ICDE 2020
International Conference on Data Engineering

Repairing Entities using Star Constraints in Multi-relational Graphs

Peng Lin[1] Qi Song[1] Yinghui Wu[2,3] Jiaxing Pi[4]

1 Washington State University

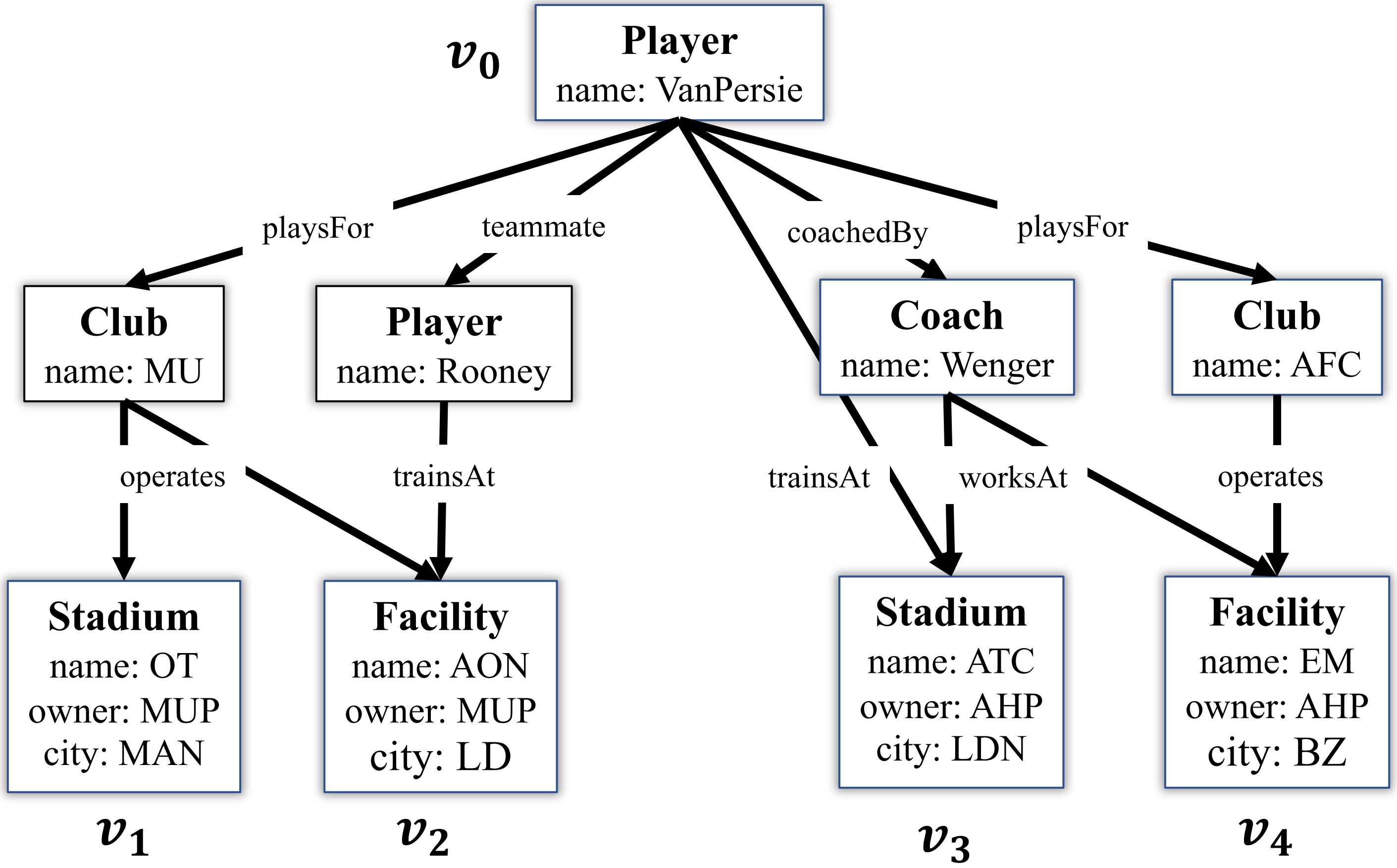
2 Case Western Reserve University EST. 1826

3 Pacific Northwest National Laboratory

4 SIEMENS

Erroneous entities: how to capture?

