Explaining Missing Data in Graphs: A Constraint-based Approach

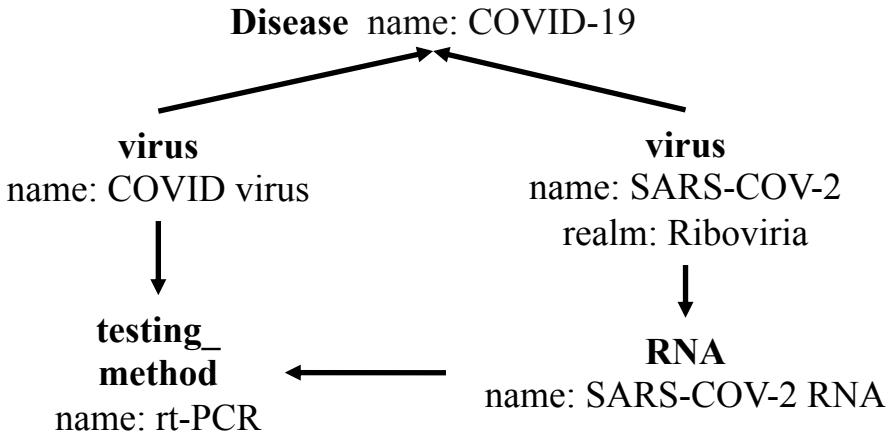
Qi Song[1] Peng Lin[2] Hanchao Ma[3] Yinghui Wu[3,4]

1 amazon 2 Washington State University 3 Case Western Reserve University think beyond the possible 4 Pacific Northwest National Laboratory

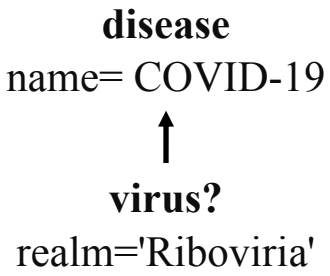
Explaining Missing Data in Graphs

- Real-world graphs are imcomplete: attribute-values of entities and relations are ofte missing
- Clarify **why** certain expected data is missing, **whether** such data can be restored, and **how**.
- Knowledge futions, user-centric data quality, query suggestion, etc

Graph G: COVID-19 medical knowledge base[1]



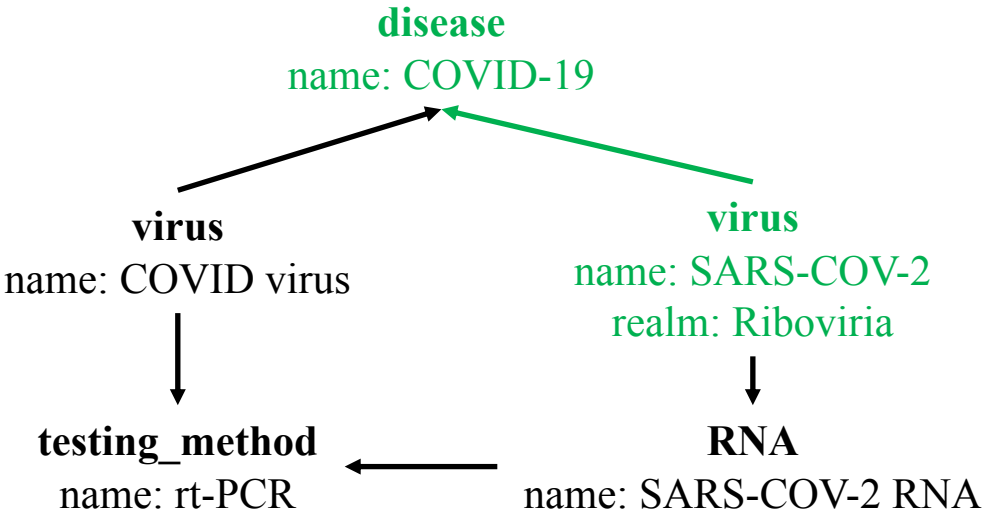
Query Q: Find all ***viruses*** that may be relevant to COVID-19 and has a realm 'Riboviria'



