Neuro-symbolic methods for KG Complex Queries

Zihao Wang

CSE, HKUST

Table of content

1. Background
2. Problem definition and general strategy
3. Tree-Form Queries (TFQs) and their solution
4. TFQs and Existential First Order (EFO) queries
5. Neuro-symbolic solutions for EFO queries

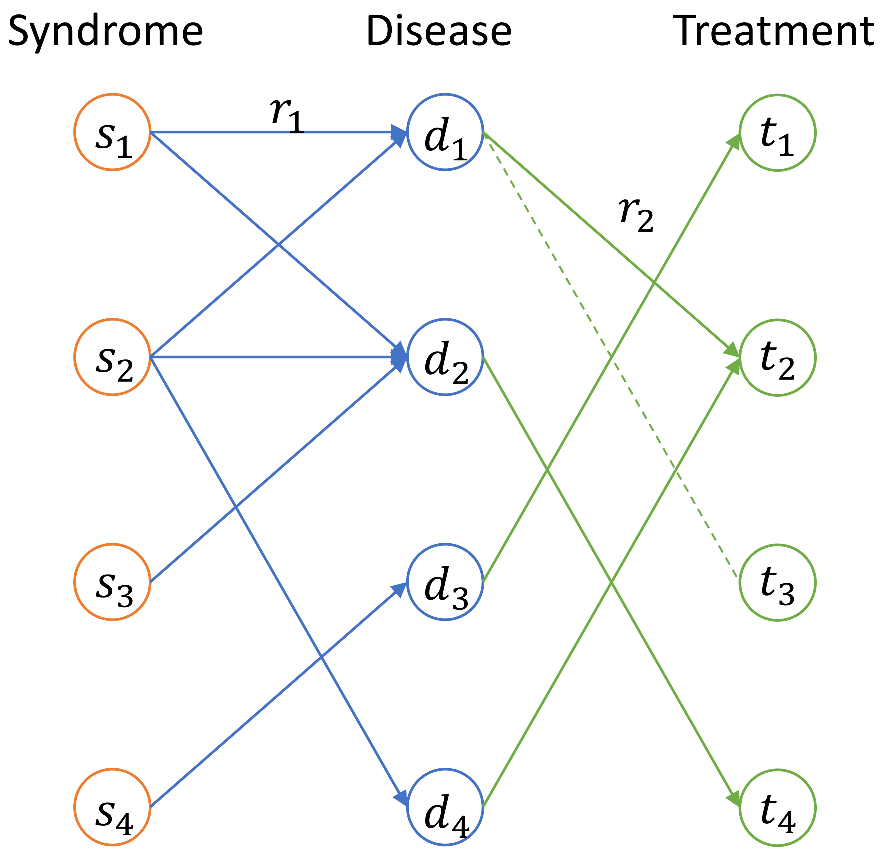
1. Background

1.1 Logical query examples

1.2 Motivations and challenges for neuro-symbolic approach

Background: Logical query examples

Logical query in natural language	Logical query in formal language
What are the treatments for disease "d_4"?	$y.\, r_2(d_4, y)$
What are the treatments for syndrome "s_1"?	$y.\, \exists x.\, r_1(s_1, x) \wedge r_2(x, y)$
What are the common syndromes of diseases "d_1" and "d_4"?	$y.\, r_1(y, d_1) \wedge r_1(y, d_4)$
What are the treatment for "d_1" but NOT "d_4"	$y.\, r_2(d_1, y) \wedge \neg r_2(d_4, y)$
What are the syndromes for diseases "d_1" or "d_2"	$y.\, r_1(y, d_1) \vee r_1(y, d_2)$

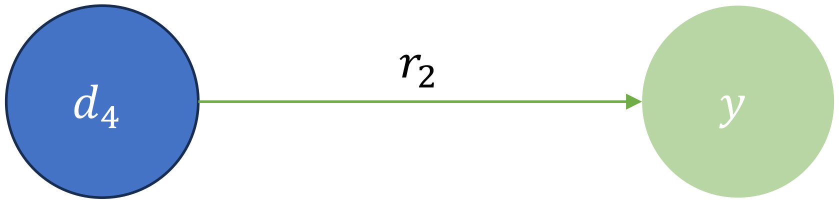


A simplified medical KG.
A more recent and complex one: https://www.nature.com/articles/s41597-023-01960-3

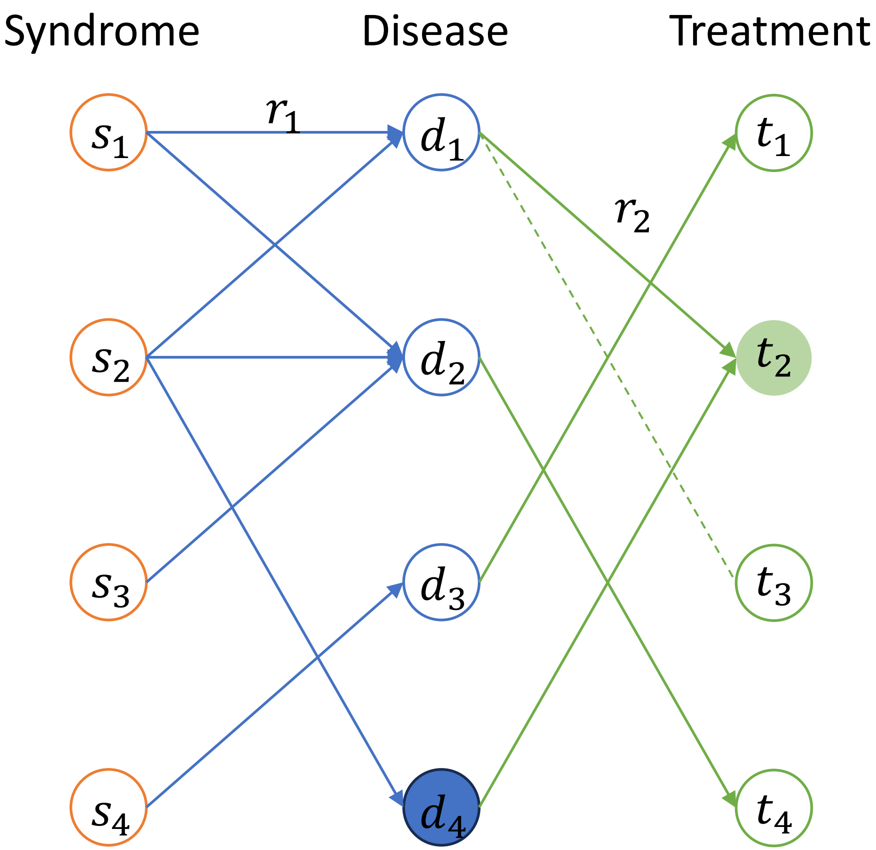
Background: Logical query examples

Logical query in natural language	Logical query in formal language
What is the treatment for disease “d_4”?	$y.r_2(d_4, y)$

A query graph representation of
One-hop query / Link prediction



Nodes in a query graph: entities and variables
Edges in a query graph: the logical predicate (with negation)

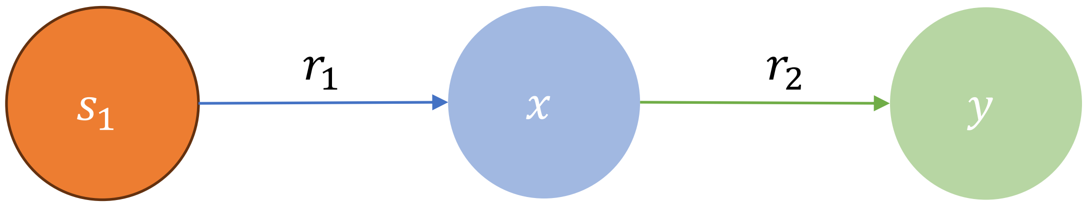


Traversal on the knowledge graph

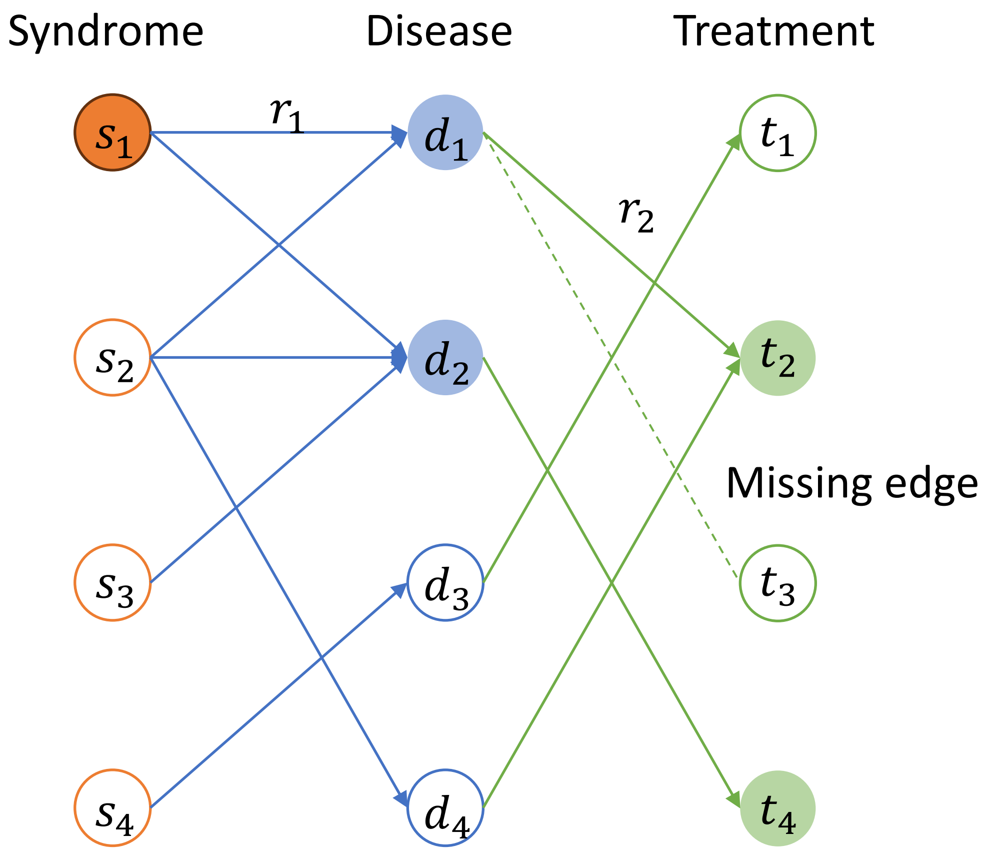
Background: Logical query examples

Logical query in natural language	Logical query in formal language
What are the treatments for syndrome “s_1”?	$y.\exists x. r_1(s_1, x) \wedge r_2(V_1, x)$

A query graph representation for multi-hop query / path query



Nodes in a query graph: entities and variables
Edges in a query graph: the logical predicate (with negation)

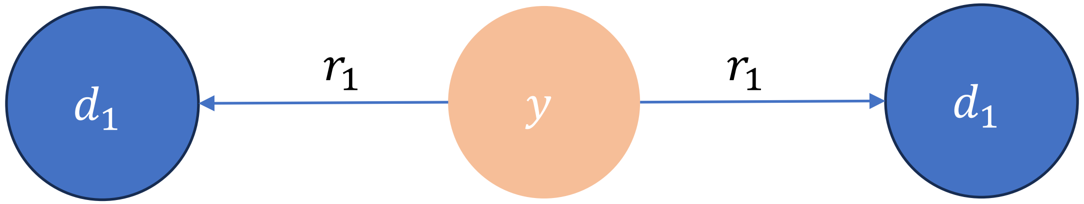


Traversal on the knowledge graph

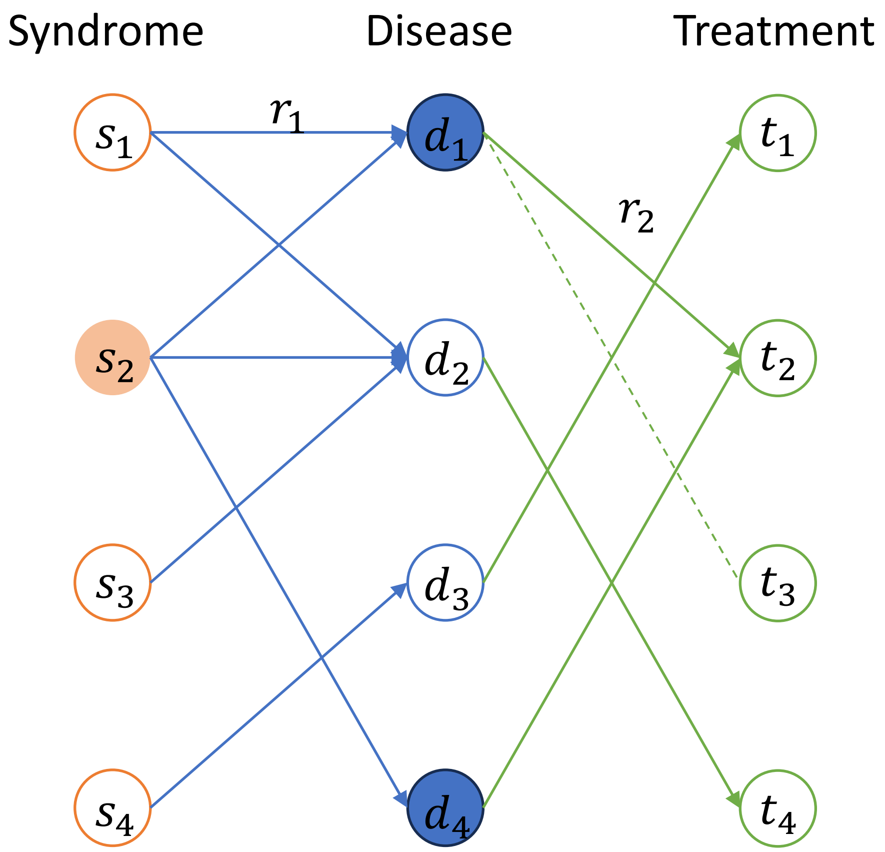
Background: Logical query examples

Logical query in natural language	Logical query in formal language
What is the common syndrome of diseases "d_1" and "d_4"?	$y.\, r_1(y, d_1) \wedge r_1(y, d_4)$

A query graph representation for multi-constraint query



Nodes in a query graph: entities and variables
Edges in a query graph: the logical predicate (with negation)

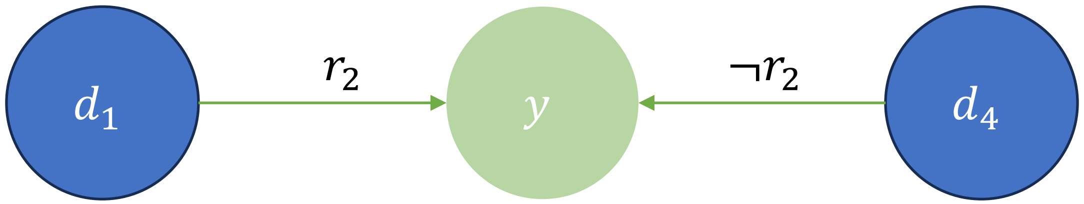


Traversal on the knowledge graph

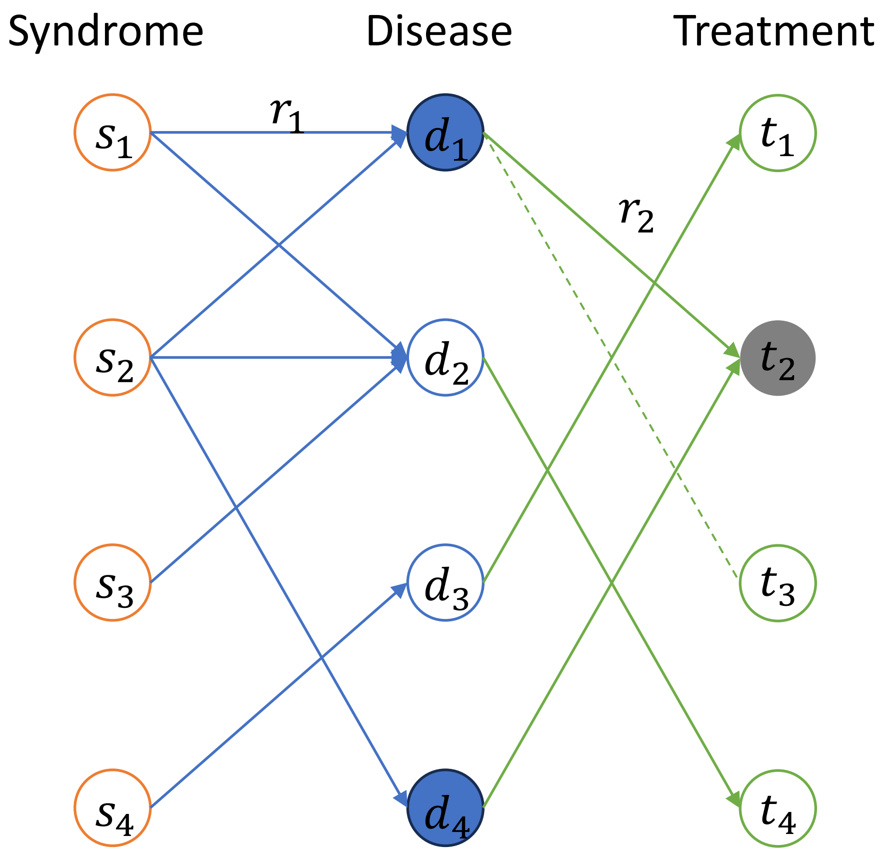
Background: Logical query examples

Logical query in natural language	Logical query in formal language
What are the treatment for “d_1” but NOT “d_4”	$y.r_2(d_1,y) \wedge \neg r_2(d_4,y)$