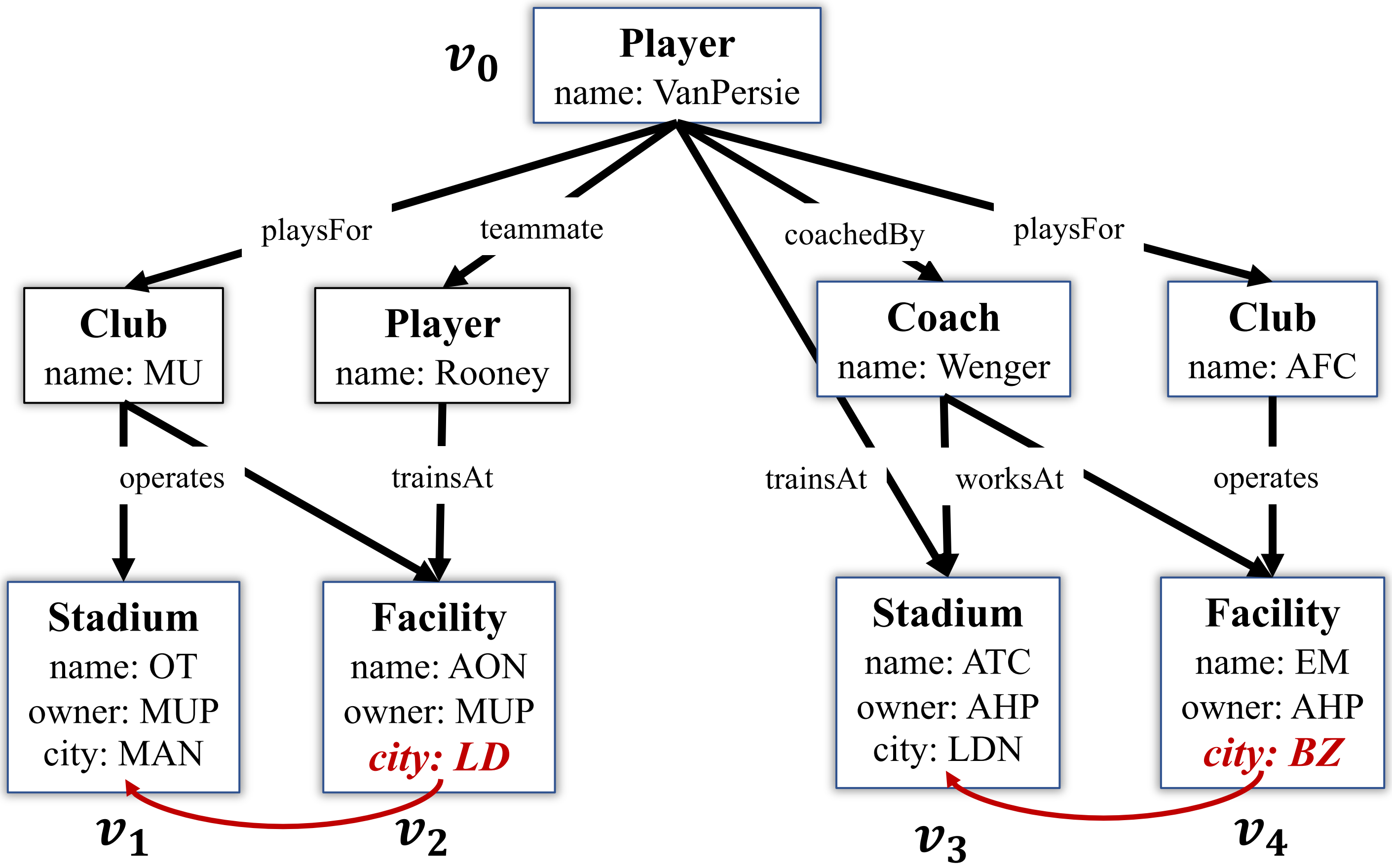
- Multi-relational graphs: a labeled graph with attributes on nodes



Graph G: a football database

Erroneous entities: how to capture?

