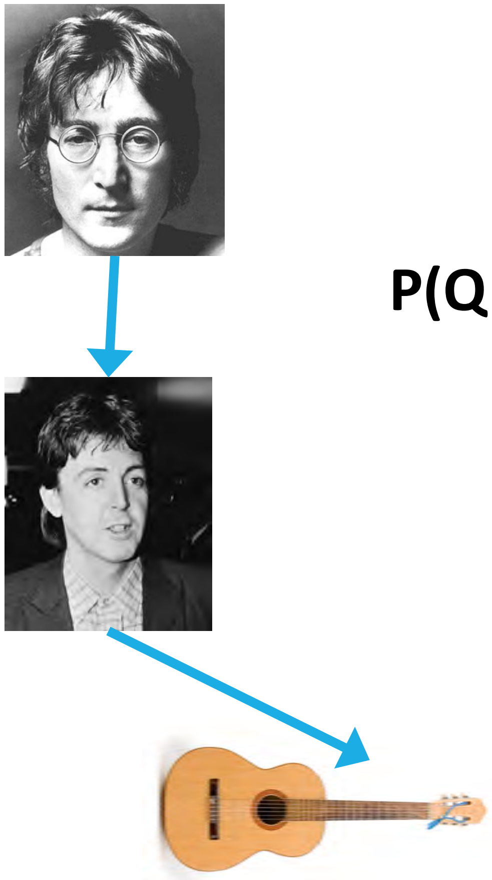


Path Weight of path

$$\mathbf{P(Q|\pi=\langle coworker,playsInstrument\rangle)\ W_{\pi}}$$

# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**



Path

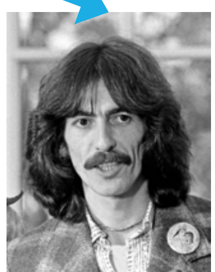
Weight of path

$$P(Q|\pi=\langle \text{coworker,playsInstrument} \rangle)\ W_{\pi}$$

# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

**$P(Q|\pi=\langle \text{coworker,playsInstrument} \rangle)\ W_\pi$**



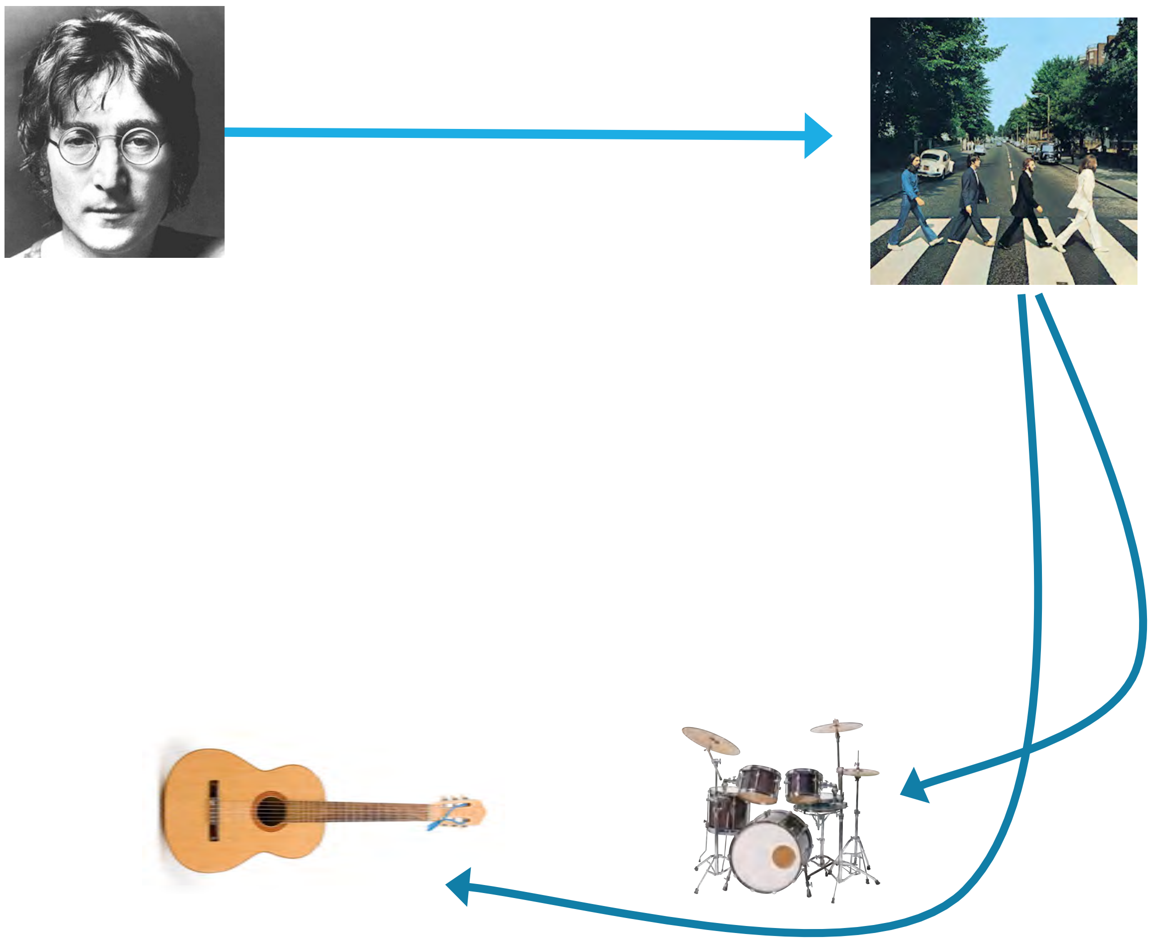
# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