A query graph representation for
multi-constraint + logical negation query



Nodes in a query graph: entities and variables
Edges in a query graph: the logical predicate (with negation)

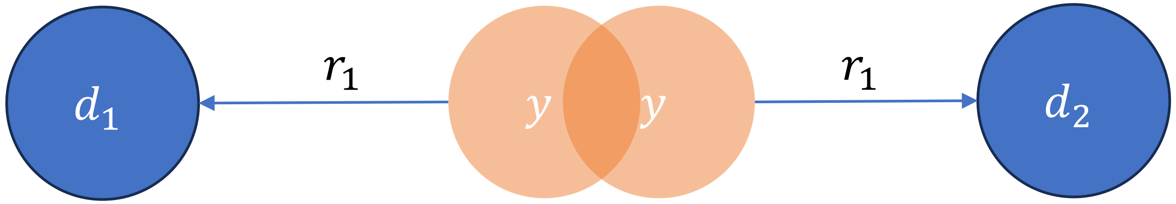


Traversal on the knowledge graph

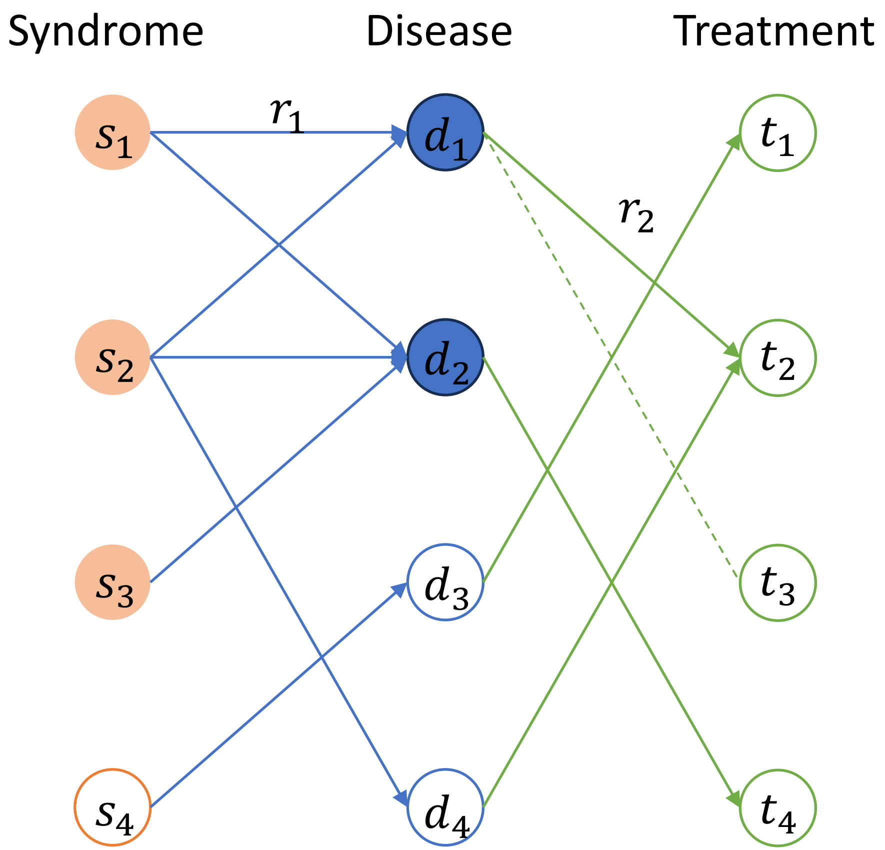
Background: Logical query examples

Logical query in natural language	Logical query in formal language
What are the syndromes for diseases “d_1” or “d_2”	$y.r_1(y, d_1) \vee r_1(y, d_2)$

A query graph representation for multi-constraint disjunctive query



Nodes in a query graph: entities and variables
Edges in a query graph: the logical predicate (with negation)
Simple graph CANNOT represent the disjunction operation. Two components for two parts



Traversal on the knowledge graph

Background: Summary

Questions for rigorous research:

- How to define the semantics and syntax?
- How to represent and answer a query?

Challenge for knowledge graph in general:

- The graph is incomplete, traversal will fail!

Background: Summary

- Semantics of queries
  - Highly interpretable with logical conditions,
  - Related to complex demands of knowledge.
- Syntax of queries
  - variables and entities,
  - logical connectives:
    - Conjunction
    - Disjunction
    - Negation
- Representation
  - A query graph
- In the previous examples, the answers are identified by
  - Symbolic traversal on the knowledge graph

2. Problem definition and general strategies

2.1 Notations and definitions

2.2 Two general strategies

Problem definition and general strategies Notations

Data

- A knowledge graph, triple set $\mathcal{KG}_o = \{(h, r, t)\}$,
- For simplicity, the relations $\mathcal{R}$ and entities $\mathcal{E}$ are assumed to be known.
  - (This assumption can be undermined)

Open World Assumption (OWA)

- An observed knowledge graph $\mathcal{KG}_o$,
- An unobserved knowledge graph $\mathcal{KG}_u$,
- $\mathcal{KG}_o \subset \mathcal{KG}_u$

Problem definition and general strategies
Query syntax

Syntax of Existential First Order (EFO) query family
(organized as Unions of Conjunctive Queries (UCQ))

- An UCQ query is represented as the disjunction of conjunctive queries,
$$UCQ(y; x_1, \dots, x_n) = \bigvee_{j=1,\dots,N} CQ_j(y; x_1, \dots, x_n)$$
- Each conjunctive query is the conjunctive of atomic formulas,
$$CQ_j(y; x_1, \dots, x_n) = y. \exists x_1, \dots, \exists x_n. \bigwedge_{k=1,\dots,M_j} a_{jk}$$
- Each atomic formula is $a_{jk} = r(t_s, t_o)$, or $a_{jk} = \neg r(t_s, t_o)$,
  - where r is the binary relation in KG,
  - t_s and t_o are the subjective/objective terms, respectively,
  - Each term is either an entity or variable $(y, x_1, \dots, x_n)$.

Problem definition and general strategies
Query semantics

The answer set $A = \{a \in \mathcal{E}: Q(y = a; x_1, \ldots, x_n) = \text{True}\}$, depends on the <u>semantics of the substitution</u> $Q(y = a; x_1, \ldots, x_n)$.

Q: How to evaluate $Q(y = a; x_1, \ldots, x_n)$?

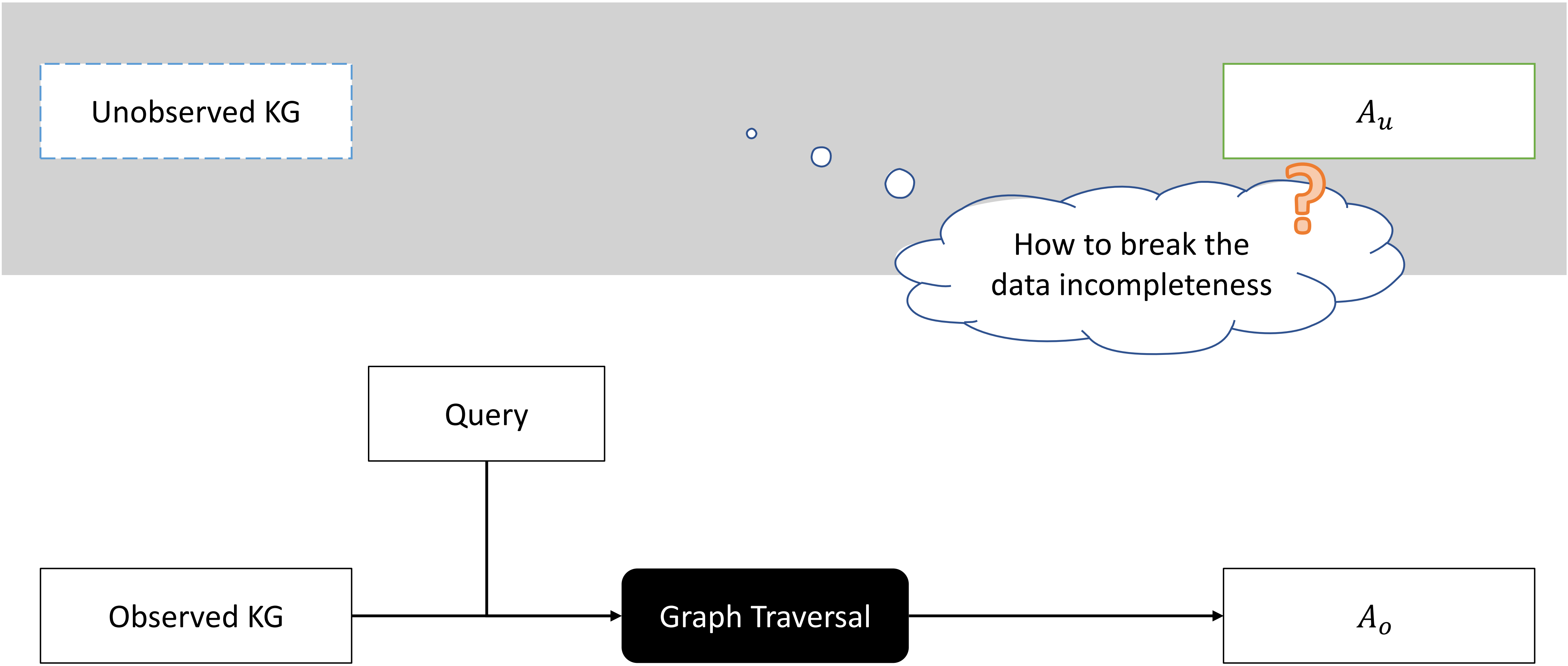
A: Evaluate the expansion, reduce to the model checking problem

$$\bigvee_{j=1,\ldots,N} \exists x_1, \ldots, \exists x_n. \bigwedge_{k=1,\ldots,M_j} a_{jk} \Big|_{y=a}$$

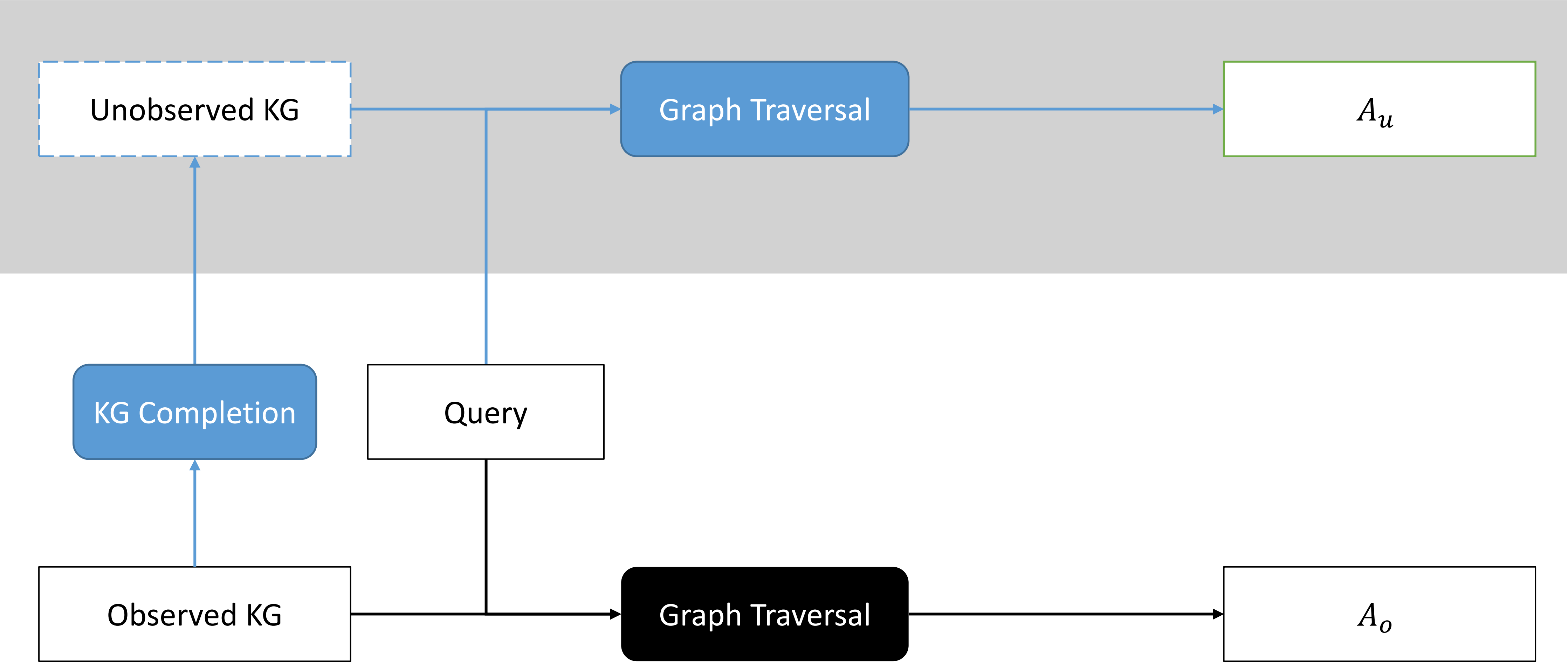
Then, it eventually depends on each atomic formula a_{jk}.

- Closed world evaluation (Answer set A_o):
  - $r(s, o) = \text{True}$ if and only if $(\mathrm{s}, \mathrm{r}, \mathrm{o}) \in \mathcal{KG}_o$, which is traversal on observed KG.
- Open world evaluation (Answer set A_u):
  - $r(s, o) = \text{True}$ if and only if $(\mathrm{s}, \mathrm{r}, \mathrm{o}) \in \mathcal{KG}_u$, where $\mathcal{KG}_u$ is unobserved.