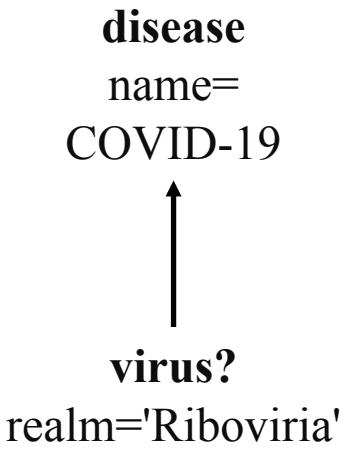
1. https://github.com/Knowledge-Graph-Hub/kg-covid-19

Explaining Missing Data in Graphs

Graph G: COVID-19 medical knowledge base



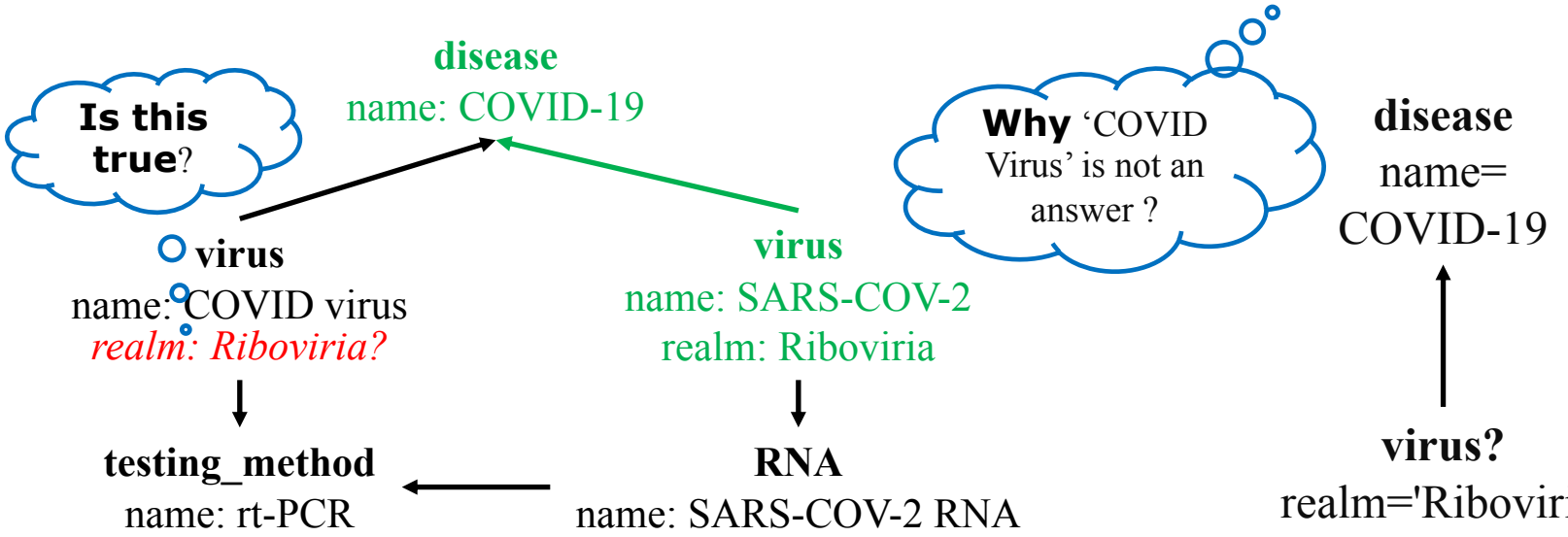
Query Q: Find all ***viruses*** that may be relevant to COVID-19 and has a realm 'Riboviria'



Explaining Missing Data in Graphs

Graph G: COVID-19 medical knowledge base

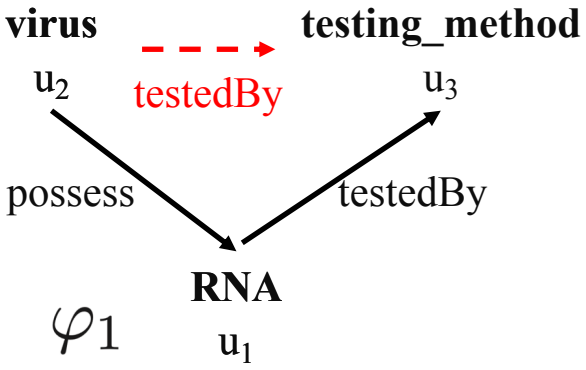
Query Q: Find all ***viruses*** that may be relevant to COVID-19 and has a realm 'Riboviria'