$\mathbf{P(Q|\pi=<coworker,playsInstrument>)\ W_{\pi}}$

# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**



# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

**$P(Q|\pi=\langle \text{albumArtist},\text{hasInstrument}\rangle)\ W_{\pi}$**

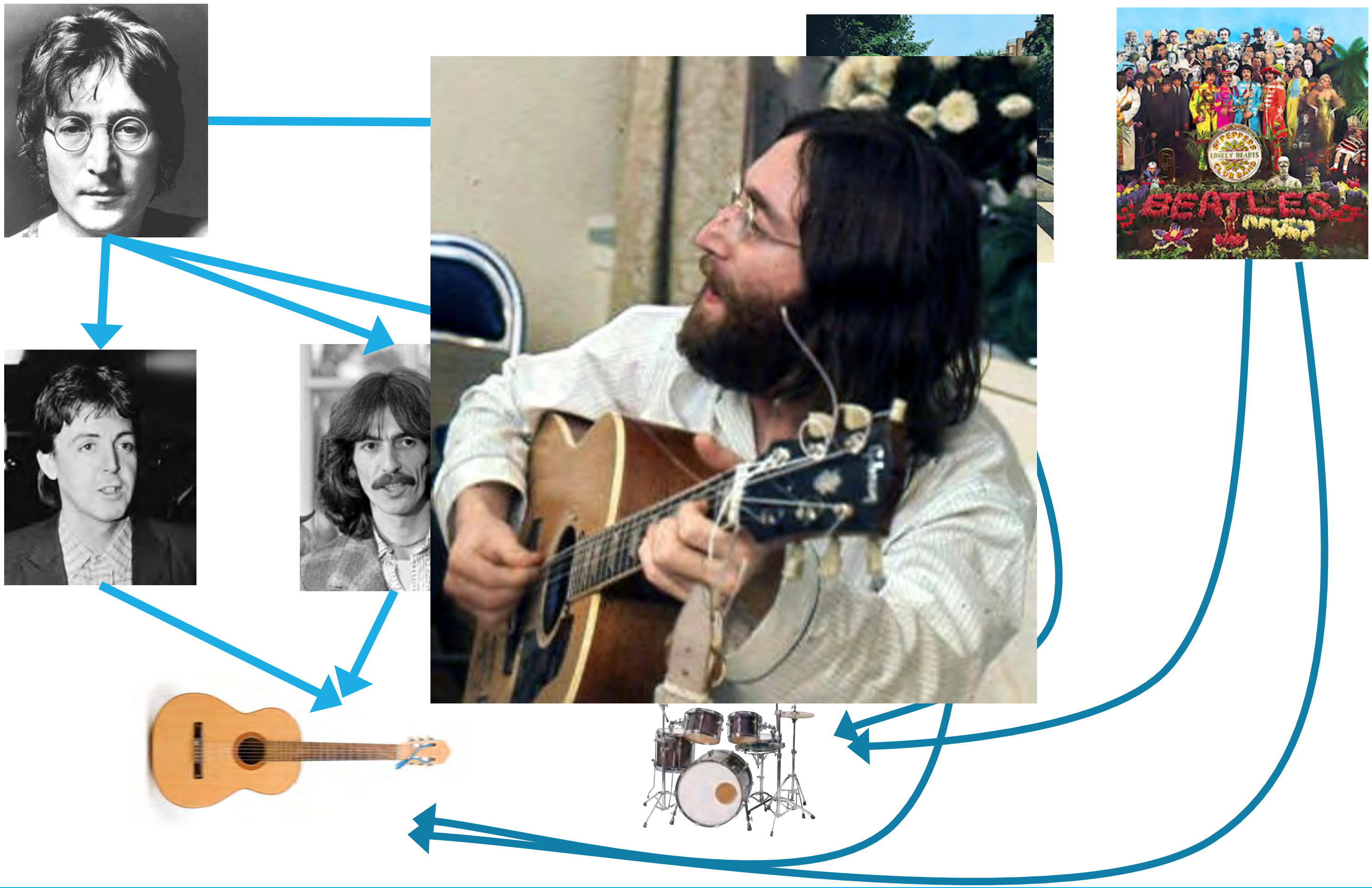
# Random Walk Illustration

**Query Q: R(Lennon, PlaysInstrument, ?)**

**P(Q|π=<albumArtist,hasInstrument>) $W_\pi$**

# Random Walk Illustration

**Query: R(Lennon, PlaysInstrument, ?)**



# Recent Random Walk Methods

## PRA: Path Ranking Algorithm

- Performs random walk of **imperfect knowledge graph**
- Estimates **transition probabilities** using KG
- For each relation, learns **parameters for paths** through the KG