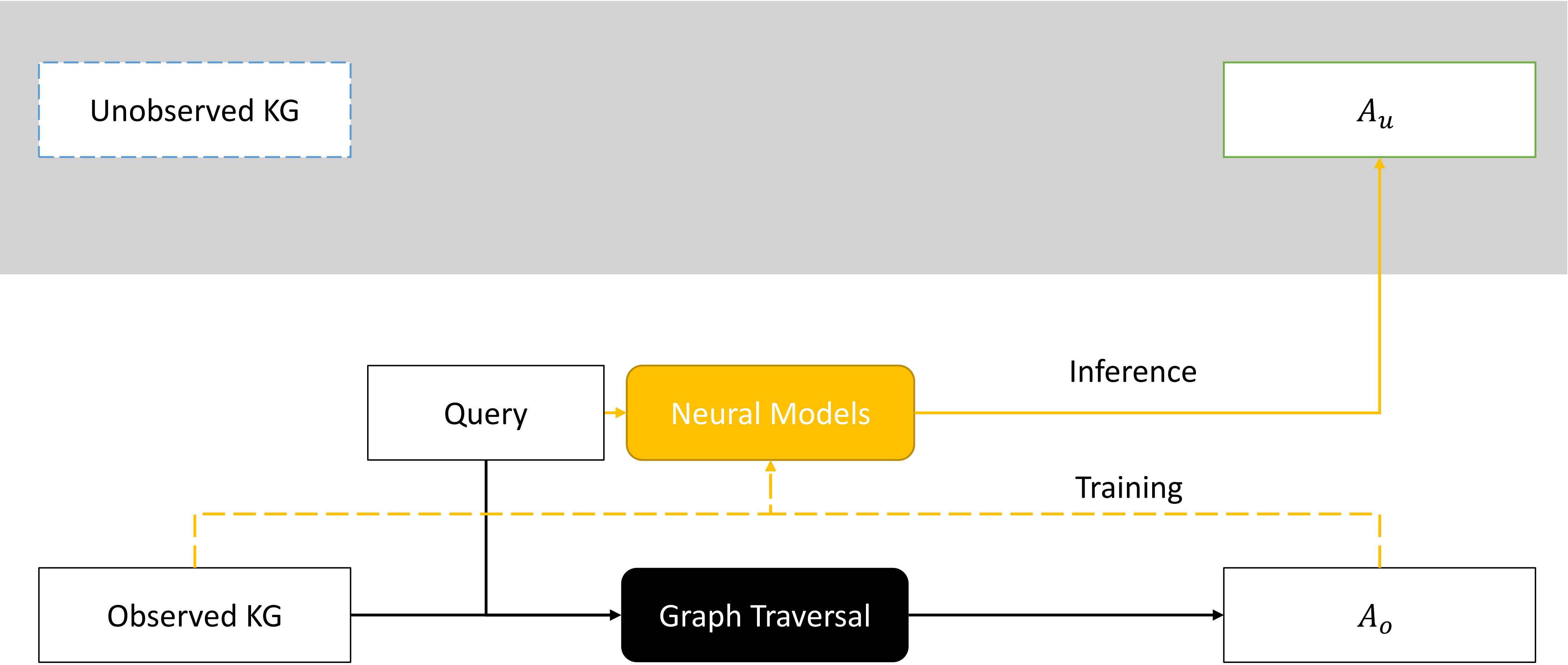
Problem definition and general strategies
The challenge of the open world problem



Problem definition and general strategies
Symbolic strategy: completion and search



Problem definition and general strategies
Neural strategy: end-to-end training



3. Tree-Formed Queries (TFQ)

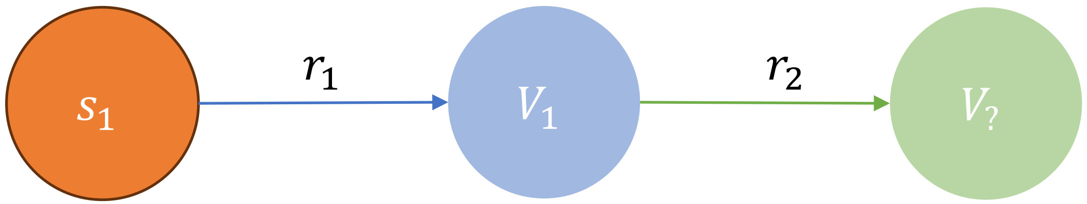
Reduction to computational graphs

The design space of neural models for TFQ

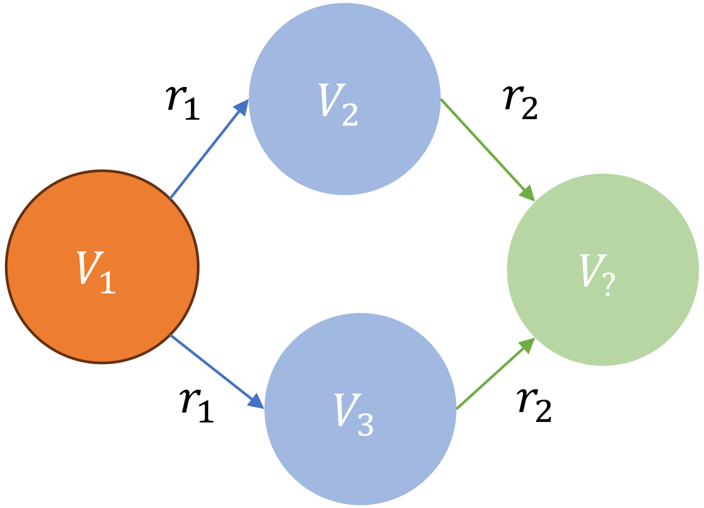
Tree-Formed Queries (TFQ)

✓We can interpretate logical queries in natural language

- How can we compute logical queries?
  - In general the symbolic methods are NP-complete.
- What are simpler query families to handle?
  - Convert the query answering process as set operations.
  - Result in a computational tree.



A simple query graph

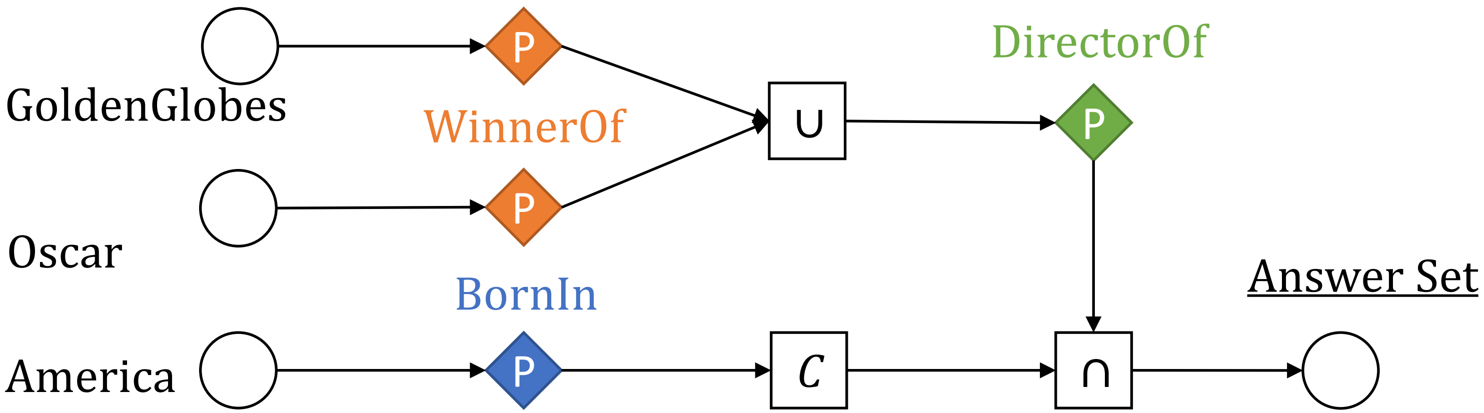


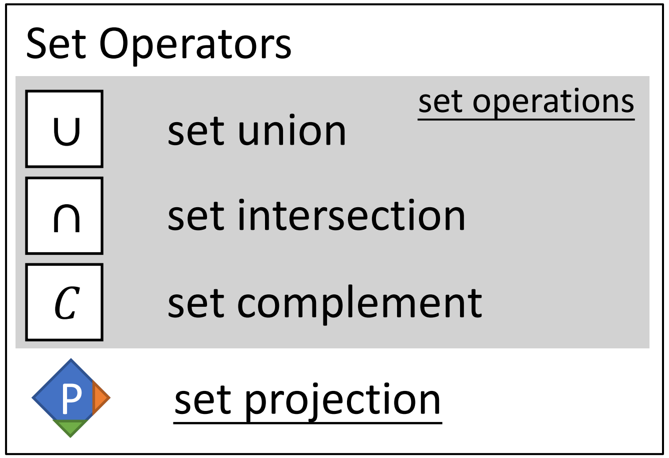
A complex query graph

Tree-Formed Queries (TFQ)
We begin with a working example

- Tree-form query family contains the queries that can be converted into the computational tree.
- What is a computational tree? A working example

Natural Language: Find non-American directors whose movie won Golden Globes or Oscar?
Logical Formula: $q = V_? \exists V_1.(\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \vee \mathrm{Won}(V_1, \mathrm{Oscar})) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(V_?, V_1)$
Set Operator Tree: $\mathrm{DirectorOf}(\mathrm{WinnerOf}(\mathrm{GoldenGlobes}) \cup \mathrm{WinnerOf}(\mathrm{Oscar})) \cap \mathrm{BornIn}(\mathrm{America})^C$





Tree-Formed Queries (TFQ)
A working example (1/4)

We use $V_1, V_?$ instead of x, y to demonstrate quantifiers are more fundamental than choices of letters.

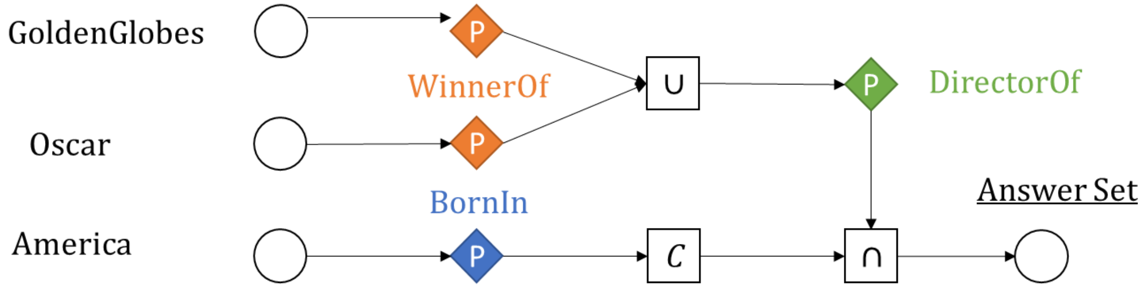
$$q = V_? \exists V_1 . \boxed{(\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \vee \mathrm{Won}(V_1, \mathrm{Oscar})}) \wedge \boxed{\neg \mathrm{BornIn}(V_?, \mathrm{America})} \wedge \boxed{\mathrm{Direct}(V_?, V_1)}$$

- Skolemization
  - $\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \xrightarrow{\text{Skolemization}} \widehat{V_1} = \mathrm{WinnerOf}(\mathrm{GoldenGlobes})$
  - $\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \xrightarrow{\text{Skolemization}} \widehat{V_1} = \mathrm{WinnerOf}(\mathrm{Oscar})$
  - $\mathrm{Direct}(V_?, V_1) \xrightarrow{\text{Skolemization}} \widehat{V_?} = \mathrm{DirectorOf}(V_1)$
  - $\neg\, \mathrm{BornIn}(V_?, \mathrm{America}) \xrightarrow{\text{Skolemization}} \widehat{V_?} = (\neg \mathrm{BornIn})(\mathrm{America})$

Skolemization:
One way to eliminate the quantified variables

- Remove V_1 and replace the connectives with set operations, then we get computational tree.

$$q = \boxed{\mathrm{DirectorOf}} \boxed{(\mathrm{WinnerOf}(\mathrm{GoldenGlobes}) \cup \mathrm{WinnerOf}(\mathrm{Oscar})}) \cap \boxed{\mathrm{BornIn}(\mathrm{America})^C}$$



1. Eliminate V_1
2. Replace operations

Tree-Formed Queries (TFQ)
A working example (2/4)

Eliminate $\leftrightarrow$: $\frac{f \leftrightarrow g}{(f \rightarrow g) \wedge (g \rightarrow f)}$

Eliminate $\rightarrow$: $\frac{f \rightarrow g}{\neg f \vee g}$

Move $\neg$ inwards: $\frac{\neg (f \wedge g)}{\neg f \vee \neg g}$

Move $\neg$ inwards: $\frac{\neg (f \vee g)}{\neg f \wedge \neg g}$

Eliminate double negation: $\frac{\neg \neg f}{f}$

Distribute $\vee$ over $\wedge$: $\frac{f \vee (g \wedge h)}{(f \vee g) \wedge (f \vee h)}$

- The query in the example is an existential first order query

$$q = V_? \exists V_1. (\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \vee \mathrm{Won}(V_1, \mathrm{Oscar})) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(\mathrm{V}_?, \mathrm{V}_1)$$

- Convert it to a Unions of Conjunctive Query (UCQ)

$$\begin{gathered} q = V_? \exists V_1. \big(\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(\mathrm{V}_?, \mathrm{V}_1)\big) \\ \vee (\mathrm{Won}(V_1, \mathrm{Oscar}) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(\mathrm{V}_?, \mathrm{V}_1)) \end{gathered}$$

$$q = \mathrm{Union}\begin{pmatrix} V_? \exists V_1. \big(\mathrm{Won}(V_1, \mathrm{GoldenGlobes}) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(\mathrm{V}_?, \mathrm{V}_1)\big), \\ V_? \exists V_1. \big(\mathrm{Won}(V_1, \mathrm{Oscar}) \wedge \neg \mathrm{BornIn}(V_?, \mathrm{America}) \wedge \mathrm{Direct}(\mathrm{V}_?, \mathrm{V}_1)\big) \end{pmatrix}$$

- For each conjunctive query
  - Convert the logical constraints to set operations.

Solutions for TFQs
A working example (3/4)

- An UCQ query

$$q = \text{SetUnion}\begin{pmatrix} V_?\exists\, V_1.\left(\text{Won}(V_1, \text{GoldenGlobes}) \wedge \neg\text{BornIn}(V_?, \text{America}) \wedge \text{Direct}(\text{V}_?, \text{V}_1)\right), \\ V_?\exists\, V_1.\left(\text{Won}(V_1, \text{Oscar}) \wedge \neg\text{BornIn}(V_?, \text{America}) \wedge \text{Direct}(\text{V}_?, \text{V}_1)\right) \end{pmatrix}$$

- For each conjunctive query (taking the first one as the example)
  - The atomic queries are Skolemized
    - $\text{Won}(V_1, \text{GoldenGlobes}) \xrightarrow{\text{Skolemization}} \widehat{V_1} = \text{WinnerOf}(\text{GoldenGlobes})$
    - $\text{Direct}(\text{V}_?, \text{V}_1) \xrightarrow{\text{Skolemization}} \widehat{V_?} = \text{DirectorOf}(V_1)$
    - $\neg\ \text{BornIn}(V_?, \text{America}) \xrightarrow{\text{Skolemization}} \widehat{V_?} = \neg\text{BornIn}(\text{America})$
  - Eliminate the existential variable V_1, the free variable $V_?$ should satisfies the two conditions
    - $\widehat{V_?} = \text{DirectorOf}\left(\text{WinnerOf}(\text{GoldenGlobes})\right)$
    - $\widehat{V_?} = \neg\text{BornIn}(\text{America})$
  - Reorganize them with the set operations
    - $V_? = \text{DirectorOf}\left(\text{WinnerOf}(\text{GoldenGlobes})\right) \cap \neg\text{BornIn}(\text{America}) = \text{DirectorOf}\left(\text{WinnerOf}(\text{GoldenGlobes})\right) - \text{BornIn}(\text{America})$
    - Note that the intersection + logical negated projection can be considered as set difference.

Solutions for TFQs
A working example (4/4)

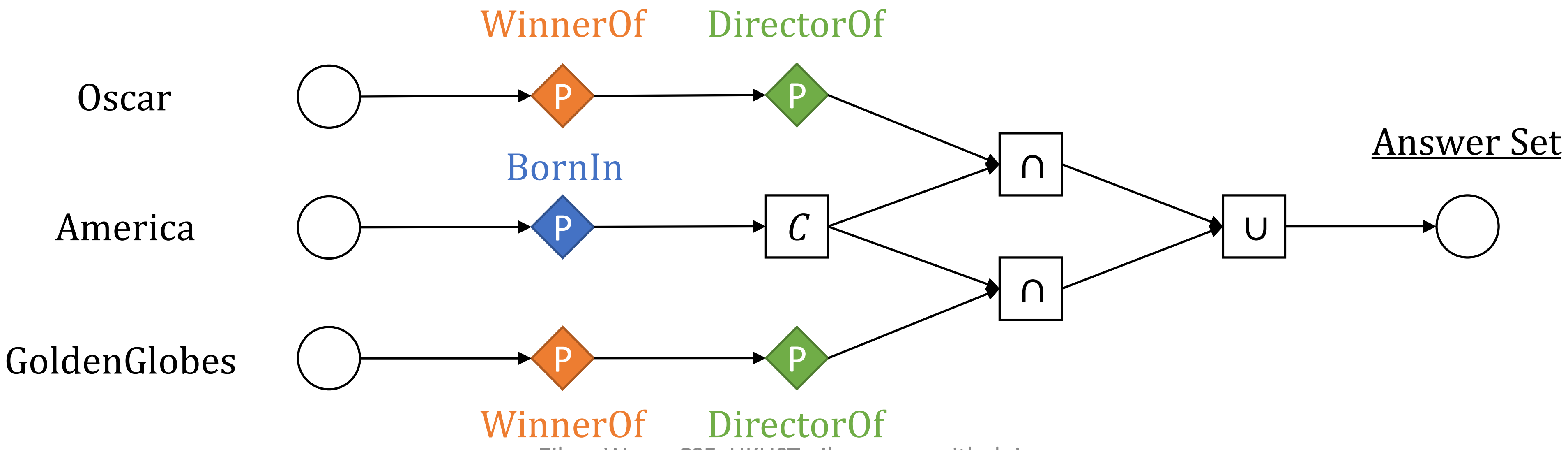
- For the first conjunctive query

$$V_? = \text{DirectorOf}\big(\text{WinnerOf}(\text{GoldenGlobes})\big) - \text{BornIn}(\text{America})$$

- Similarly for the other conjunctive query

$$V_? = \text{DirectorOf}\big(\text{WinnerOf}(\text{Oscar})\big) - \text{BornIn}(\text{America})$$

Then the computational tree



Quick Summarization

- What is a computational tree of a TFQ?
  - A tree whose <u>leaves are entities</u>, <u>intermediate nodes are set operators</u>, and <u>the root node is the answer set.</u>
  - The answer set can be computed by executing set operations following the bottom-up order.
- One query can be represented with multiple equivalent computational trees
  - For a logical query, there are many logical equivalent forms
  - Each logical form yields one or more computational trees
- Operators in trees can be different
  - Intersection & complement = intersection & difference
  - For UCQs, no need for “union” because we can collect answers from each CQ
- Can UCQs always be converted to TFQs?
  - It will be discussed in the next lecture

Solutions for TFQs
The design space of neural TFQ answering

Concept	Definition	Comment
Entity set	$\mathcal{E}$	The entity set in KG
Relation set	$\mathcal{R}$	The relation set in KG
Set embedding space	$\mathcal{X}$	Embedding space
Set embedding lookup	$E_{\mathcal{X}}: \mathcal{E} \mapsto \mathcal{X}$	Singleton set embedding
Entity embedding space	$\mathcal{Y}$	Embedding space
Entity embedding lookup	$E_{\mathcal{Y}}: \mathcal{E} \mapsto \mathcal{Y}$	Entity embedding
Set intersection	$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	Binary or N-ary
Set union	$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	Binary or N-ary
Set complement	$C: \mathcal{X} \mapsto \mathcal{X}$	Replaceable with set difference
Set projection	$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	One-hop link prediction
Scoring function	$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	How much an entity is in a set

Converting to computational tree makes it possible to model set operations with neural networks

Set Operators

set operations

$\cup$ set union

$\cap$ set intersection

C set complement

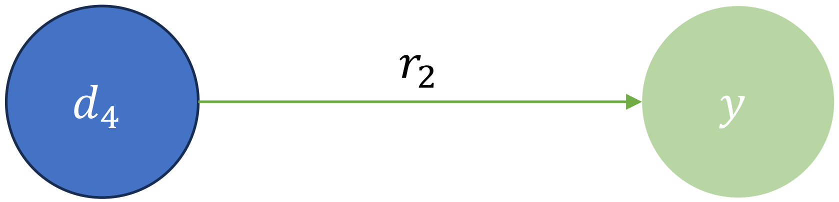
P set projection

Wang, Z., Yin, H., & Song, Y. (2021). Benchmarking the combinatorial generalizability of complex query answering on knowledge graphs. *arXiv preprint arXiv:2109.08925*.