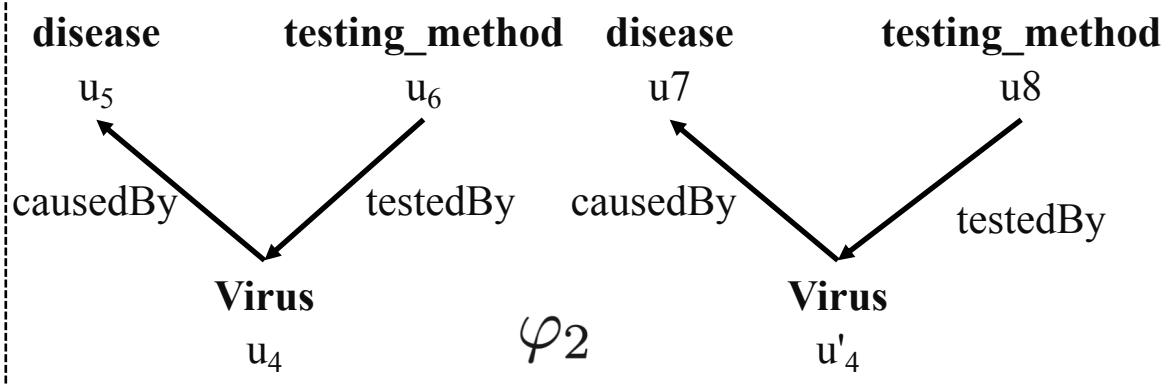


new data needs to be inferred!

Graph Data Constraints

- Data constraints can be used to capture missing data in a graph G.
  - Graph association rules: $P \rightarrow r(u, u')$ (*infers missing edges*)
  - Key Constraints: $P \rightarrow (u.id = u'.id)$ *(node equality; or other equality constraints)*