## ProPPR: Programming with Personalized PageRank

- Constructs **proof graph**
  - Nodes are partially-ground clauses with one or more facts
  - Edges are proof-transformations
- **Parameters** are learned for each **ground entity** and **rule**

# Recent Random Walk Methods

## PRA: Path Ranking Algorithm

- Performs random walk of **imperfect knowledge graph**
- Estimates **transition probabilities** using KG
- For each relation, learns **parameters for paths** through the KG

## ProPPR: Programming with Personalized PageRank

- Constructs **proof graph**
  - Nodes are partially-ground clauses with one or more facts
  - Edges are proof-transformations
- **Parameters** are learned for each **ground entity** and **rule**

# PRA in a nutshell

$$\mathrm{score}(q.s \rightarrow e; q) = \sum_{\pi_i \in \Pi_b} P(q.s \rightarrow e; \pi_i) W_{\pi_i}$$

# PRA in a nutshell

$$\text{score}(q.s \to e; q) = \sum_{\pi_i \in \Pi_b} P(q.s \to e; \pi_i) W_{\pi_i}$$

Filter paths based on HITS and accuracy

# PRA in a nutshell

$$\mathrm{score}(q.s \to e; q) = \sum_{\pi_i \in \Pi_b} P(q.s \to e; \pi_i) W_{\pi_i}$$

Filter paths based on HITS and accuracy

Estimate probabilities efficiently with dynamic programming

# PRA in a nutshell

$$\mathrm{score}(q.s \to e; q) = \sum_{\pi_i \in \Pi_b} P(q.s \to e; \pi_i) W_{\pi_i}$$

Filter paths based on HITS and accuracy

Estimate probabilities efficiently with dynamic programming

Path weights are learned with logistic regression

# Recent Random Walk Methods

## PRA: Path Ranking Algorithm

- Performs random walk of **imperfect knowledge graph**
- Estimates **transition probabilities** using KG
- For each relation, learns **parameters for paths** through the KG

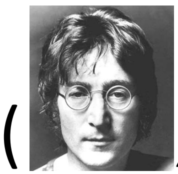
## ProPPR: ProbLog + Personalized PageRank

- Constructs **proof graph**
  - Nodes are partially-ground clauses with one or more facts
  - Edges are proof-transformations
- **Parameters** are learned for each **ground entity** and **rule**

# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**

R(  ,Coworker,X)
R(X,PlaysInstrument,Y)

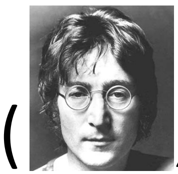
R(  ,AlbumArtist,J)
R(J,HasInstrument,K)

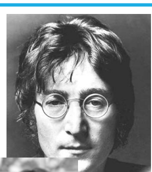
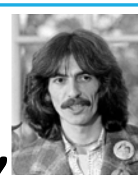
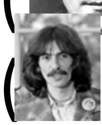
Unbound variables in proof tree!

# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**

R(,Coworker,X)
R(X,PlaysInstrument,Y)

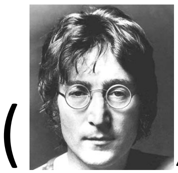
R(,AlbumArtist,J)
R(J,HasInstrument,K)

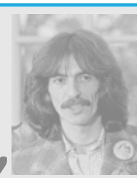
R(,Coworker,)
R(,PlaysInstrument,Y)

# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**

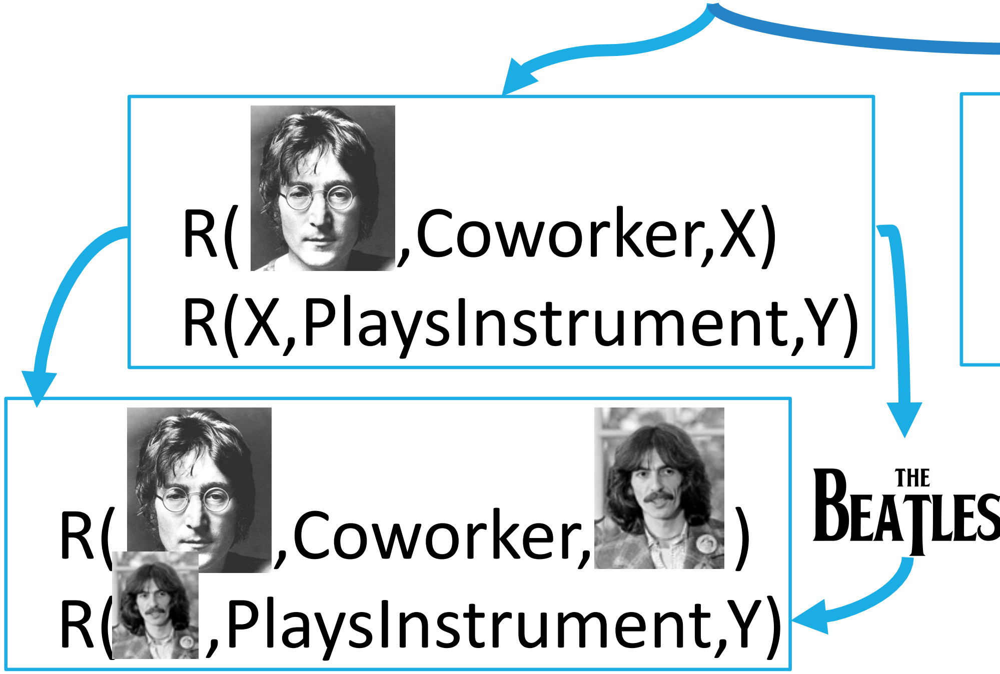
R(  ,Coworker,X)
R(X,PlaysInstrument,Y)