Solutions for TFQs
Neural TFQ intuition explained

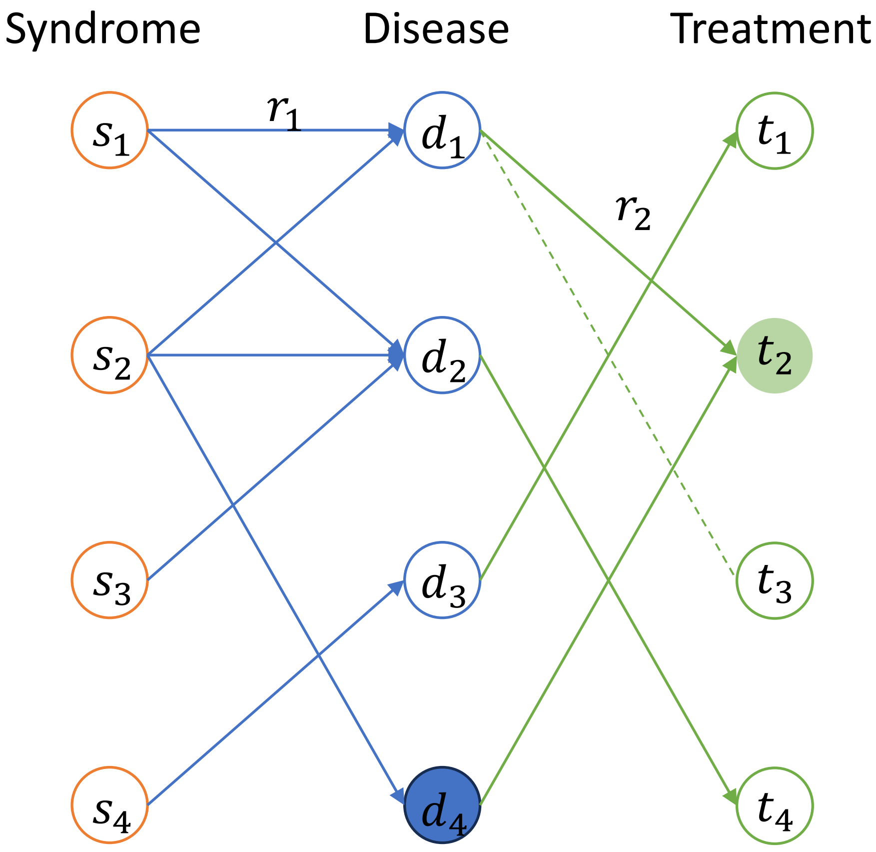
Logical query in natural language	Logical query in formal language
What is the treatment for disease "d_4"?	$y.r_2(d_4, y)$

One-hop Set Projection



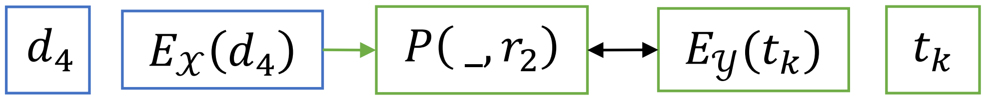
1. Computation of query embedding: $\hat{y} = P(E_{\mathcal{X}}(d_4), r_2)$
2. Rank entities by $s(\hat{y}, E_{\mathcal{Y}}(t_1)), s(\hat{y}, E_{\mathcal{Y}}(t_2))$,...

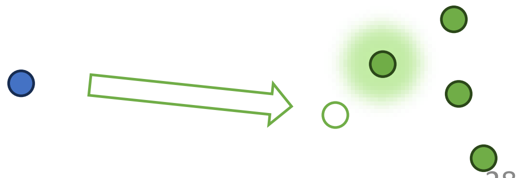
$E_{\mathcal{X}}(\cdot)$: find the embedding of an answer set
$E_{\mathcal{Y}}(\cdot)$: find the embedding of an entity



Traversal on the knowledge graph

The computational DAG



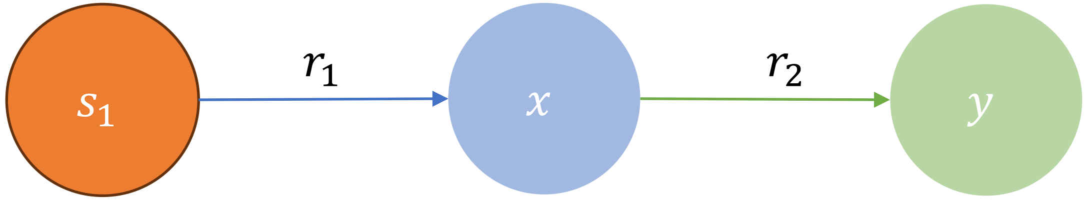


Neural projection in embedding space

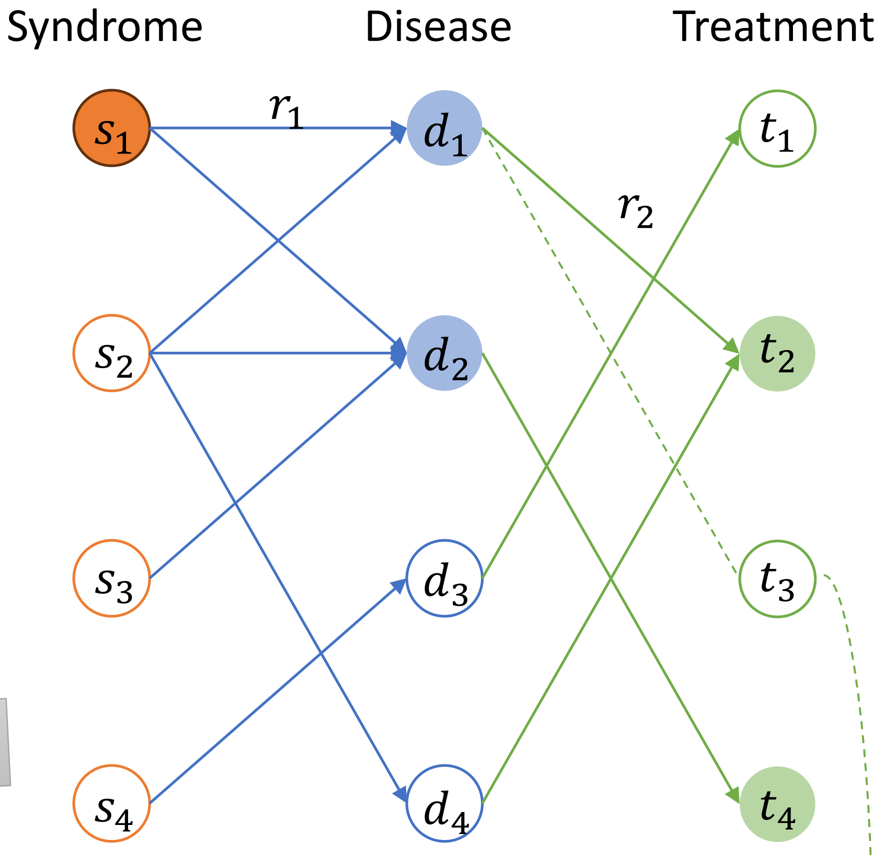
Solutions for TFQs
Neural TFQ intuition explained

Logical query in natural language	Logical query in formal language
What are the treatments for syndrome "s_1"?	$y.\exists x. r_1(s_1, x) \wedge r_2(x, y)$

Multi-hop Set Projection

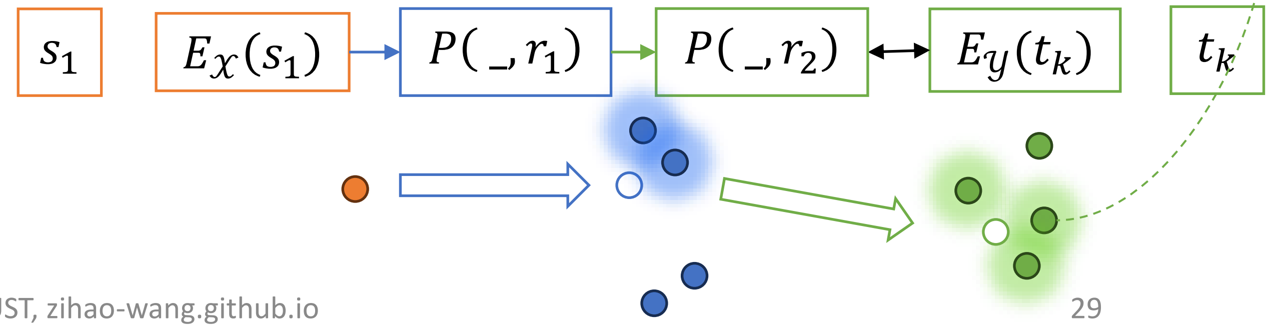


1. Computation of query embedding:
   $$\hat{y} = P(P(E_{\mathcal{X}}(s_1), r_1), r_2)$$
2. Rank entities by $s(\hat{y}, E_{\mathcal{Y}}(t_1)), s(\hat{y}, E_{\mathcal{Y}}(t_2)), \ldots$
3. The missing t_3 is recovered in embedding space.



Traversal on the knowledge graph

The computational DAG

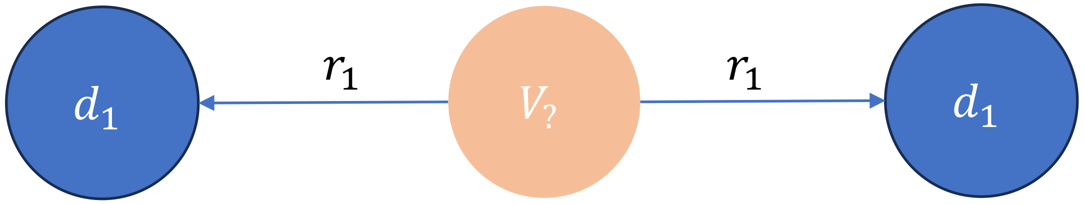


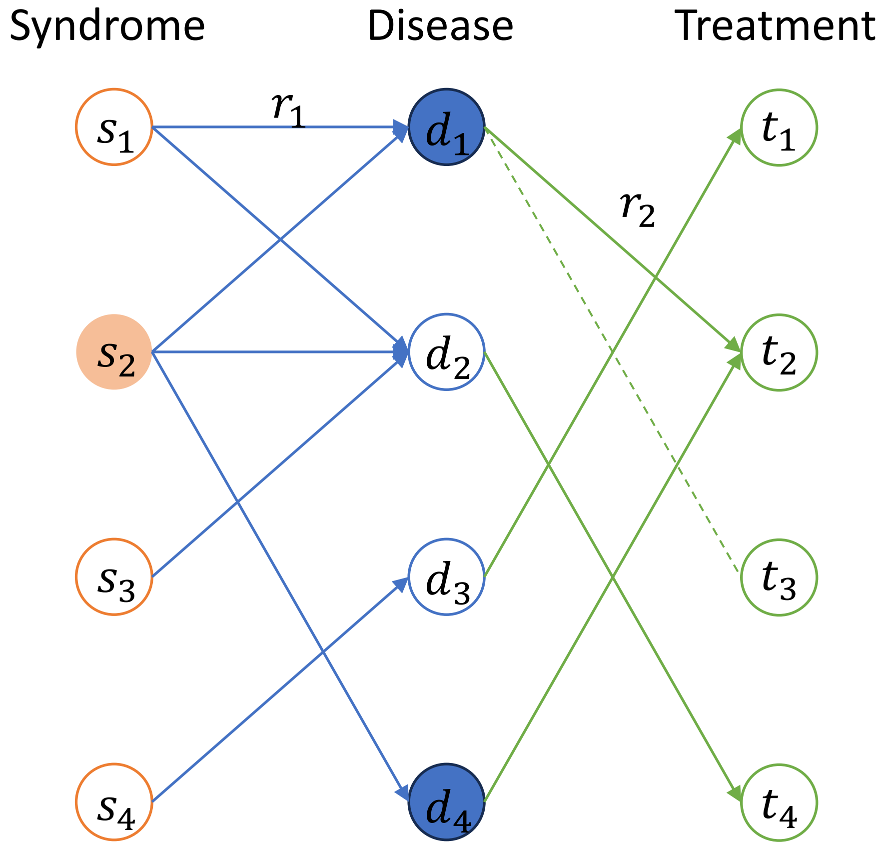
Multi-hop neural projection in embedding space

Solutions for TFQs
Neural TFQ intuition explained

Logical query in natural language	Logical query in formal language
What is the common syndrome of diseases "d_1" and "d_4"?	$V_?.\, r_1(V_?, d_1) \wedge r_1(V_?, d_4)$

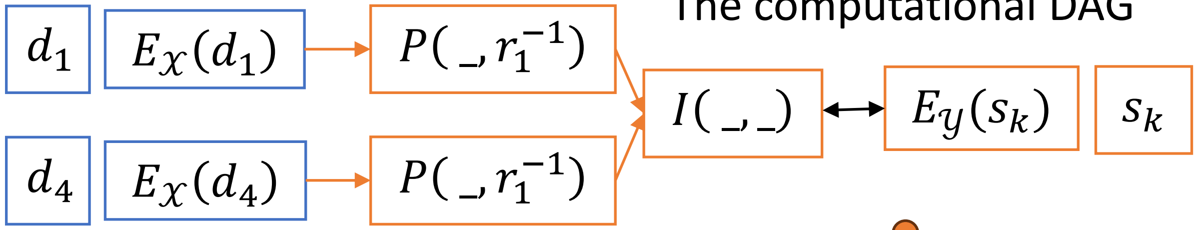
Set Intersection



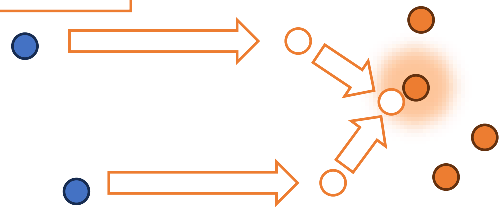


Traversal on the knowledge graph

The computational DAG



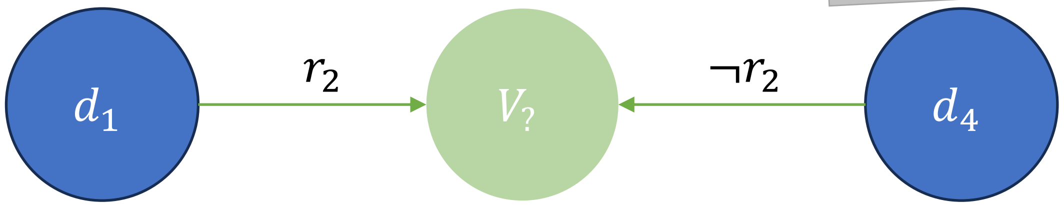
1. Computation of query embedding:
   $\hat{y} = I\left(P(E_\mathcal{X}(d_1), r_1^{-1}), P(E_\mathcal{X}(d_4), r_1^{-1})\right)$
2. Rank entities by $s\left(\hat{y}, E_\mathcal{Y}(s_1)\right), s\left(\hat{y}, E_\mathcal{Y}(s_2)\right)$,…