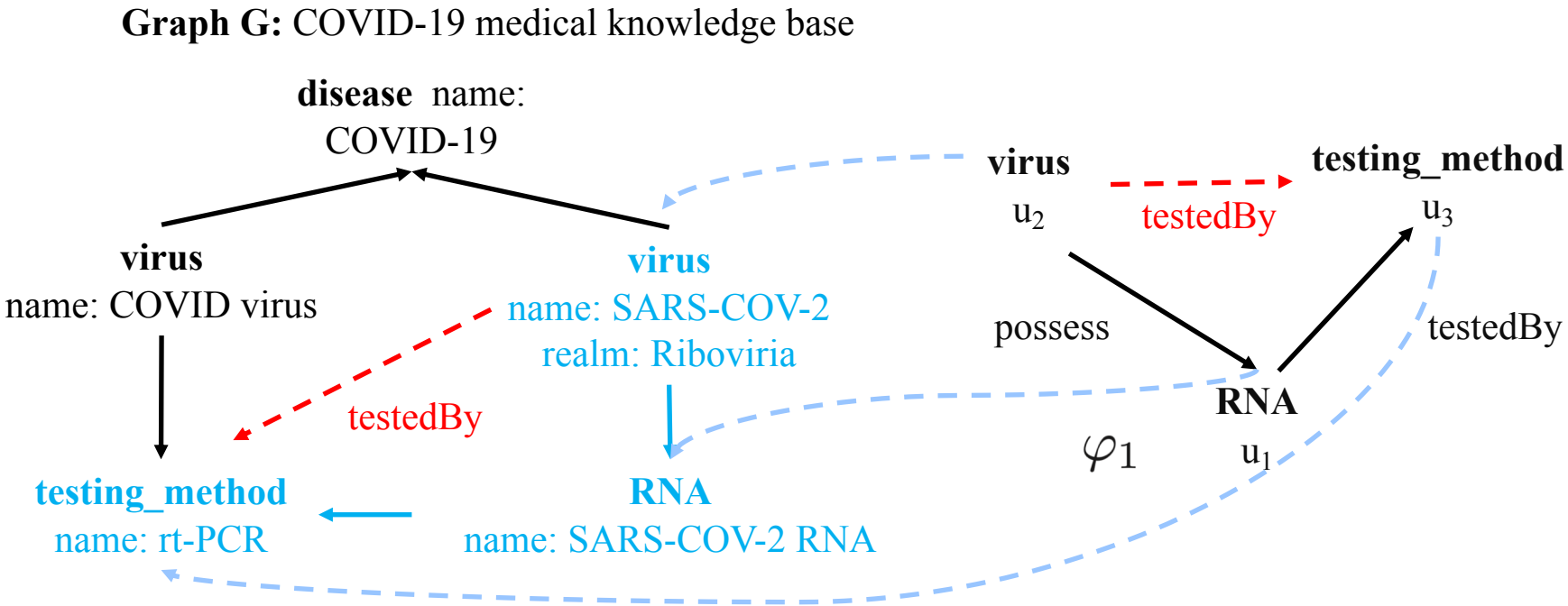


φ_1 (Graph association rule): "if a virus possesses a specific RNA which can be identified by a testing method, then this method can be used to identify the virus".

φ_2 (graph key): "if two viruses cause the same disease and can be identified by the same testing method, then they refer to the same realm".

Explaining Missing Edges via Graph Association Rules

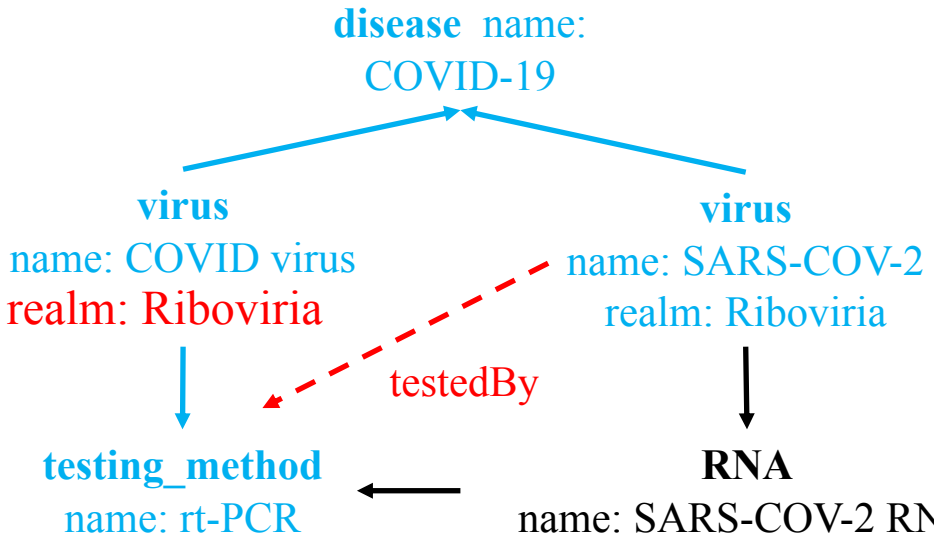
- Enforcing φ_1 "inserts" a missing edge between 'SARS-COV-2' and 'rt-PCR'.

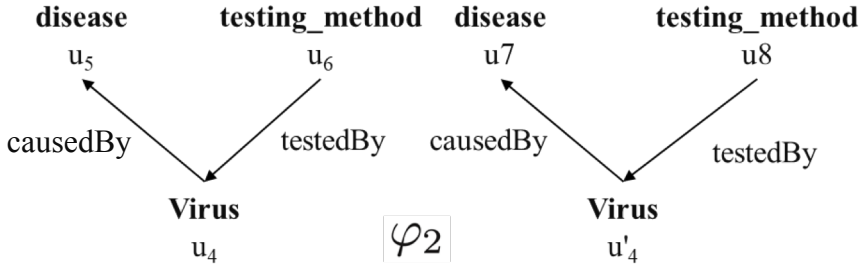


Inferring Missing Attribute Values via Node Equivalence/Keys

- Enforcing φ_1 "inserts" a missing edge between 'SARS-COV-2' and 'rt-PCR'.
- Enforcing φ_2 enriches the missing "realm" information of virus $v1$.