R(  ,AlbumArtist,J)
R(J,HasInstrument,K)

R(  ,Coworker,  )
R(  ,PlaysInstrument,Y)

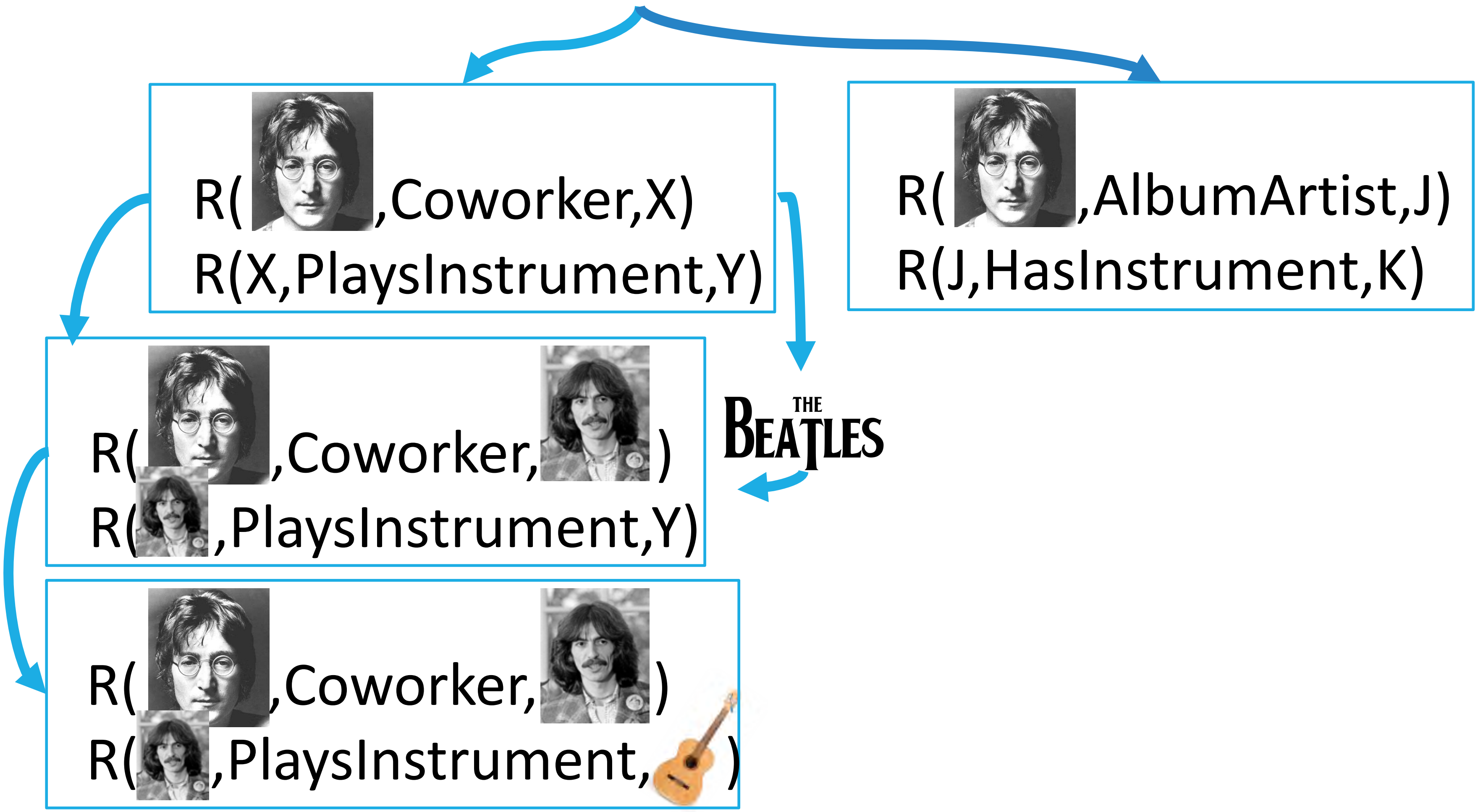
# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**