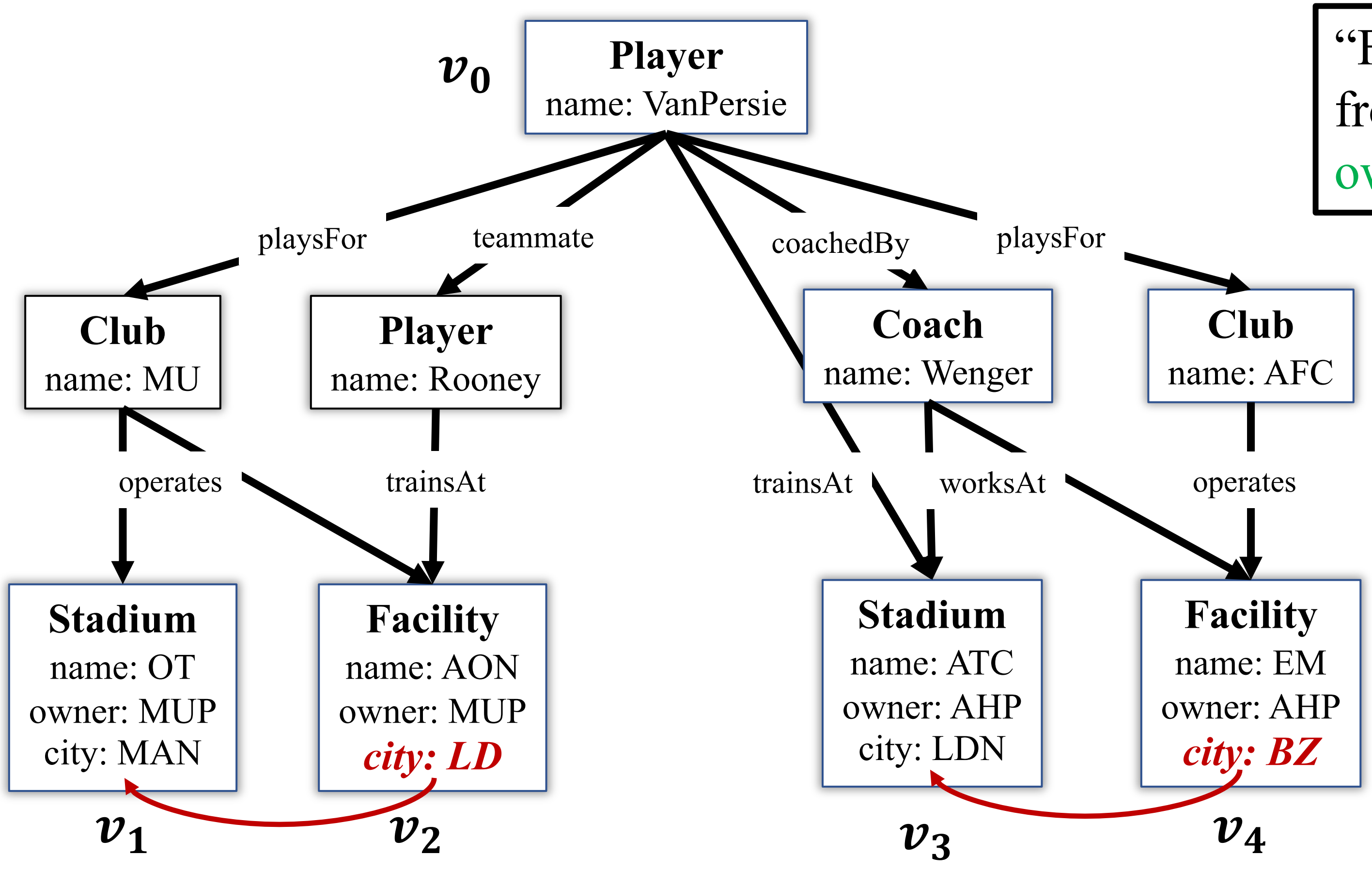
- Multi-relational graphs: a labeled graph with attributes on nodes
- Entity errors: incorrect node attributes



Graph G: a football database

Erroneous entities: how to capture?