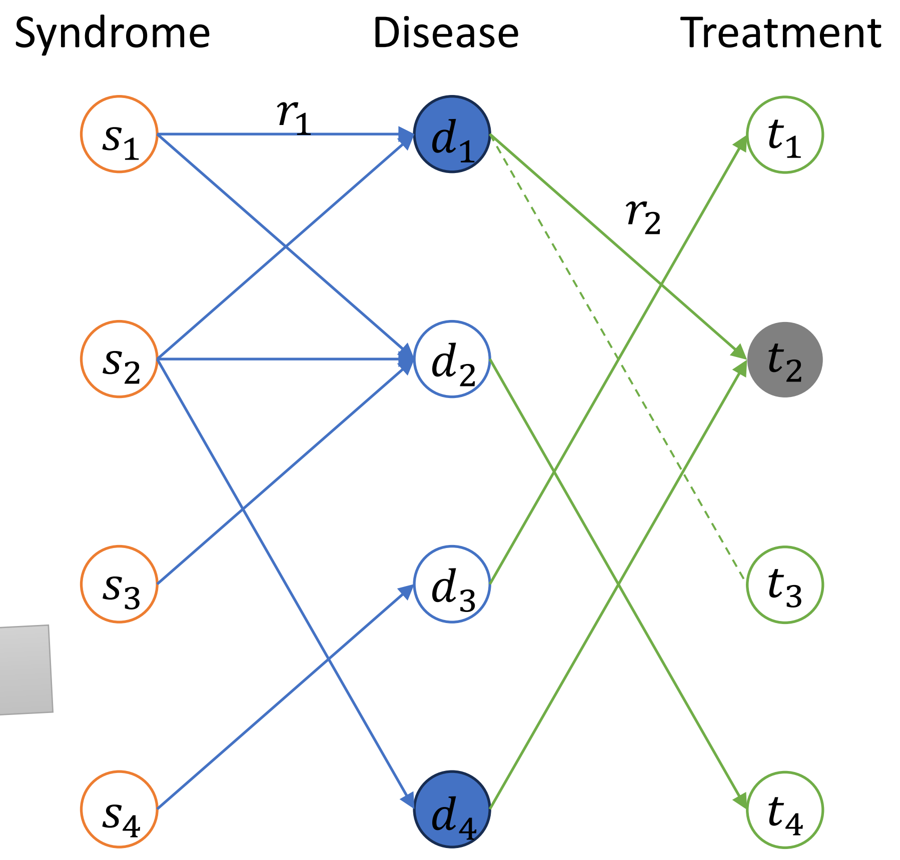


Solutions for TFQs
Neural TFQ intuition explained

Logical query in natural language	Logical query in formal language
What are the treatment for “d_1” but NOT “d_4”	$V_?. r_2(d_1, V_?) \wedge \neg r_2(d_4, V_?)$

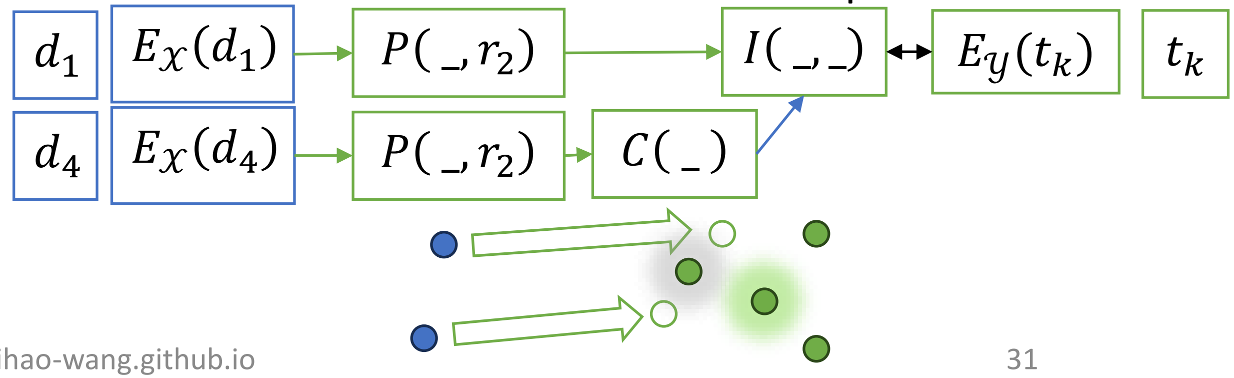
Set Intersection & Compliment





Traversal on the knowledge graph

The computational DAG

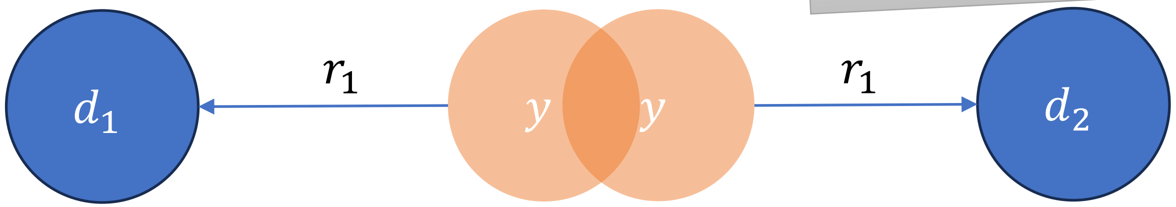


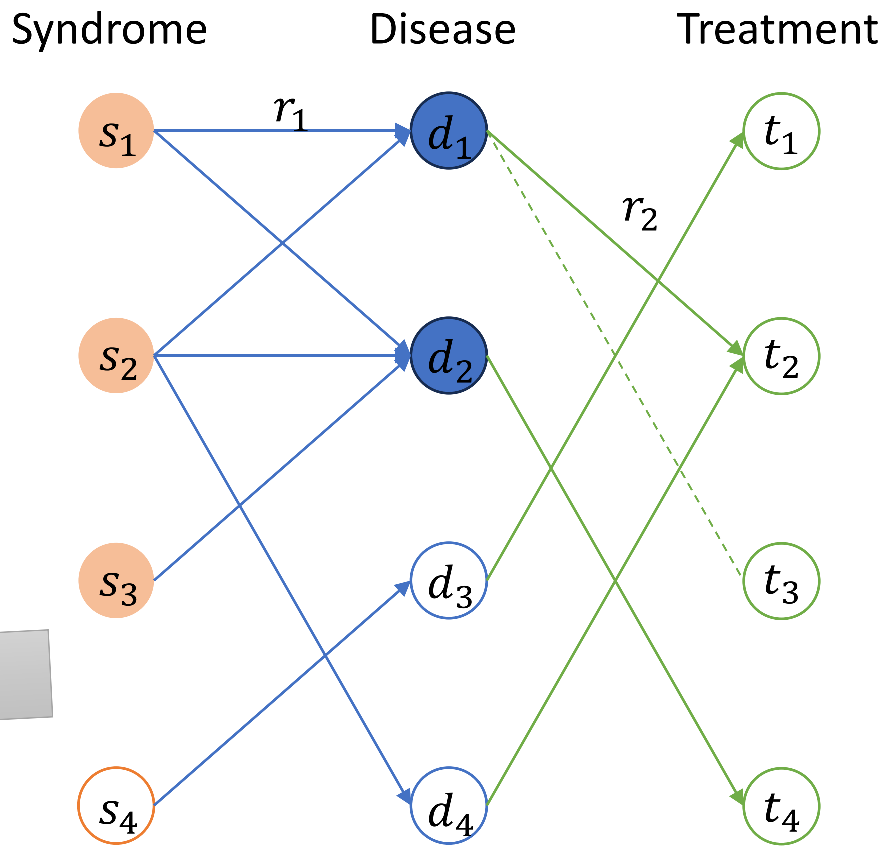
1. Computation of query embedding:
   $$\hat{y} = I\Big(P(E_\chi(d_1), \mathrm{r}_2), C\big(P(E_\chi(d_4), r_2)\big)\Big)$$
2. Rank entities by $s\big(\hat{y}, E_y(s_1)\big), s\big(\hat{y}, E_y(s_2)\big)$,…
3. The missing t_3 is recovered in embedding space.

Background:
Logical query examples

Logical query in natural language	Logical query in formal language
What are the syndromes for diseases "d_1" or "d_2"	$y.\, r_1(y, d_1) \vee r_1(y, d_2)$

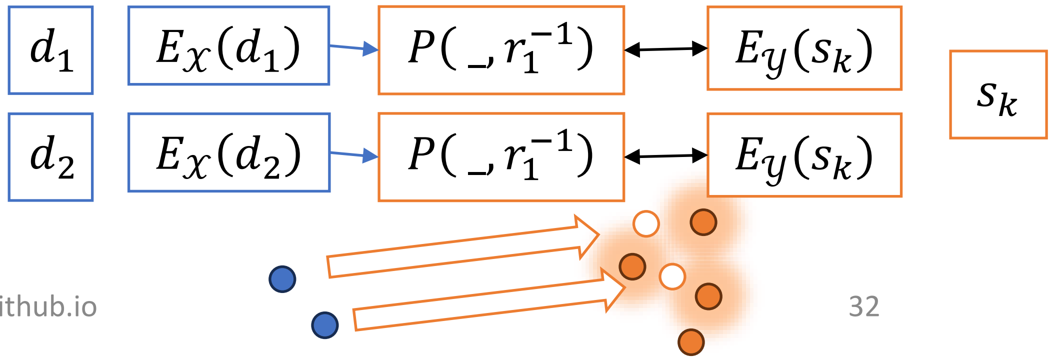
UCQ





Traversal on the knowledge graph

The computational DAG



1. Computation of query embedding of each CQ

   $$\widehat{y_1} = P(E_\chi(d_1), \mathrm{r}_1^{-1}), \widehat{y_2} = P(E_\chi(d_4), \mathrm{r}_1^{-1})$$

2. Compute scores separately

   $$s_{11} = s\left(\widehat{y_1}, E_y(s_1)\right), s_{21} = s\left(\widehat{y_2}, E_y(s_1)\right), \ldots$$

3. Final score is collected by $s_i = \max(s_{1i}, s_{2i})$
4. Rank entities by the final score s_i

Summary of the design intuitions

- Following the computational tree, forward passing of neural networks can simulate the graph traversal in the embedding space.
- UCQ provides a way to handle disjunction. But there are also other ways.
- The entire model is composed of several neutralized set operators and properly defined functions.
  - We will introduce several concrete designs later.
- The missing answers can be found because of the generalizability of neural models.
  - We will explain how to train models with their designs
  - But in general, it is negative sampling $L(q) = -\sum_{a \in A_q} s(q,a) + \sum_{i=1,\dots,k, e_i^- \notin A_q} s(q, e_i^-)$

Solutions for TFQs
A unified template for neural TFQ models

Concept	Definition	Comment
~~Entity set~~	$\cancel{\mathcal{E}}$	Known notation
~~Relation set~~	$\cancel{\mathcal{R}}$	Known notation
Set embedding space	$\mathcal{X}$	[Query Embedding: Slot 1]
~~Set embedding lookup~~	$\cancel{E_{\mathcal{X}}: \mathcal{E} \mapsto \mathcal{X}}$	Simplified
Entity embedding space	$\mathcal{Y}$	[Entity embedding: Slot 2]
~~Entity embedding lookup~~	$\cancel{E_{\mathcal{Y}}: \mathcal{E} \mapsto \mathcal{Y}}$	Simplified
Set intersection	$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	[Slot 3]
Set union	$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	[Slot 4]
Set complement	$C: \mathcal{X} \mapsto \mathcal{X}$	[Slot 5]
Set projection	$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	[Slot 6]
Scoring function	$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	[Slot 7]

Each method will be introduced by filling 7 slots

Solutions for TFQs
Vector embedding space: GQE

Definition	Comment
$\mathcal{X}$	$q \in \mathbb{R}^d$
$\mathcal{Y}$	$a \in \mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$I(q_1, \ldots, q_n) = W\Psi\big(\mathrm{MLP}(q_1), \ldots, \mathrm{MLP}(q_n)\big)$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	NA
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$P(q, r) = \mathrm{R_r} q$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q, a) = \frac{q^T a}{\lVert q \rVert \lVert a \rVert}$

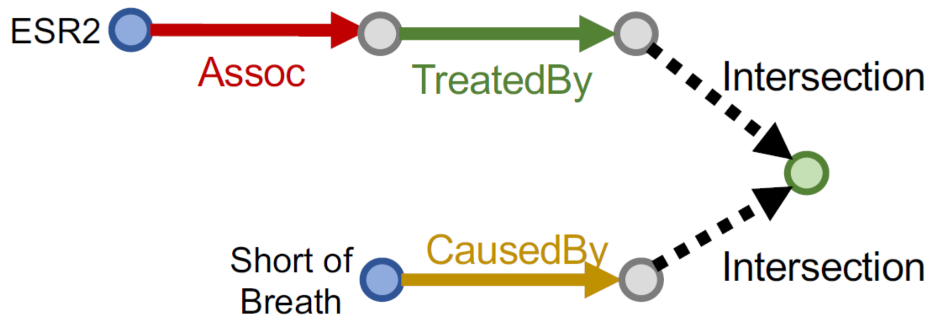
MLP: multi-layer perceptron
Ψ: a permutation invariant operator
W: a matrix
R_r: a matrix indexed by relation r

Training objective $L(q) = \max(0, 1 - s(q, a) + s(q, e_-))$

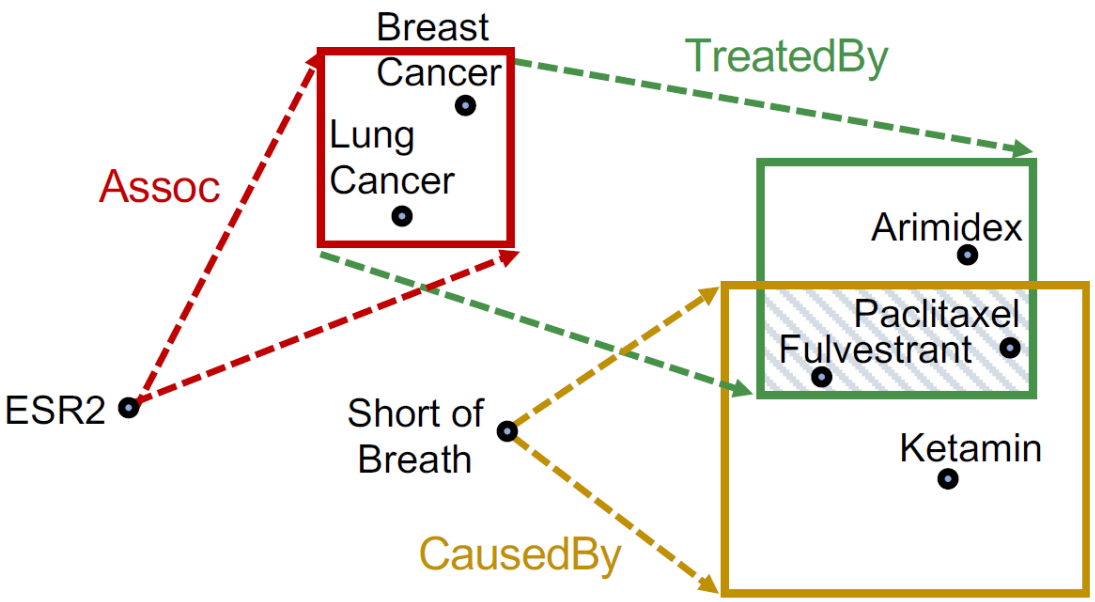
Hamilton, W., Bajaj, P., Zitnik, M., Jurafsky, D., & Leskovec, J. (2018). Embedding logical queries on knowledge graphs. *Advances in neural information processing systems*, *31*.