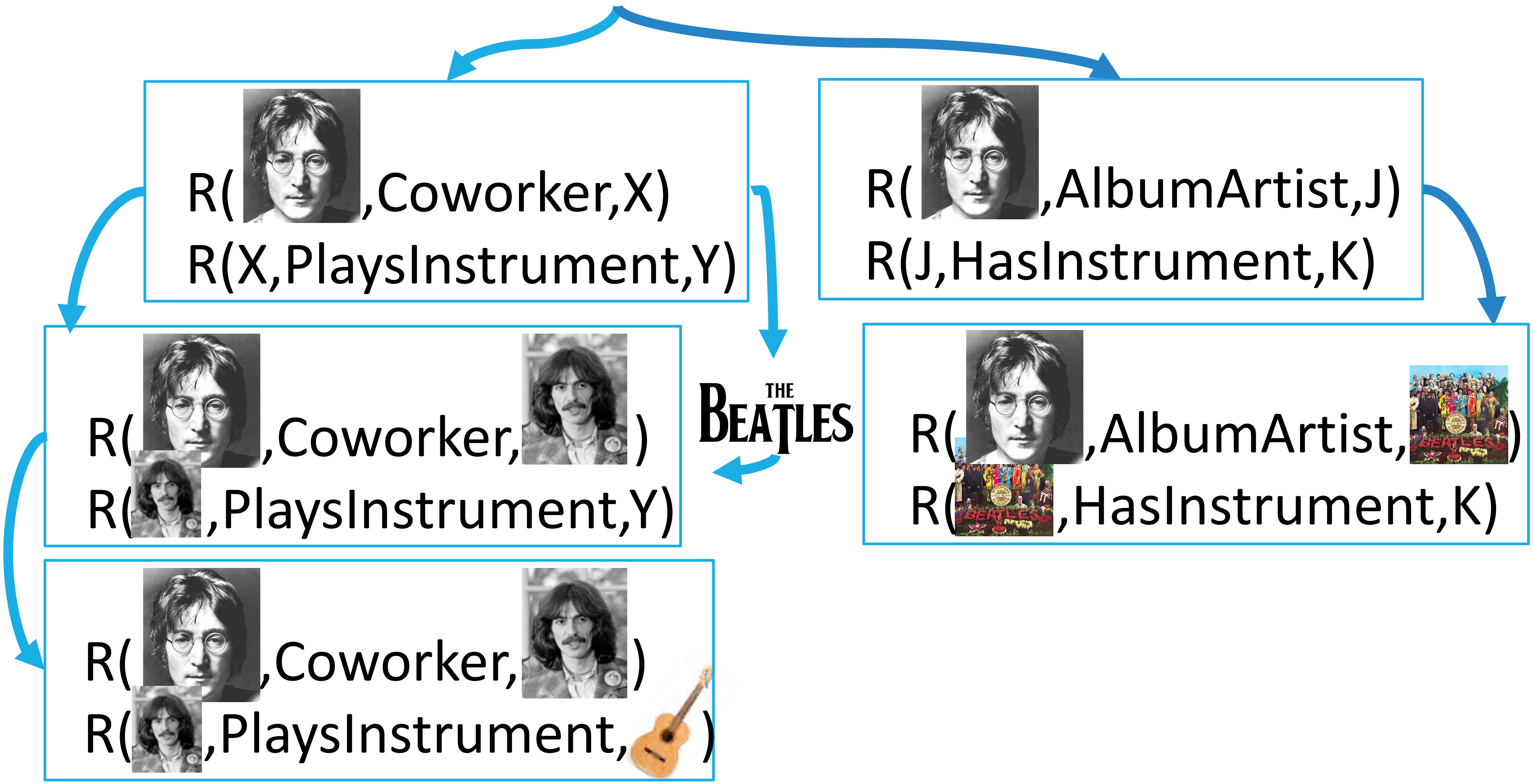
# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**