- Multi-relational graphs: a labeled graph with attributes on nodes
- Entity errors: incorrect node attributes
- Semantics: *relevant paths from a center node*

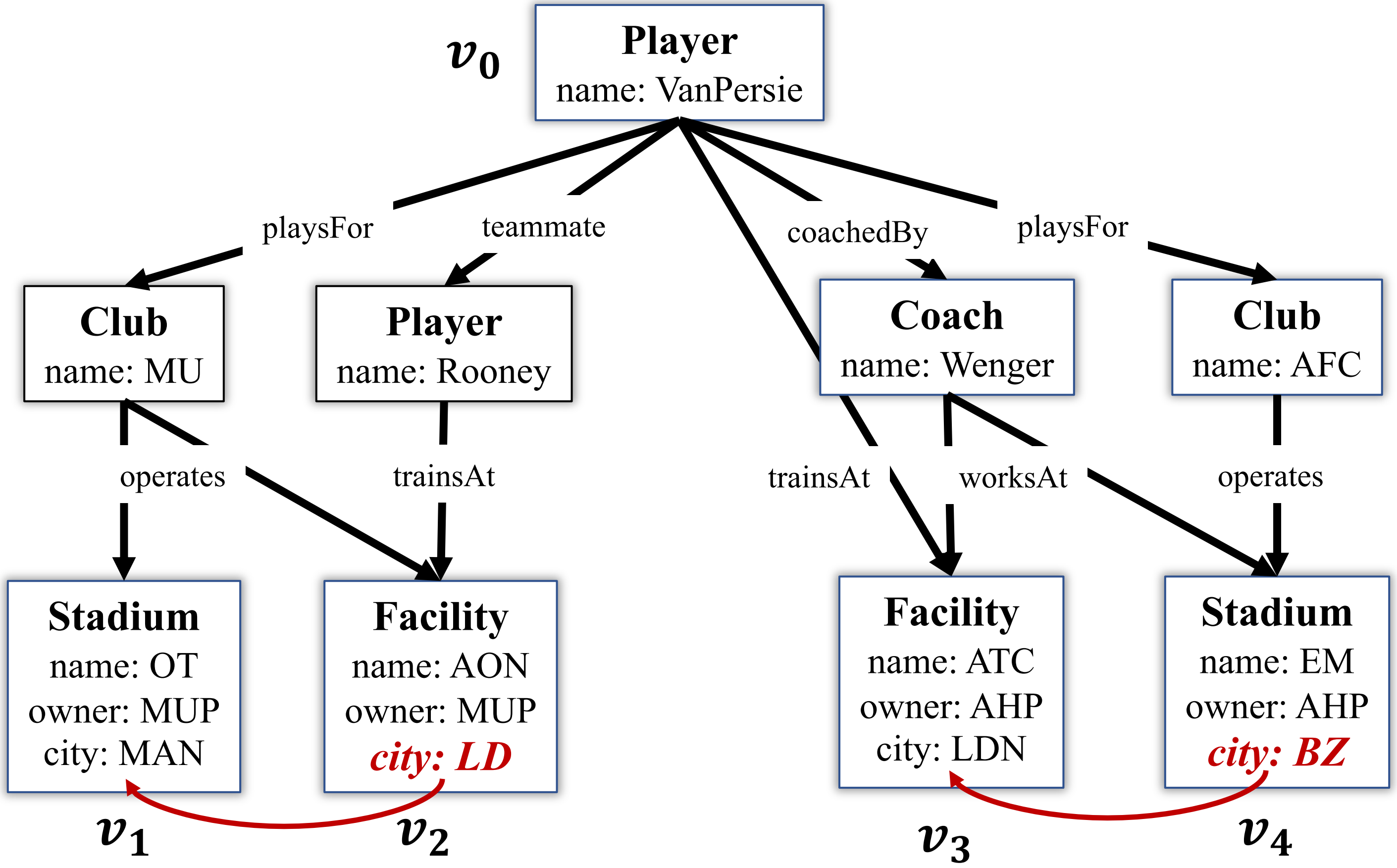


Graph G: a football database

"For stadium and facility *relevant* to player (v_0) from Premier League, if they have the same owner, then they should locate at the same city."