Solutions for TFQs
Geometric embedding space: Q2B (0/4)

Query Plan



Embedding Space



Slides credit: Jure Leskovec,

Solutions for TFQs
Geometric embedding space: Q2B (1/4)

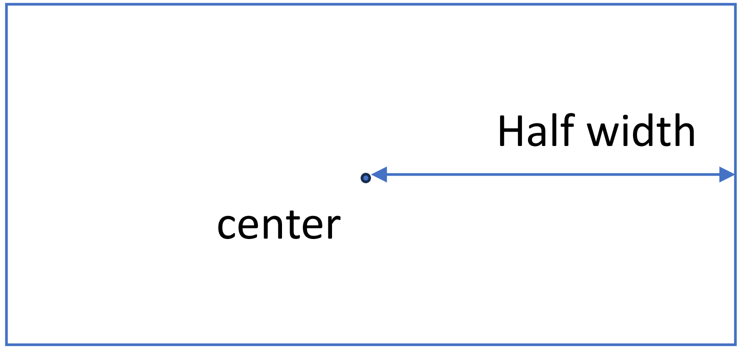
Definition	Comment
$\mathcal{X}$	q is a box in $\mathbb{R}^d$
$\mathcal{Y}$	$a \in \mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$q_I = I(q_1, \dots, q_i, \dots, q_n)$ $c^I = \sum a_i c^{q_i}, a_i = \mathrm{softmax}_{i=1,\dots,n}(\mathrm{MLP}(q_i))$ $w^I = \min\{w^{q_1}, \dots, w^{q_n}\}\sigma(\mathrm{Deepset}(q_1, \dots, q_n))$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	NA
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$\mathrm{Box}\left(c^{P(q,r)}, w^{P(q,r)}\right) = \mathrm{Box}(c_q + c_r, w_q + w_r)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q,a) = \gamma - \mathrm{dist}_{outside}(q,a) - \alpha \mathrm{dist}_{inside}(q,a)$

A box in $\mathbb{R}^d$ is parameterized by a vector

$$(c^q, w^q) \in \mathbb{R}^d \times \mathbb{R}^d_+$$

$$q = \mathrm{Box}(c^q,\ w^q) = \{v \in \mathbb{R}^d : c_i^q - w_i^q < v_i < c_i^q + w_i^q\}$$

- c the center of a box
- w the half width of a box



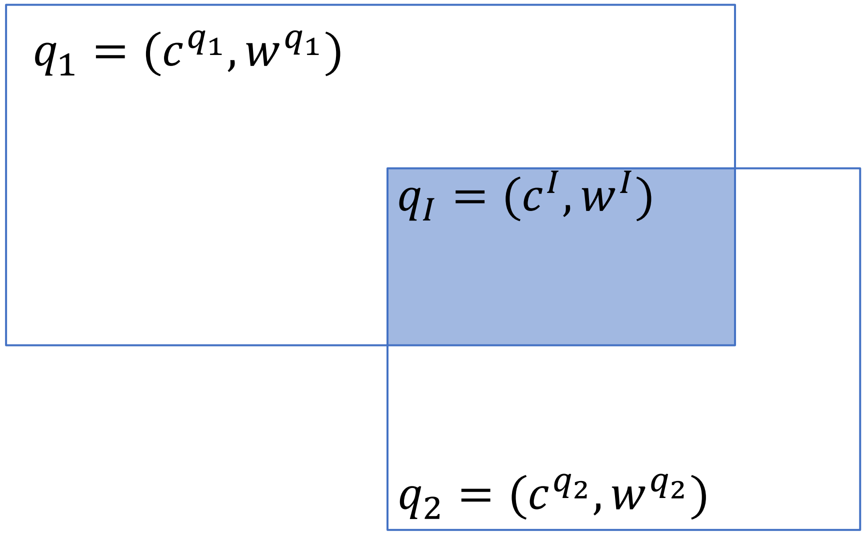
Training objective $L(q) = -\log\sigma\big(s(q,a)\big) - \sum_{j=1,\dots,k} \frac{1}{k} \log\sigma\left(s(q, e_j^-)\right)$

Ren, H., Hu, W., & Leskovec, J. (2020). Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969*.

Solutions for TFQs
Geometric embedding space: Q2B (2/4)

Definition	Comment
$\mathcal{X}$	q is a box in $\mathbb{R}^d$
$\mathcal{Y}$	$a \in \mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$q_I = I(q_1, \dots, q_i, \dots, q_n)$ $c^I = \sum a_i c^{q_i}, a_i = \mathrm{softmax}_{i=1,\dots,n}(\mathrm{MLP}(q_i))$ $w^I = \min\{w^{q_1}, \dots, w^{q_n}\}\sigma(\mathrm{Deepset}(q_1, \dots, q_n))$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	NA
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$\mathrm{Box}\left(c^{P(q,r)}, w^{P(q,r)}\right) = \mathrm{Box}(c_q + c_r, w_q + w_r)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q,a) = \gamma - \mathrm{dist}_{outside}(q,a) - \alpha \mathrm{dist}_{inside}(q,a)$

The intuition for the intersection of multiple boxes



Training objective $L(q) = -\log \sigma\left(s(q,a)\right) - \sum_{j=1,\dots,k} \frac{1}{k} \log \sigma\left(s\left(q, e_j^-\right)\right)$

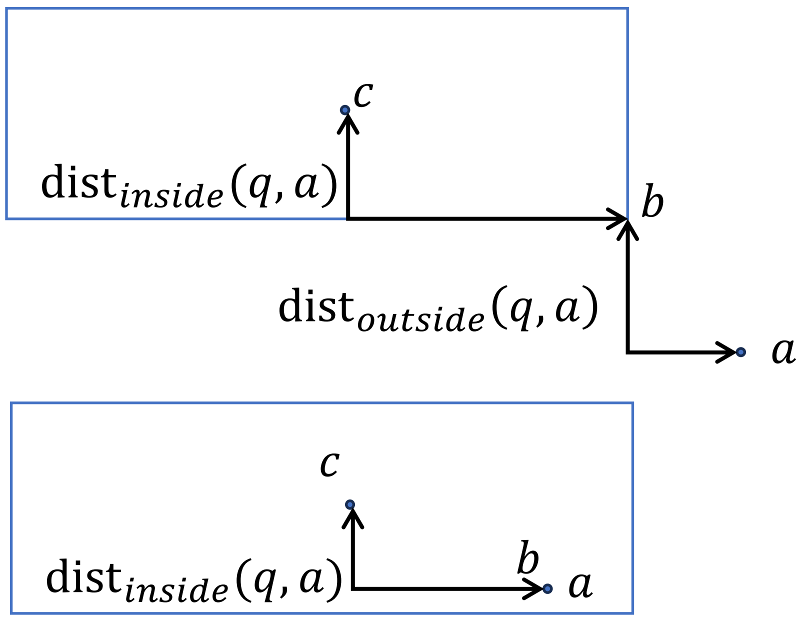
Ren, H., Hu, W., & Leskovec, J. (2020). Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969*.

Solutions for TFQs
Geometric embedding space: Q2B (3/4)

Definition	Comment
$\mathcal{X}$	q is a box in $\mathbb{R}^d$
$\mathcal{Y}$	$a \in \mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$q_I = I(q_1, \ldots, q_i, \ldots, q_n)$ $c^I = \sum a_i c^{q_i}, a_i = \mathrm{softmax}_{i=1,\ldots,n}(\mathrm{MLP}(q_i))$ $w^I = \min\{w^{q_1}, \ldots, w^{q_n}\}\sigma(\mathrm{Deepset}(q_1, \ldots, q_n))$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	NA
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$\mathrm{Box}\left(c^{P(q,r)}, w^{P(q,r)}\right) = \mathrm{Box}(c_q + c_r, w_q + w_r)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q,a) = \gamma - \mathrm{dist}_{outside}(q,a) - \alpha \mathrm{dist}_{inside}(q,a)$

Training objective $L(q) = -\log \sigma\big(s(q,a)\big) - \sum_{j=1,\ldots,k} \frac{1}{k} \log \sigma\left(s(q, e_j^-)\right)$

The intuition for the scoring function
Outside distance and inside distance



Ren, H., Hu, W., & Leskovec, J. (2020). Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969*.

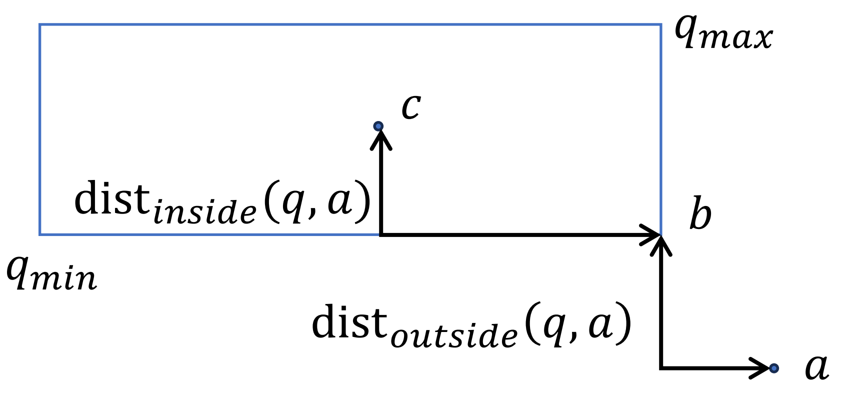
Solutions for TFQs
Geometric embedding space: Q2B (4/4)

Definition	Comment
$\mathcal{X}$	q is a box in $\mathbb{R}^d$
$\mathcal{Y}$	$a \in \mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$q_I = I(q_1, \dots, q_i, \dots, q_n)$ $c^I = \sum a_i c^{q_i}, a_i = \mathrm{softmax}_{i=1,\dots,n}(\mathrm{MLP}(q_i))$ $w^I = \min\{w^{q_1}, \dots, w^{q_n}\}\sigma(\mathrm{Deepset}(q_1, \dots, q_n))$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	NA
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$\mathrm{Box}\left(c^{P(q,r)}, w^{P(q,r)}\right) = \mathrm{Box}(c_q + c_r, w_q + w_r)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q,a) = \gamma - \mathrm{dist}_{outside}(q,a) - \alpha \mathrm{dist}_{inside}(q,a)$

The realization for the scoring function

Outside distance and inside distance:

$$q_{max,i} = c_i^q + w_i^q$$
$$q_{min,i} = c_i^q - w_i^q$$
$$dist_{outside}(a;q) = \|\max(a - q_{max}, 0) + \max(q_{min} - a, 0)\|_1$$
$$dist_{inside}(a;q) = \|c^q - \min(q_{max}, \max(q_{min}, a))\|_1$$



Training objective $L(q) = -\log\sigma\left(s(q,a)\right) - \sum_{j=1,\dots,k} \frac{1}{k} \log\sigma\left(s(q, e_j^-)\right)$

Ren, H., Hu, W., & Leskovec, J. (2020). Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969*.

Solutions for TFQs with set complement
Probability embedding space: BetaE

Q: How to model the set complement?

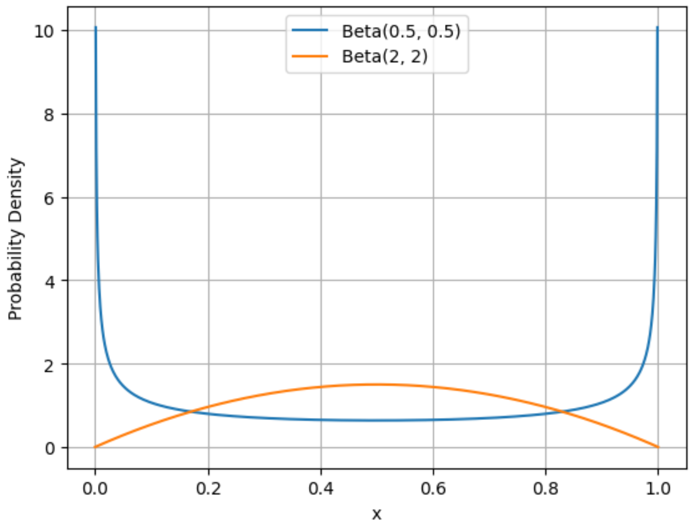
A: Use inductive bias of probability families

- P.d.f. of Beta distribution $\mathrm{Beta}(\alpha, \beta)$

$$f(x; \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$

where $\Gamma(x)$ is the Gamma function.

- Set embedding: d Beta distributions.



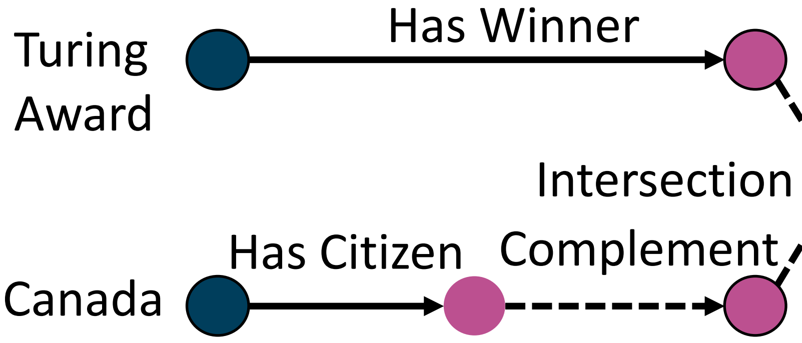
Definition	Comment
$\mathcal{X}$	$q = (\alpha_1^q, \beta_1^q, \dots, \alpha_d^q, \beta_d^q) \in [0, \infty)^{2d}$
$\mathcal{Y}$	$a \in [0, \infty)^{2d}$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$I(q_1, \dots, q_i, \dots, q_n) = \sum_i \alpha_i q_i$, $\alpha_i = \mathrm{softmax}(\mathrm{NN}(q_i))$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	UCQ
$C: \mathcal{X} \mapsto \mathcal{X}$	$\boldsymbol{C(q) = 1/q}$
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$P(q, r) = \mathrm{MLP}_r(q)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q, a) = \gamma - \sum_{i=1,\dots,d} \mathrm{KL}\left(B(\alpha_i^q, \beta_i^q) \vert B(\alpha_i^a, \beta_i^a)\right)$

Training objective $L(q) = -\log \sigma\big(s(q, a)\big) - \sum_{j=1,\dots,k} \frac{1}{k} \log \sigma\left(s(q, e_j^-)\right)$

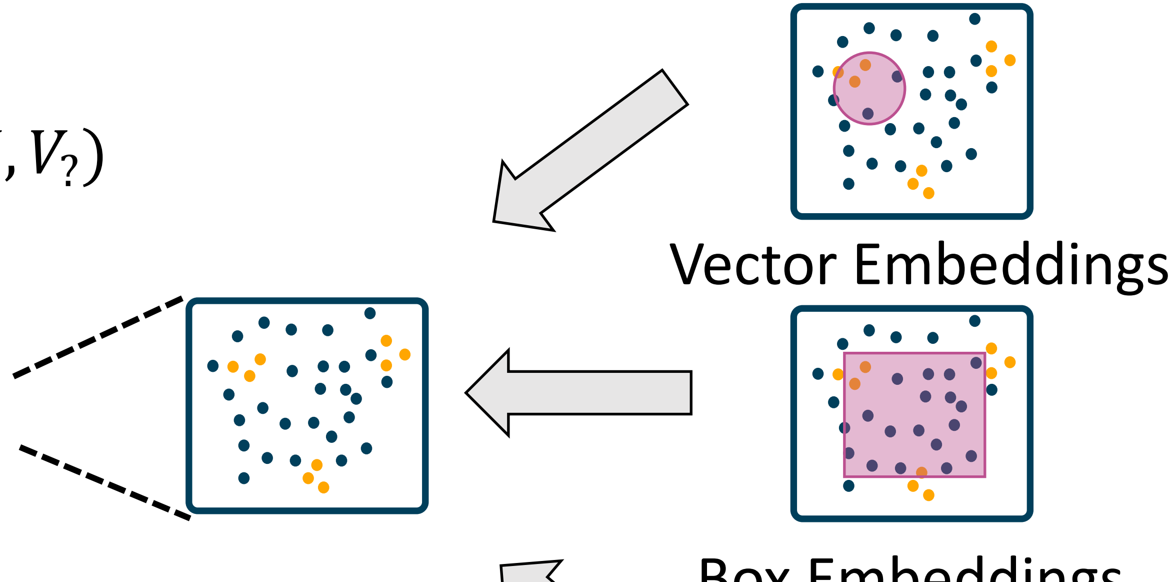
Ren, H., & Leskovec, J. (2020). Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems*, *33*, 19716-19726.