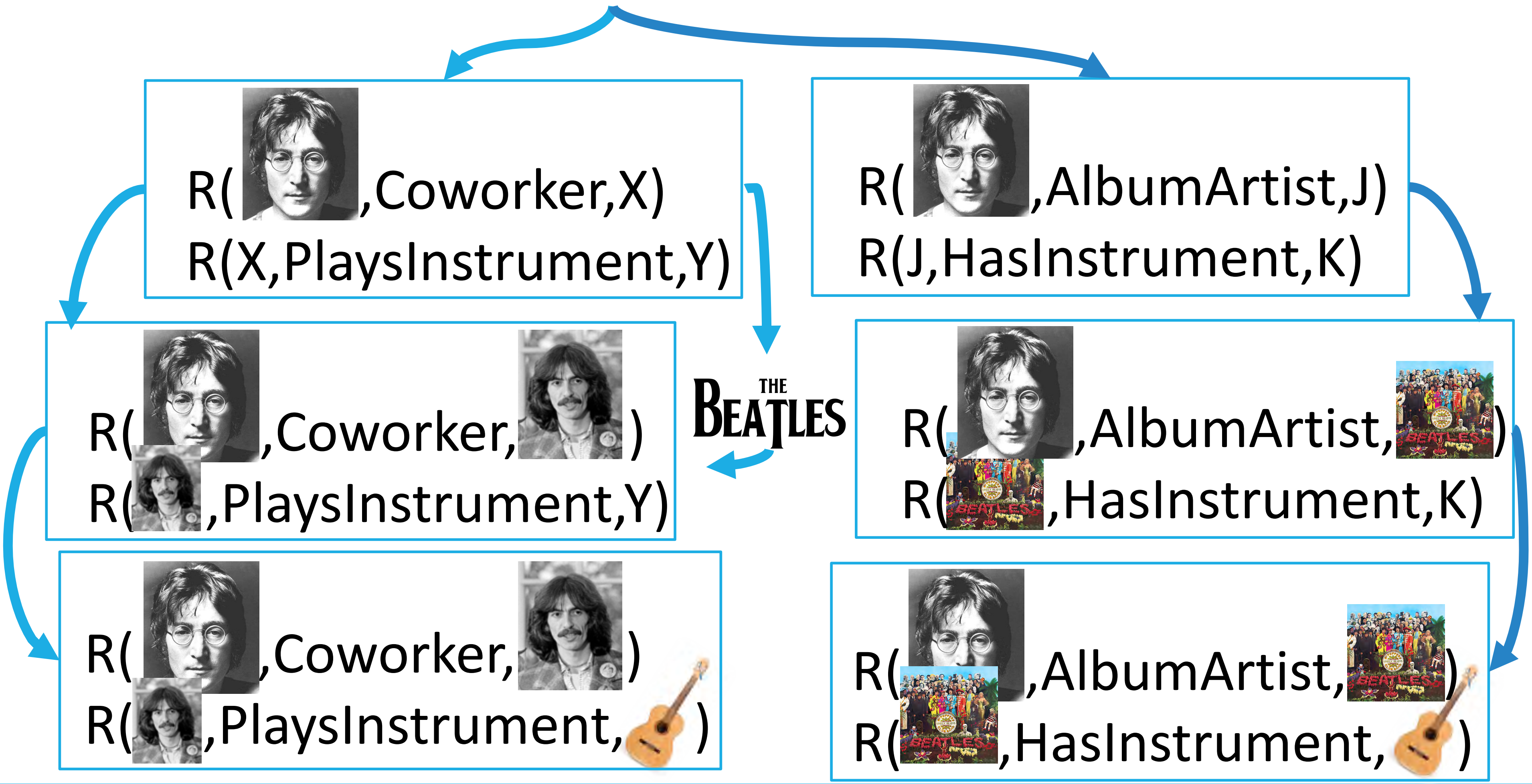
# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**



# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**



# ProPPR-ized PRA example

**Query Q: R(Lennon, PlaysInstrument, ?)**

R( ,Cowor
R(X,PlaysInstru

lbumArtist,J)
nstrument,K)

R( ,Coworker,
R( ,PlaysInstrume

,AlbumArtist, )
HasInstrument,K)

R( ,Coworker, )
R( ,PlaysInstrument, )

R( ,AlbumArtist, )
R( ,HasInstrument, )