Regular path queries

- Regular expressions: $R = l|l^{\leq k}|R \cdot R|R \cup R$

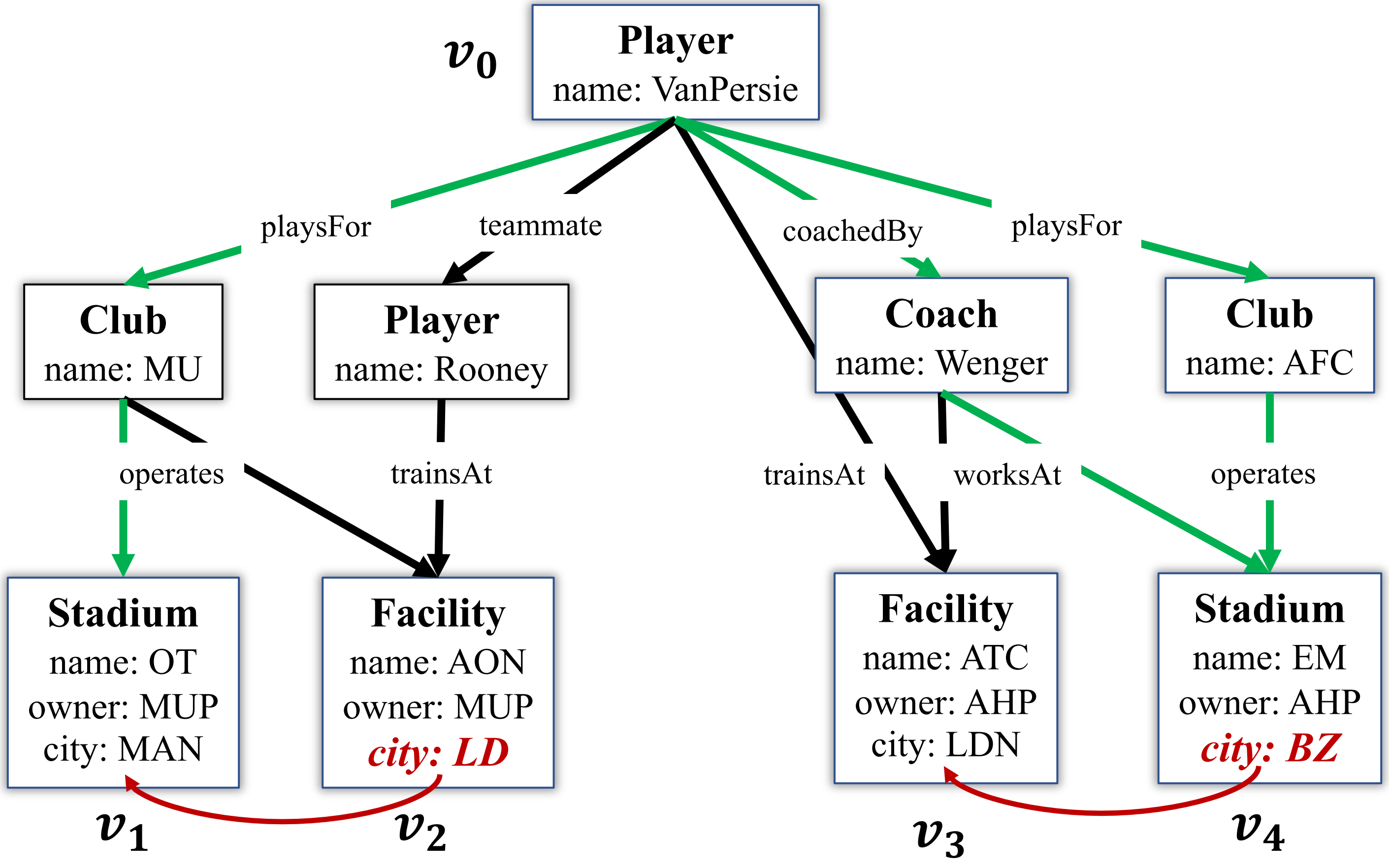


Graph G: a football database

Regular path queries

- Regular expressions: $R = l | l^{\leq k} | R \cdot R | R \cup R$

- Paths from **Player** to **Stadium**
- $R_1 = (\text{playsFor} \cdot \text{operates}) \cup (\text{coachedBy} \cdot \text{worksAt})$