Solutions for TFQs with set union
Empirical sample embedding space: Q2P

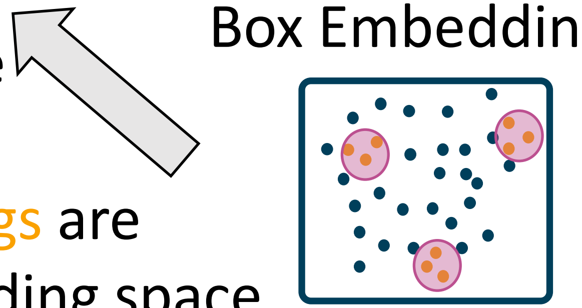
$$q = V_? \,.\, \exists V : Win(TuringAward, V) \wedge \neg Citizen(Canada, V) \wedge Graduate(V, V_?)$$



Computation Graph



Embedding Space



The multi-hop logical operations make the query answers diversified

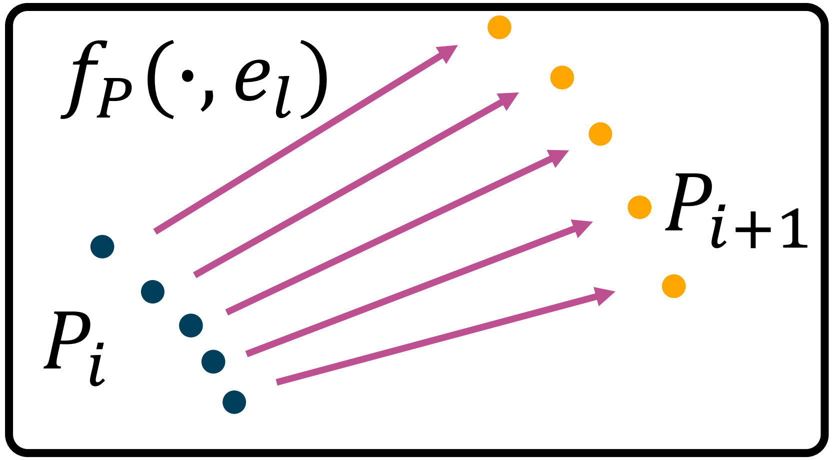
The answers embeddings are scattered in the embedding space

$$P_i = [p_i^{(1)}, p_i^{(2)}, p_i^{(3)}, \dots, p_i^{(K)}]$$

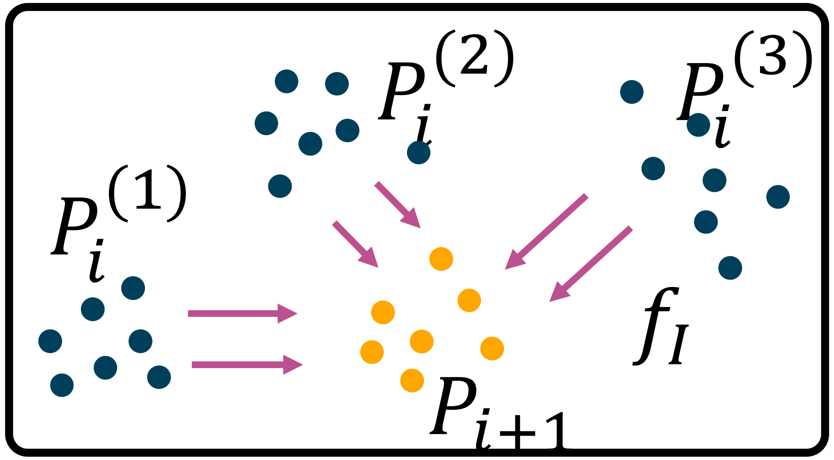
Bai, J., Wang, Z., Zhang, H., & Song, Y. (2022). Query2Particles: Knowledge graph reasoning with particle embeddings. *arXiv preprint arXiv:2204.12847*.

Solutions for TFQs with set union

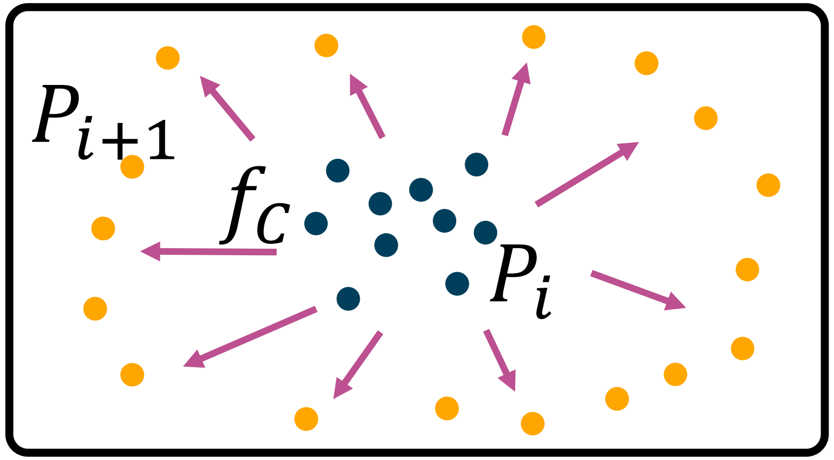
Empirical sample embedding space: Q2P



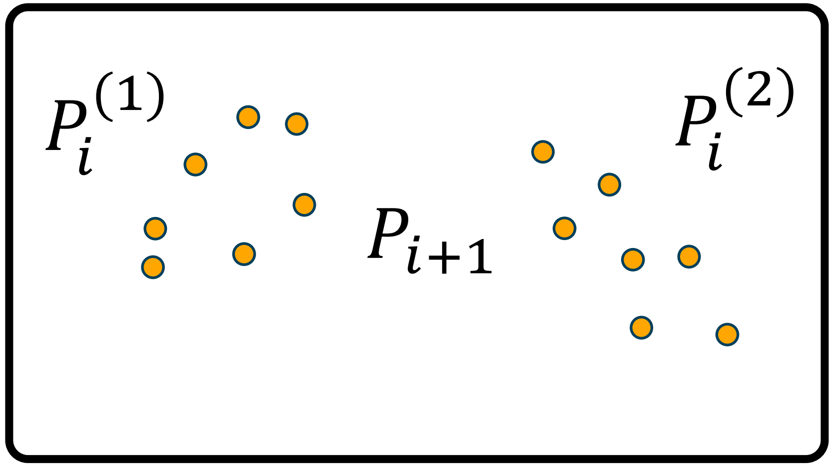
Relational Projection



Intersection



Complement



Union

Bai, J., Wang, Z., Zhang, H., & Song, Y. (2022). Query2Particles: Knowledge graph reasoning with particle embeddings. *arXiv preprint arXiv:2204.12847*.

Solutions for TFQs with set union
Empirical sample embedding space: Q2P

Gated Transition for customizing the directions of transitions for each vector in particles:

$A_i = (1 - Z) \odot P_i + Z \odot T$

Here Z is the update gate, and T is transition for each particles. They are computed from P_i and the relation embedding e_l for relation l

Definition	Comment
$\mathcal{X}$	Multiple particles in $\mathbb{R}^d$
$\mathcal{Y}$	$\mathbb{R}^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$A_i = \texttt{self-attn}(P_i)$ $P_{i+1} = \texttt{MLP}(A_i)$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	Merge Particles
$C: \mathcal{X} \mapsto \mathcal{X}$	$A_i = \texttt{self-attn}(P_i)$ $P_{i+1} = \texttt{MLP}(A_i)$
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$A_i = (1 - Z) \odot P_i + Z \odot T$ $P(q,r) = \texttt{self-attn}(A_i)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q,a) = \max_{k=1,2,3,\ldots,K} < p_T^{(k)}, a >$

Training objective $L(q) = -\log \frac{e^{s(q,a)}}{\sum_{v \in \mathcal{E}} e^{s(q,v)}}$

Bai, J., Wang, Z., Zhang, H., & Song, Y. (2022). Query2Particles: Knowledge graph reasoning with particle embeddings. *arXiv preprint arXiv:2204.12847*.

Solutions for TFQs with fuzzy logic
Fuzzy logic embedding space (1/4)

Designing set operators is tricky. Can we make it more theoretical?

Trying to incorporate it with fuzzy logic t-norm.

- A t-norm is a function T: [0, 1] × [0, 1] → [0, 1]
- Consider conjunction query $(q_1 \wedge q_2)(y; x_1, \ldots)$
  - Consider the fuzzy Truth Value $TV[(q_1 \wedge q_2)(y = a)]$
  $$TV[(q_1 \wedge q_2)(y = a)] = TV[q_1(y = a) \wedge q_2(y = a)]$$
  - Introduce t-norm ⊤, then,
  $$TV[q_1(y = a) \wedge q_2(y = a)] = TV[q_1(y = a)] \top TV[q_2(y = a)]$$
- Also, for disjunction and negation,
  - $TV[q_1(y = a) \vee q_2(y = a)] = TV[q_1(y = a)] \perp TV[q_2(y = a)]$
  - $TV[\neg q_1(y = a)] = 1 - TV[q_1(y = a)]$

We consider Godel t-norm in this lecture
- $a \top b = \min(a, b)$
- $a \perp b = \max(a, b)$

Solutions for TFQs with fuzzy logic
Fuzzy logic embedding space (2/4)

- A matrix $M_{q,a} = TV\big(q(y = a)\big)$ records everything we need.
  where q is an arbitrary query and a is an entity.
  $$s(q, a) = M_{q,a}$$
- The range of q looks infinitely large, but what really matters is
  - Finite positive atomic queries, so that only finite rows should be recorded M^{atomic}.
  - t-norm computation (introduced in previous page) generates infinite rows.
- We can of course consider the low rank decomposition of M^{atomic}.
  $$M^{atomic}_{q,a} \approx \vec{q}^T \vec{a}$$
- $\vec{q}^T \in \mathbb{R}^d$ is the query embedding of an atomic query $q(y) = r(e, y)$.
- Any atomic query can be written as $q(y) = r(e, y)$ by allowing reverse relation.

Solutions for TFQs with fuzzy logic
Fuzzy logic embedding space (3/4)

- Low rank (rank d) decomposition of M^{atomic}

$$M_{q,a}^{atomic} \approx \vec{q}^T \vec{a}$$

  - $\vec{q}$ is the query embedding
  - $\vec{a}$ is the entity embedding
- If we further assume that
  - $\vec{q} \in [0,1]^d$,$\vec{a} \in [0,1]^d$
  - t-norm is linear, for convenience we consider Godel t-norm
- Let q_1, q_2 be atomic query, then

$$M_{q_1 \wedge q_2,a} = \overrightarrow{q_1 \wedge q_2}^T \vec{a} = M_{q_1,a} \top M_{q_2,a} = \min\left(M_{q_1,a}, M_{q_2,a}\right) = \min\left(\overrightarrow{q_1}^T \vec{a}, \overrightarrow{q_2}^T \vec{a}\right) = \min(\overrightarrow{q_1}^T, \overrightarrow{q_2}^T)\vec{a}$$

- Conclusion:

$$\overrightarrow{q_1 \wedge q_2} = \min(\overrightarrow{q_1}, \overrightarrow{q_2}) = \overrightarrow{q_1} \top \overrightarrow{q_2}$$

Solutions for TFQs with fuzzy logic Fuzzy logic embedding space (4/4)

- We omit $\vec{q}$ as q.

$$q_1 \wedge q_2 = q_1 \top q_2$$

- Similarly

$$q_1 \vee q_2 = q_1 \perp q_2$$
$$\neg q = 1 - q$$

- Then we parameterize atomic query $q_{atomic} = r(q_{in}, y)$ as $\mathrm{MLP}_r(q_{in})$, we see the projection is

$$P(q_{in}, r) = \mathrm{MLP}_r(q_{in})$$

- *Then we get the FuzzQE*

Definition	Comment
$\mathcal{X}$	$q \in [0,1]^d$
$\mathcal{Y}$	$a \in [0,1]^d$
$I: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$I(q_1, \dots, q_i, \dots, q_n) = q_1 \top \dots \top q_n$
$U: \mathcal{X} \times \cdots \times \mathcal{X} \mapsto \mathcal{X}$	$U(q_1, \dots, q_i, \dots, q_n) = q_1 \perp \cdots \perp q_n$
$C: \mathcal{X} \mapsto \mathcal{X}$	$C(q) = 1 - q$
$P: \mathcal{X} \times \mathcal{R} \mapsto \mathcal{X}$	$P(q, r) = \mathrm{MLP}_r(q)$
$s: \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$	$s(q, a) = q^T a$

Chen, X., Hu, Z., & Sun, Y. (2022, June). Fuzzy logic based logical query answering on knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 3939-3948).

Summary so far

- An overview of the problem + a detailed discussion about TFQ
  - Definition
  - Set operator parameterization
    - In vector, geometric region, probability distribution, empirical samples, …
    - Heuristics for projection, intersection, negation, and union.
    - Fuzzy logic motivated methods.
- Questions
  - How far is TFQ away from EFO1?
  - Methods beyond simulating set operators?

Tree-form queries and existential first order queries

1. The syntactical definition of tree-form queries
2. Relation between TFQ and EFO

TFQ vs EFO: The syntax of TFQ (1/5)

Tree-form query family contains the queries that can be converted into the computational tree.

To formal define TFQ, we should describe logical queries that expresses

- Atomic query
- Set projection
- Set intersection
- Set union
- Set complement

TFQ vs EFO: The syntax of TFQ (2/5)

To formal define TFQ, we should describe logical queries that expresses

- ➢Atomic query
- Set projection
- Set complement
- Set intersection
- Set union

Let $\mathcal{T}$ be the set of all TFQs, then

➢Set of all atomic queries.

$S_{atomic} = \{q(y) = r(h, y)$: $r \in \mathcal{R}, h \in \mathcal{E}\} \in \mathcal{T}$

Note: let $r \in \mathcal{R}$ be a relation and r^{-1} be its inverse, then $r^{-1} \in \mathcal{R}$.

TFQ vs EFO: The syntax of TFQ (3/5)

To formal define TFQ, we should describe logical queries that expresses

- Atomic query
- ➢Set projection
- Set complement
- Set intersection
- Set union

Let $\mathcal{T}$ be the set of all TFQs, then

- $S_{atomic} \in \mathcal{T}$

➢If $\phi(z) \in \mathcal{T}$, then

$$\exists z. \phi(z) \wedge r(z, y) \in \mathcal{T}$$

$\phi(z)$ is a TFQ, $TV[\phi(z = a)]$ describes the probability of a being the answer of $\phi(z)$

TFQ vs EFO: The syntax of TFQ (4/5)

To formal define TFQ, we should describe logical queries that expresses

- Atomic query
- Set projection
- ➢Set complement
- Set intersection
- Set union

Let $\mathcal{T}$ be the set of all TFQs, then

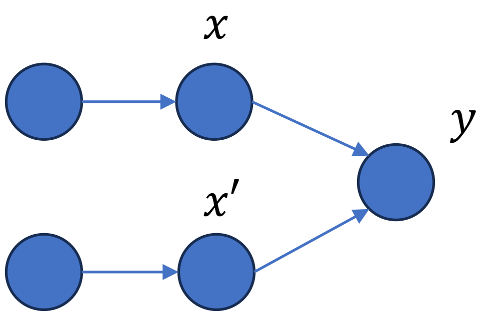
- $S_{atomic} \in \mathcal{T}$
- If $\phi(z) \in \mathcal{T}$, then
  $\exists z. \phi(z) \land r(z, y) \in \mathcal{T}$

➢If $\phi \in \mathcal{T}$, then
$\neg\phi(y) \in \mathcal{T}$

TFQ vs EFO: The syntax of TFQ (5/5)

To formal define TFQ, we should describe logical queries that expresses

- Atomic query
- Set projection
- Set complement
- ➢ Set intersection
- ➢ Set union



Let $\mathcal{T}$ be the set of all TFQs, then

- $S_{atomic} \in \mathcal{T}$
- If $\phi(z) \in \mathcal{T}$, then
$$\exists z. \phi(z) \wedge r(z, y) \in \mathcal{T}$$
- If $\phi \in \mathcal{T}$, then
$$\neg\phi(y) \in \mathcal{T}$$
- ➢ If $\phi, \psi \in \mathcal{T}$, then
$$\phi(y) \wedge^* \psi(y) \in \mathcal{T}$$
$$\phi(y) \vee^* \psi(y) \in \mathcal{T}$$

Note *: the existential variables in $\phi(y)$ and $\psi(y)$ are assumed to be not shared.

TFQ vs EFO: The syntax of TFQ (5/5)

Tree-form query

Let $\mathcal{T}$ be the set of all TFQs, then

- $S_{atomic} = \{r(h,y): r \in \mathcal{R}, h \in \mathcal{E}\} \in \mathcal{T}$

  h is an entity, y is a variable
- If $\phi \in \mathcal{T}$, then
$$\neg\phi(y) \in \mathcal{T}$$
- If $\phi, \psi \in \mathcal{T}$, then
$$\phi(y) \wedge^* \psi(y) \in \mathcal{T}$$
$$\phi(y) \vee^* \psi(y) \in \mathcal{T}$$

Note*: the existential variables in $\phi(y)$ and $\psi(y)$ are assumed to be not shared.

- If $\phi(z) \in \mathcal{T}$, then
$$\exists z. \phi(z) \wedge r(z,y) \in \mathcal{T}$$

Existential First Order (EFO) query

Let $\mathcal{Q}$ be the set of all EFO query, then

- $F_{atomic} = \{r(t_1,t_2), r \in \mathcal{R}\} \in \mathcal{Q}$

  t_1 and t_2 are either entities or variables.
- If $\phi \in F_{atomic}$, then
$$\neg\phi(y) \in \mathcal{Q}$$
- If $\phi, \psi \in \mathcal{Q}$, then
$$\phi(y) \wedge \psi(y) \in \mathcal{Q}$$
$$\phi(y) \vee \psi(y) \in \mathcal{Q}$$

Note1: the existential variables in $\phi(y)$ and $\psi(y)$ can be shared.