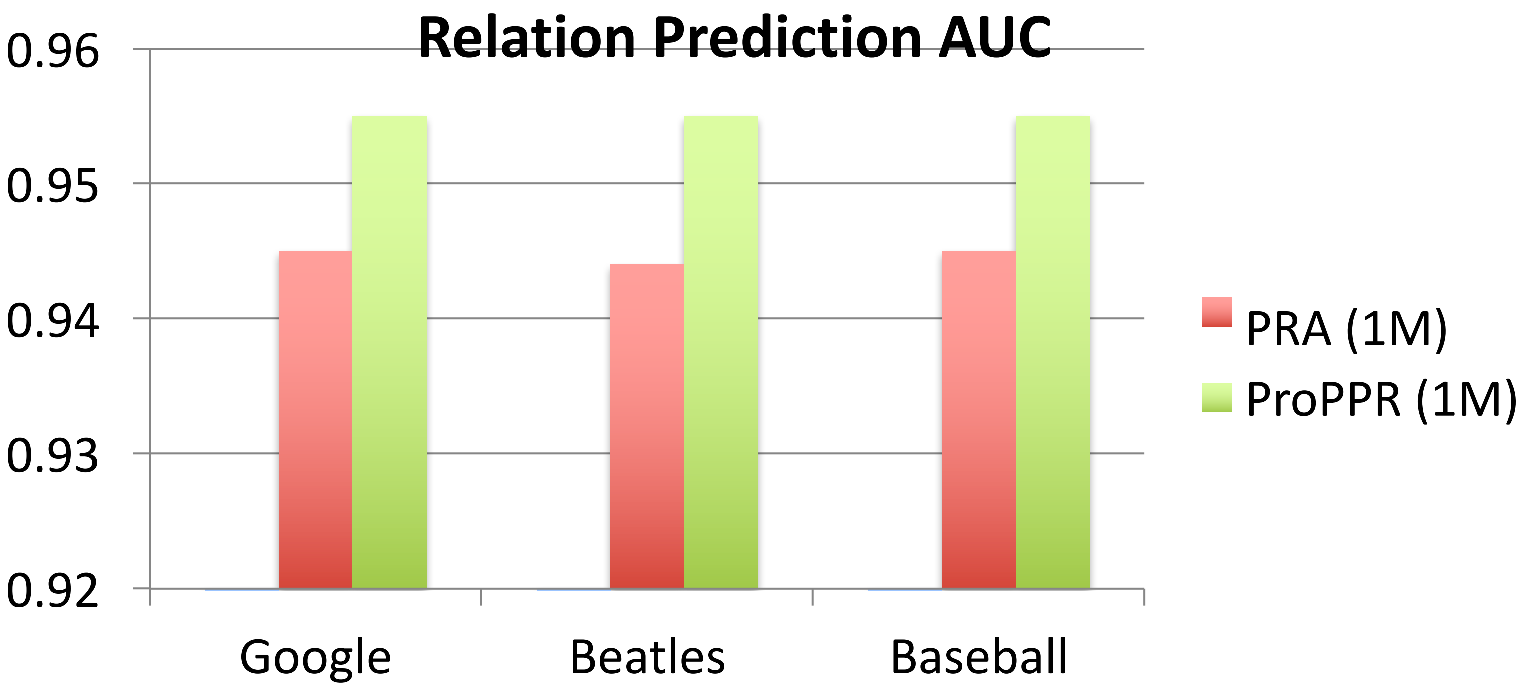
# ProPPR in a nutshell

$$\min_{\mathbf{w}} - \left( \sum_{k \in +} \log \mathbf{p}_{\nu_0}[u_+^k] + \sum_{k \in -} \log(1 - \mathbf{p}_{\nu_0}[u_-^k] \right) + \mu ||\mathbf{w}||_2^2$$

- Input: queries, positive answers, negative answers
- Goal: $\mathbf{p}_{\nu_0}[u_+^k] \geq \mathbf{p}_{\nu_0}[u_-^k]$ (page rank from RW)
- Learn: random walk weights
- Train via stochastic gradient descent

# Results from PRA and ProPPR

- Task:
  - 1M extractions for 3 domains;
  - ~100s of training queries
  - ~1000s of test queries
  - AUC of extractions alone is 0.7



# Random Walks: Pros/Cons

## BENEFITS

- KG query estimation independent of KG size
- Model training produces interpretable, logical rules
- Robust to noisy extractions through probabilistic form

## DRAWBACKS

- Full KG completion task inefficient
- Training data difficult to obtain at scale
- Input must follow probabilistic semantics

# Two classes of Probabilistic Models

## GRAPHICAL MODELS

- Possible facts in KG are variables
- Logical rules relate facts
- Probability $\propto$ satisfied rules
- Universally-quantified

## RANDOM WALK METHODS

- Possible facts posed as queries
- Random walks of the KG constitute “proofs”
- Probability $\propto$ path lengths/transitions
- Locally grounded

# Path Ranking Algorithm (PRA)

(Lao & Cohen, MLJ 2010)

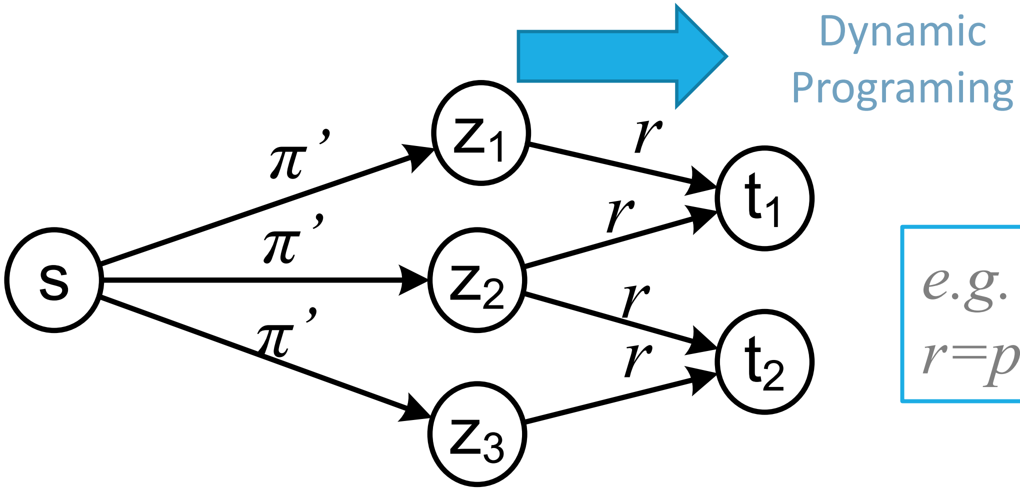
$$score(s,t) = \sum_{\pi \in \boxed{B}} \boxed{P(s \to t; \pi)}\boxed{\theta_\pi}$$