Graph G: COVID-19 medical knowledge base



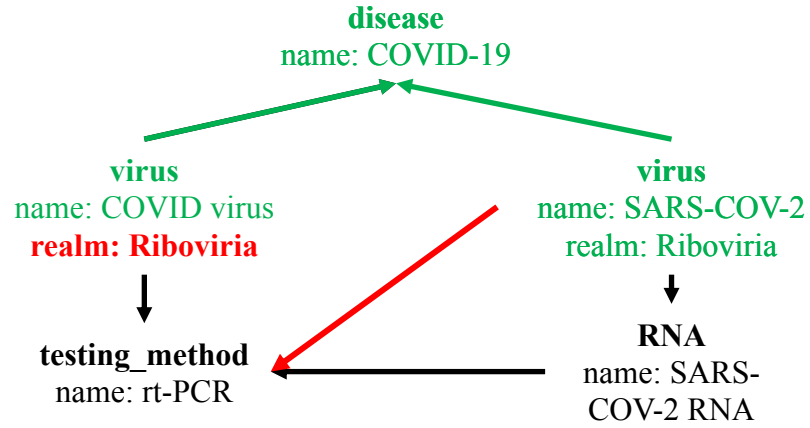


Putting these into a "sequence"

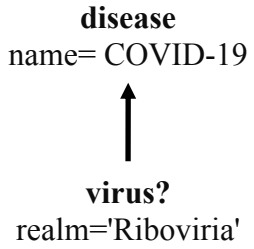
- Enforcing φ_1 "inserts" a missing edge between 'SARS-COV-2' and 'rt-PCR'.
- Enforcing φ_2 enriches the missing "realm" information of virus $v1$.
- These two steps enrich G and restores 'COVID virus' in $Q(G)$

Idea: Constraint-based Explanation

Graph G: COVID-19 medical knowledge base



Query Q: Find all viruses that may be relevant to COVID-19 and has a realm 'Riboviria'



Answer: SARS-COV-2, COVID virus

Constraint-based Explanation

- Idea: perform graph rewriting process that transforms graphs by "enforcing" data constraints to verify the occurrence of missing elements.
- Given a missing element g, a query Q with result $Q(G_1)$ in graph G_1, and data constraints Σ, g can be explained by Σ in G_1 (**Σ-explainable**), if

$$G_1 \xrightarrow{\varphi 1} G_2 \xrightarrow{\varphi 2} \ldots G_n \xrightarrow{\varphi n} G'$$

such that $g \in Q(G')$.

- An "explanation" : a sequence of "actions" of enforcement of data constraints.

Occam's razor: minimal explanation

Church-Rosser Property

- For any ∑ of GKs and GARs, any sequence is terminating, and any terminating sequences generate graphs up to graph homomorphism

merge operator: $\circ(v,v')$

Use a union function to merge two nodes v, v' into a new node v''(an equivalent class)

insertion operator: $\oplus((v,v'),r)$

Insert an edge with label r between v and v'

- Information looseness: this process does not modify data;
- Query result preserving: this process preserves the initial query result.

Explaining Missing Data: Problem Statement

- **Input:** graph G, a missing element $g \notin G$, graph data constraints **Σ**, a bound b,
- **Output:** a minimal explanation ρ for g, such that