- If $\phi \in \mathcal{Q}$ and x is a variable, then
$$\exists x. \phi \in \mathcal{Q}$$

Note2: we can always use this rule to make sure there is only one free variable (non-quantified). This subset is also known as EFO_1.

$$CQ_j(y; x_1, \dots, x_n) = y. \exists x_1, \dots, \exists x_n. \bigwedge_{k=1,\dots,M_j} a_{jk}$$

TFQ vs EFO: Definitions reveals more questions

- Are TFQ the same as EFO?
  - If not, which one is a larger set?
  - If not, are methods introduced in the previous lecture still effective for EFO?
    - If not, we need more methods!

TFQ vs EFO: Are TFQ $\mathcal{T}$ the same as EFO $\mathcal{Q}$?

$\mathcal{T} - \mathcal{Q}$ is not empty

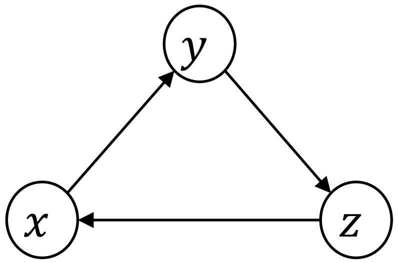
Construction:

- $r_1(a, y) \in \mathcal{T}$
- $\exists x. r_1(a, x) \wedge r_2(x, y) \in \mathcal{T}$
- $\neg \exists x. r_1(a, x) \wedge r_2(x, y) \in \mathcal{T}$
- $\phi(y) = \forall x. \neg r_1(a, x) \vee \neg r_2(x, y)$
  $\phi(y) \in \mathcal{T}$
- There is a universal quantifier, so it is not existential.

$\mathcal{Q} - \mathcal{T}$ is not empty

Construction:

- $r(x, y), r(y, z), r(z, x) \in \mathcal{Q}$
- $\phi(y) = \exists x, z. r(x, y) \wedge r(y, z) \wedge r(z, x)$
  $\phi(y) \in Q$
- This is a triangle but not a tree.



TFQ vs EFO

- Fine grained characterization of the differences, refer to the paper

Yin, H., Wang, Z., & Song, Y. (2023). Rethinking Complex Queries on Knowledge Graphs with Neural Link Predictors. arXiv preprint arXiv:2304.07063.

https://arxiv.org/abs/2304.07063

- We need methods designed for the generic EFO query?

Neuro-symbolic methods for EFO queries

Inference

Search in the continuous space

Learning to search in the continuous space

Methods: Fuzzy inference with truth values

Key idea

- Calculate the truth value rigorously defined with fuzzy logic.

Problem definition

- Given a link predictor $f(h, r, t)$, we can compute the entire
$$M_{q,a} = TV[q(y = a)]$$

Comment

- Most straightforward way of computation, will be simplified later.

Methods: Fuzzy inference with truth values

Let $\mathcal{Q}$ be the set of all EFO query, then

- $F_{atomic} = \{r(t_1, t_2), r \in \mathcal{R}\} \in \mathcal{Q}$
  t_1 and t_2 are either entities or variables.
- If $\phi \in F_{atomic}$, then
  $$\neg\phi(y) \in \mathcal{Q}$$
- If $\phi, \psi \in \mathcal{Q}$, then
  $$\phi(y) \wedge \psi(y) \in \mathcal{Q}$$
  $$\phi(y) \vee \psi(y) \in \mathcal{Q}$$
- If $\phi \in \mathcal{Q}$ and x is a variable, then
  $$\exists x. \phi \in \mathcal{Q}$$

Some rules to calculate the truth value.

- If $\phi \in F_{atomic}$, then
  $$TV[r(a,b)] = f(a,r,b)$$
- If $\phi \in F_{atomic}$, then
  $$TV[\neg\phi] = 1 - TV[\phi]$$
- If $\phi, \psi \in \mathcal{Q}$, then
  $$TV[\phi \wedge \psi] = TV[\phi] \top TV[\psi]$$
  $$TV[\phi \vee \psi] = TV[\phi] \perp TV[\psi]$$
- If $\phi \in \mathcal{Q}$ and x is a variable, then
  $$TV[\exists x. \phi(x)] = \perp^*_{a \in \mathcal{E}} TV[\phi(x = a)]$$

Yin, H., Wang, Z., & Song, Y. (2023). Rethinking Complex Queries on Knowledge Graphs with Neural Link Predictors. arXiv preprint arXiv:2304.07063.

Methods: Fuzzy inference with truth values

Some rules to calculate the truth value.

- If $\phi \in F_{atomic}$, then
$$TV[r(a,b)] = f(a,r,b)$$
- If $\phi \in F_{atomic}$, then
$$TV[\neg\phi] = 1 - TV[\phi]$$
- If $\phi, \psi \in \mathcal{Q}$, then
$$TV[\phi \wedge \psi] = TV[\phi] \top TV[\psi]$$
$$TV[\phi \vee \psi] = TV[\phi] \perp TV[\psi]$$
- If $\phi \in \mathcal{Q}$ and x is a variable, then
$$TV[\exists x. \phi(x)] = \perp^*_{a\in\mathcal{E}} TV[\phi(x = a)]$$

- $\wedge$ and $\vee$ are logical conjunction and disjunction, they are related to a paired fuzzy logic
  - t-norm $\top$ and
  - t-conorm $\perp$.
- $\perp^*_{a\in\mathcal{E}}$ is also a fuzzy logic t-conorm, but it is used for the existential variable.
- Also related to the *lifted inference* in probabilistic database.

Methods: Fuzzy <u>inference</u> with truth values

$$\phi(y; x_1, \ldots, x_n) = \exists x_1, \ldots, \exists x_n. \bigvee_{j=1,\ldots,N} \bigwedge_{k=1,\ldots,M_j} a_{jk}$$

- A question, why $\phi(y; x_1, \ldots, x_n)$ follows our inductive definition before?
  - Because the existential quantifier and conjunction/disjunction are exchangeable.
- The truth value of $\phi(y = a)$ is eventually

$$TV[\phi(y = a)] = \perp^*_{x_1 = e_1 \in \mathcal{E}} \ldots \perp^*_{x_n = e_n \in \mathcal{E}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk} \Big|_{y=a} \right]$$

Methods: Search problem, derivation and formulation

- When $\perp^*$ is a Godel t-conorm, the inference problem

$$TV[\phi(y=a)] = \perp^*_{x_1=e_1\in\mathcal{E}} \cdots \perp^*_{x_n=e_n\in\mathcal{E}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk}\Big|_{y=a}\right]$$

becomes an optimization problem

$$TV[\phi(y=a)] = \max_{x_1,\ldots,x_n\in\mathcal{E}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk}\Big|_{y=a}\right]$$

This complexity of this search problem <u>in general grows exponentially</u> with respect to the number of variables.

Methods: Search problem, a complex example

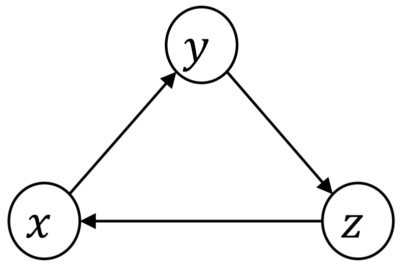
An optimization problem

$$TV[\phi(y = a)] = \max_{x_1,\ldots,x_n \in \mathcal{E}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk} \Big|_{y=a} \right]$$

For the case $\phi(y = a) = \exists x, z. r(x, y = a) \wedge r(y = a, z) \wedge r(z, x)$, we shall optimize

$$TV[\phi(y = a)] = \max_{x \in \mathcal{E}, z \in \mathcal{E}} f(x, r, a) \top f(a, r, z) \top f(z, r, x)$$

- Because x and z are dependent.



Methods: Search problem, simpler case

- For $\phi \in \mathcal{T} \cap \mathcal{Q}$, the search problem is drastically simplified because the query graph (nodes are terms and edges are atomics) is tree.
- Then one can find a topological order to remove each existential variable with $O(|\mathcal{E}|^2)$. Then the overall complexity is linear to the number of variables.
- This discussion also applies for the inference problem.

Methods: Search in the continuous space (1/3)

- The search problem

$$TV[\phi(y=a)] = \max_{x_1,\ldots,x_n \in \mathcal{E}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk} \Big|_{y=a} \right]$$

is defined over the discrete set $\mathcal{E}$.

- A continuous relaxation put $x_1, \ldots, x_n$ in the embedding space $\mathcal{X}$

$$TV[\phi(y=a)] = \max_{x_1,\ldots,x_n \in \mathcal{X}} \perp_{j=1,\ldots,N} \top_{k=1,\ldots,M_j} TV\left[a_{jk} \Big|_{y=a} \right]$$

- The optimization objective is differentiable as long as
  - $\top$ and $\perp$ are differentiable.
  - $TV\left[a_{jk} \Big|_{y=a} \right]$ are differentiable.

Methods: Search in the continuous space (2/3)

Are $TV\left[a_{jk}\big|_{y=a}\right]$ differentiable?

- Let's look inside

$$TV\left[a_{jk}\Big|_{y=a}\right] = f(h, r, t)$$

h, t are entities or variables already with assignments.

- Before, the $f(h, r, t)$ takes discrete entities as input.
- But it is eventually a link predictor.
  It is also OK to use entity embeddings

Methods: Search in the continuous space, the example

- Recall the discrete search example

$$TV[\phi(y=a)] = \max_{x\in\mathcal{E}, z\in\mathcal{E}} f(x,r,a) \top f(a,r,z) \top f(z,r,x)$$

- The continuous relaxation goes with

$$TV[\phi(y=a)] \approx \max_{x\in\mathcal{X}, z\in\mathcal{X}} f(x,r,a) \top f(a,r,z) \top f(z,r,x)$$

- This problem can be solved via gradient ascend!

Methods: Search in the continuous space (3/3)

There will be $|\mathcal{E}|$ problems if we evaluate $TV[\phi(y=a)], a \in \mathcal{E}$ separately.

A simpler trick, known as Continuous Query Decomposition (CQD)

$$TV[\exists y.\phi(y)] = \max_{x_1,\dots,x_n \in \mathcal{X}, y \in \mathcal{X}} \perp_{j=1,\dots,N} \top_{k=1,\dots,M_j} TV\left[a_{jk}\right]$$

Then with optimal $x_1^*, \dots, x_n^*$, we just need to evaluate the objective.

$$TV[\phi(y=a)] = \perp_{j=1,\dots,N} \top_{k=1,\dots,M_j} TV\left[a_{jk}\Big|_{y=a, x_1=x_1^*,\dots,x_n=x_n^*}\right]$$

Arakelyan, E., Daza, D., Minervini, P., & Cochez, M. (2020). Complex query answering with neural link predictors. *arXiv preprint arXiv:2011.03459*.

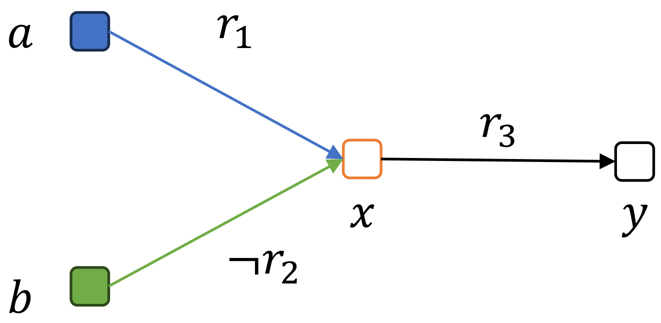
Methods: Learning to search in the continuous space

Can we skip the gradient ascend?

- Before answering yes or no, let's simplify the problem by consider the conjunctive queries separately.

$$\max_{x_1,\dots,x_n \in \mathcal{X}, y \in \mathcal{X}} \top_{k=1,\dots,M_j} TV[a_{jk}]$$

- This conjunctive queries can be considered as a query graph
- $\exists x. r_1(a, x) \wedge \neg r_2(b, x) \wedge r_3(x, y)$
- Or $\max_{x,y} f(a, r_1, x) \top [1 - f(b, r_2, x)] \top f(x, r_3, y)$



Methods: Learning to search in the continuous space

Search

Goal: optimize the embedding of x, y

Method: gradient ascend

$$\max_{x,y} f(a, r_1, x) \top [1 - f(b, r_2, x)] \top f(x, r_3, y)$$

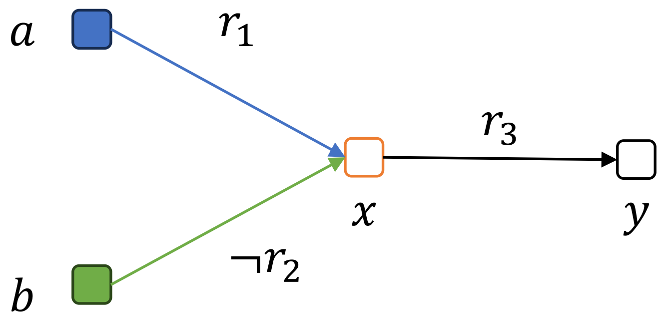
Learning to search

Goal: estimate the embedding of x, y

- Learn the pos. emb against neg. emb.

Method: neural network forward pass

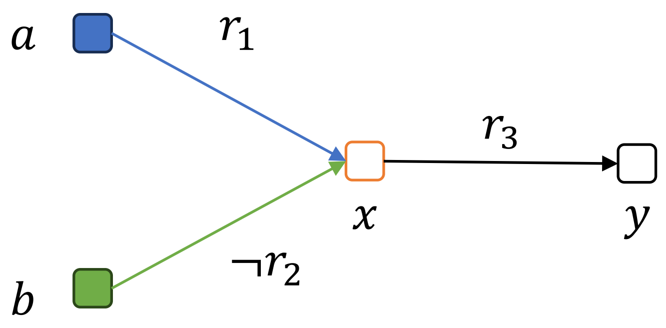
- New design problem of NN akin to the optimization process



Methods: Learning to search in the continuous space

Instead of optimizing the global objective, we optimize the local parts objective in each edge (atomic formula) with closed-form solutions

- One-hop inference problems
  - $\max_x f(h,r,x) \coloneqq \rho(h,r,h2t,0)$
  - $\max_x f(x,r,t) \coloneqq \rho(t,r,t2h,0)$
  - $\max_x 1 - f(h,r,x) \coloneqq \rho(h,r,h2t,1)$
  - $\max_x 1 - f(x,r,t) \coloneqq \rho(t,r,t2h,1)$



- $\rho(\text{entity}, \text{relation}, \text{direction}, \text{negation})$ is a message function
- A design problem: closed-form ρ for many link predictors

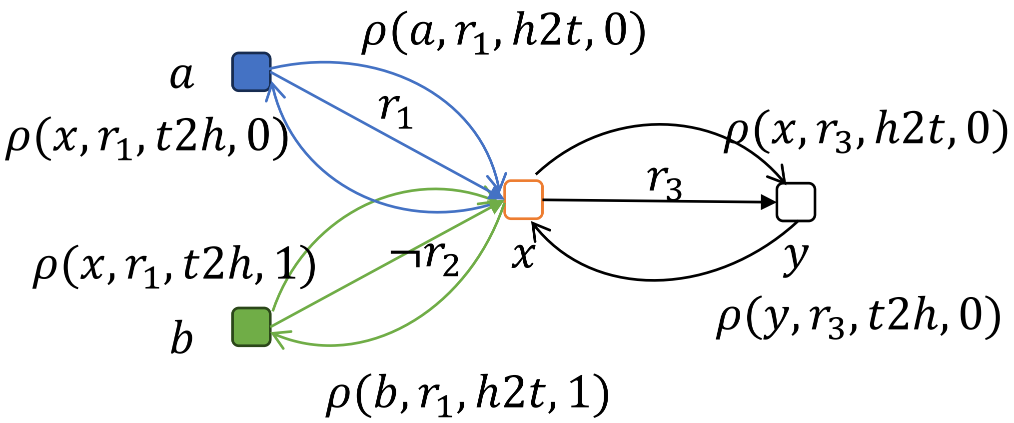
Wang, Z., Song, Y., Wong, G. Y., & See, S. (2023). Logical message passing networks with one-hop inference on atomic formulas. *arXiv preprint arXiv:2301.08859*.

Methods: Learning to search in the continuous space

The desired GNN design: Logical Message Passing Neural Networks

- Reused GIN layer with the proper message (1 MLP to train)
  - Able to approximate any functions
- Number of GIN layers = diameter of the graph
- Initialization with pretrained entity embeddings
- For variables initialized with special embeddings (2 embeddings to train)
  - One embedding for all existential variables.
  - One for the free query variable

✓ Then the embeddings of x, y are estimated



GNN end-to-end Training

Loss function: noisy contrastive estimation:

$$L = -\log \frac{e^{\cos(y,a)}}{e^{\cos(y,a)} + \sum_k e^{\cos\left(y, e_k^-\right)}}$$

The embedding of y is estimated by GNN

Entity embeddings are pretrained.

Wang, Z., Song, Y., Wong, G. Y., & See, S. (2023). Logical message passing networks with one-hop inference on atomic formulas. *arXiv preprint arXiv:2301.08859*.