a weight

*e.g. π=<coworker,playsInstrument>*

# Random Walk Calculation

$$score(s,t) = \sum_{\pi \in B} \boxed{P(s \to t; \pi)} \theta_\pi$$




*e.g. π'=<coworker>*
*r=playsInstrument*

$$P(s \to t; \pi) = \sum_z P(s \to z; \pi')P(z \to t; r)$$

# Pruning paths using labeled data

$$score(s,t) = \sum_{\pi \in \boxed{B}} P(s \to t; \pi)\theta_\pi$$

given training query set $\{(s_i, G_i)\}$

$$hits(f) = \sum_i I\left[\sum_{j \in G_i} f(s_i, t_j)\right] \geq h$$

$$accuracy(f) = \frac{1}{N}\sum_i \left[\frac{\sum_{j \in G_i} f(s_i, t_j)}{\sum_j f(s_i, t_j)}\right] \geq a$$

I(): the indicator function

N: total number of queries

# Estimating θ via Logistic Regression

$$score(s,t) = \sum_{\pi \in B} P(s \to t; \pi) \boxed{\theta_\pi}$$

for a relation *r*

generate positive and negative node pairs $\{(s_i, t_i)\}$

for each $(s_i, t_i)$ generate $(x_i, y_i)$

$x_i$ is a vector of RW features of different paths $\pi$

$y_i$ is a binary label $r(s_i, t_i)$

estimate $\theta$ by L1/L2 regularized (elastic-net) logistic regression

# Rules: Uncertain Extractions

Weight for signal T (relations)

Predicate representing uncertain relation extraction using signal T

Relation in Knowledge Graph

$$\mathbf{w_{CR-T}} : \ \textsc{CandRel}_T(E_1, E_2, R) \quad \Rightarrow \quad \textsc{Rel}(E_1, E_2, R)$$

$$\mathbf{w_{CL-T}} : \ \textsc{CandLbl}_T(E, L) \quad \Rightarrow \quad \textsc{Lbl}(E, L)$$

Weight for signal T (labels)

Predicate representing uncertain label extraction using signal T

Label in Knowledge Graph

# Rules: Entity Resolution

$$\mathbf{w_{EL}} : \textsc{SameEnt}(E_1, E_2) \wedge \textsc{Lbl}(E_1, L) \Rightarrow \textsc{Lbl}(E_2, L)$$
$$\mathbf{w_{ER}} : \textsc{SameEnt}(E_1, E_2) \wedge \textsc{Rel}(E_1, E, R) \Rightarrow \textsc{Rel}(E_2, E, R)$$
$$\mathbf{w_{ER}} : \textsc{SameEnt}(E_1, E_2) \wedge \textsc{Rel}(E, E_1, R) \Rightarrow \textsc{Rel}(E, E_2, R)$$

SameEnt predicate captures confidence that entities are co-referent

- Rules require co-referent entities to have the same labels and relations

- Creates an *equivalence class* of co-referent entities

# Rules: Ontology

**Inverse:**

$$\mathbf{w_O} : \textsc{Inv}(R, S) \wedge \textsc{Rel}(E_1, E_2, R) \Rightarrow \textsc{Rel}(E_2, E_1, S)$$

**Selectional Preference:**

$$\mathbf{w_O} : \textsc{Dom}(R, L) \wedge \textsc{Rel}(E_1, E_2, R) \Rightarrow \textsc{Lbl}(E_1, L)$$
$$\mathbf{w_O} : \textsc{Rng}(R, L) \wedge \textsc{Rel}(E_1, E_2, R) \Rightarrow \textsc{Lbl}(E_2, L)$$

**Subsumption:**

$$\mathbf{w_O} : \textsc{Sub}(L, P) \wedge \textsc{Lbl}(E, L) \Rightarrow \textsc{Lbl}(E, P)$$
$$\mathbf{w_O} : \textsc{RSub}(R, S) \wedge \textsc{Rel}(E_1, E_2, R) \Rightarrow \textsc{Rel}(E_1, E_2, S)$$

**Mutual Exclusion:**

$$\mathbf{w_O} : \textsc{Mut}(L_1, L_2) \wedge \textsc{Lbl}(E, L_1) \Rightarrow \neg\textsc{Lbl}(E, L_2)$$
$$\mathbf{w_O} : \textsc{RMut}(R, S) \wedge \textsc{Rel}(E_1, E_2, R) \Rightarrow \neg\textsc{Rel}(E_1, E_2, S)$$

(JIANG ET AL., ICDM 2012;
PUJARA ET AL., ISWC13)