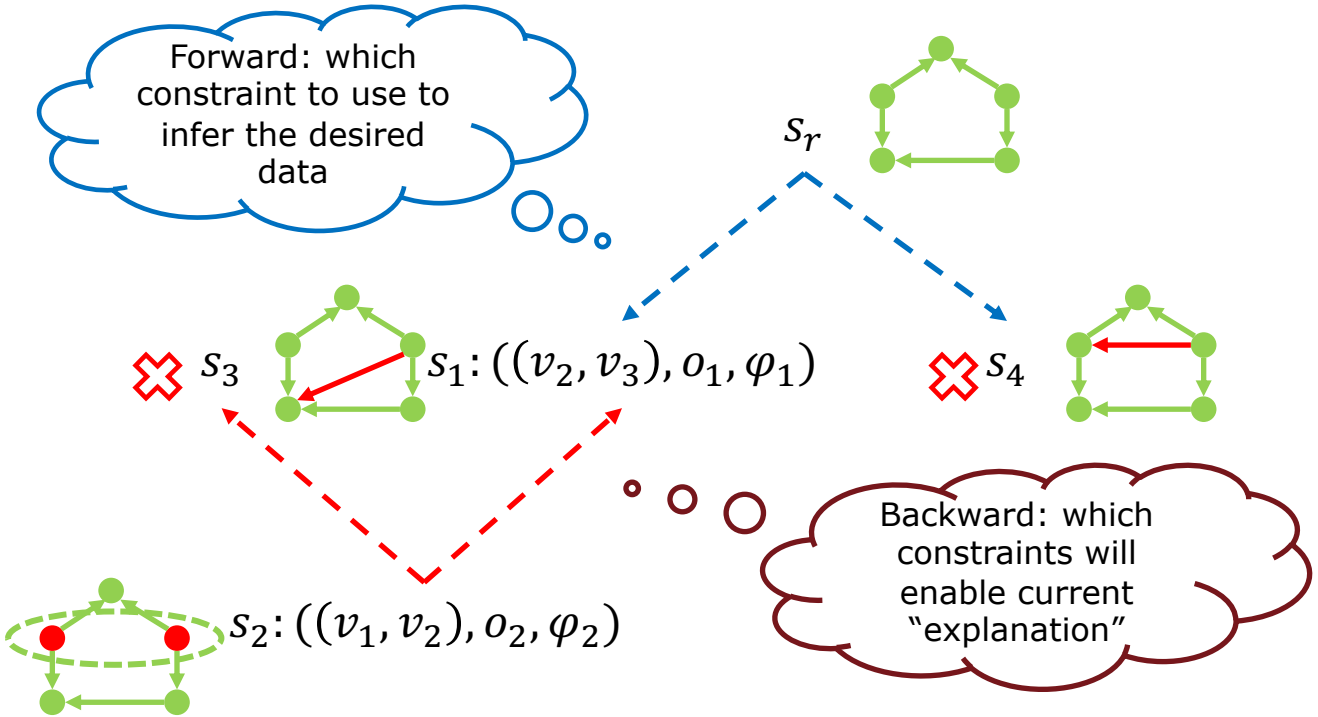
$$\rho = \arg\max_{|\rho'| \le b} cg(\rho', G)$$

$$cg(\rho, G) = \sum_{s \in \rho} supp(s, G) \cdot cg(s, G)$$

missing element g	Hardness	Description	Time cost
missing answer	NP-hard	Bi-directional algorithm	$O(T \cdot (\lvert V\rvert^2 \lvert\Sigma\rvert)^{\frac{b}{2}}$
missing edge or attribute value		Bi-directional algorithm	$O(T \cdot (\lvert V\rvert^2 \lvert\Sigma\rvert)^{\frac{b}{2}}$
wildcard '_'		Approximation of $6 \cdot \ln(\lvert V\rvert) + 1$	$O(T \cdot \lvert\Sigma\rvert \lvert V\rvert^2 \frac{B}{c_l})$

Computing Optimal Explanations - Algorithm

- Our idea: performing a bi-directional exploration



Experimental Setting

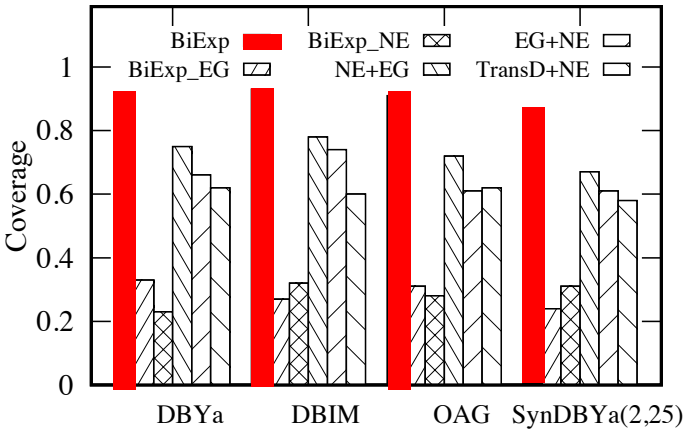
- Datasets

Name	Description	# of nodes	# of edges	# of equivalent nodes
DBYa	DBPedia+Yago	592K	4.5M	50K
IMDb	DBPedia+IMDb	33K	200K	33.4K
OAG	Aminer+Microsoft academic graph	2.5M	5.2M	106K

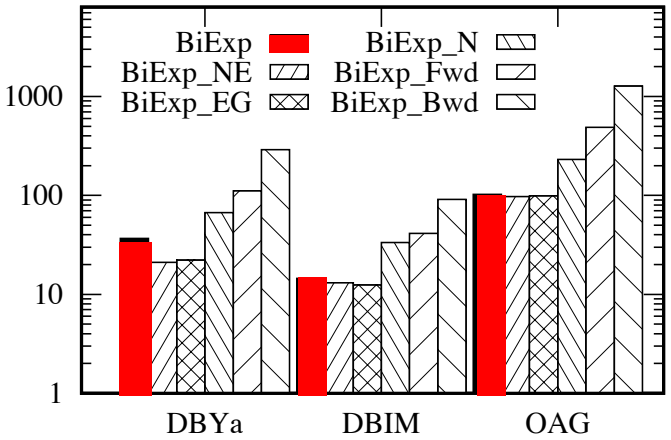
- Constraint generation: graph keys and graph association rules generator.
- Query generation: SPARQL benchmark.
- Algorithms – Explaining missing values
  - BiExp (optimized), BiExp_N (no pruning strategy)
  - BiExp_NE/BiExp_EG (with Σ contains only keys or association rules)
  - NE+EG/EG+NE (apply all keys and association rules in a batch)
  - TransD+NE (use TransD to replace association rules)
  - BiExp_Fwd/BiExp_Bwd (use only forward or backward search).

Experimental Result

- Answering Why questions: efficiency and effectiveness



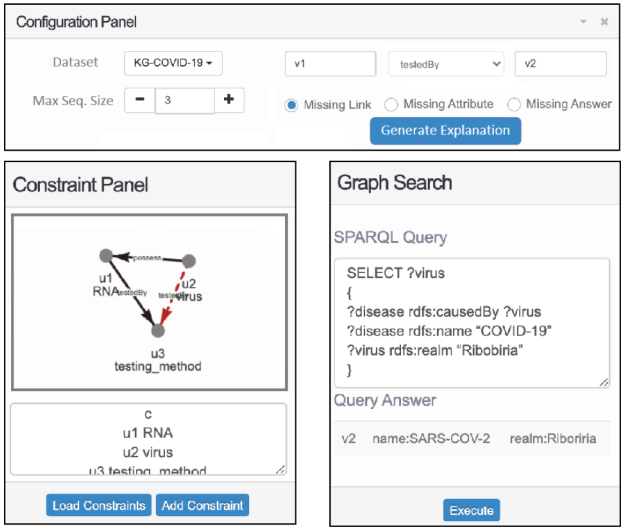
BiExp outperforms NE+EG, and TransD+NE by 17%, and 33%, respectively

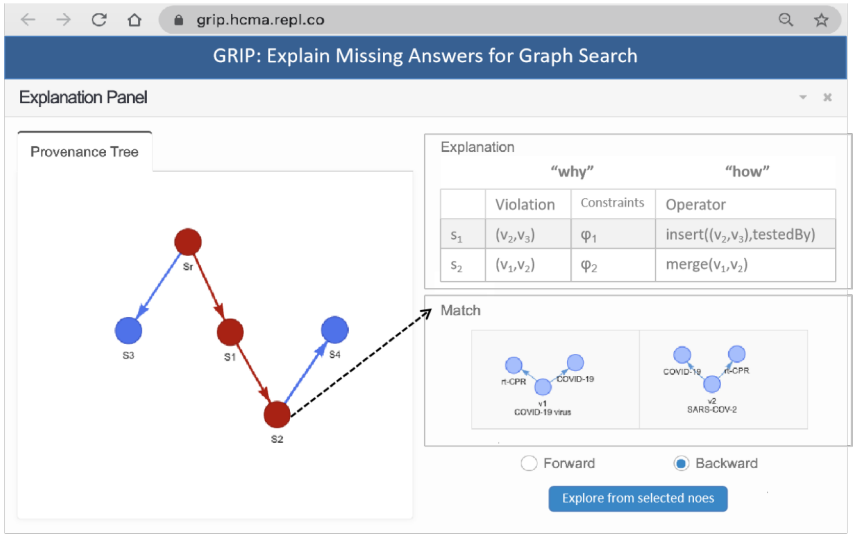


BiExp outperforms BiExp_N, BiExp_Fwd and BiExp_Bwd by 2.2, 3.5 and 8.9 times.

- BiExp improves the coverage of AMIE+Vickey by 22% and is 3.6 times faster;

GRIP: A Demo system





GRIP: Constraint-based Explanation of Missing Answers for Graph Queries

SIGMOD'21 Demo https://grip.hcma.repl.co/

Thank you!