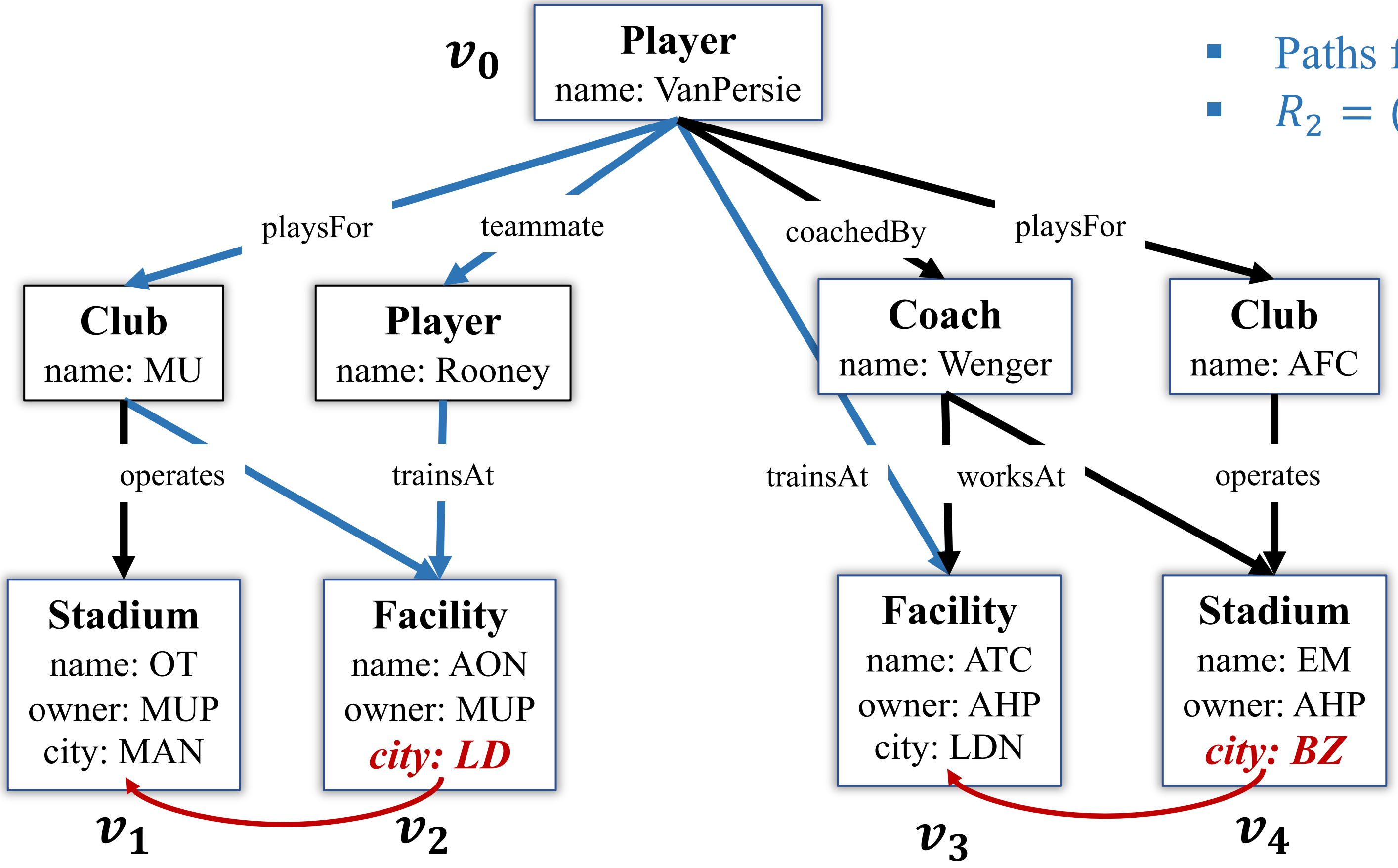
Graph G: a football database

Regular path queries

- Regular expressions: $R = l|l^{\leq k}|R \cdot R|R \cup R$

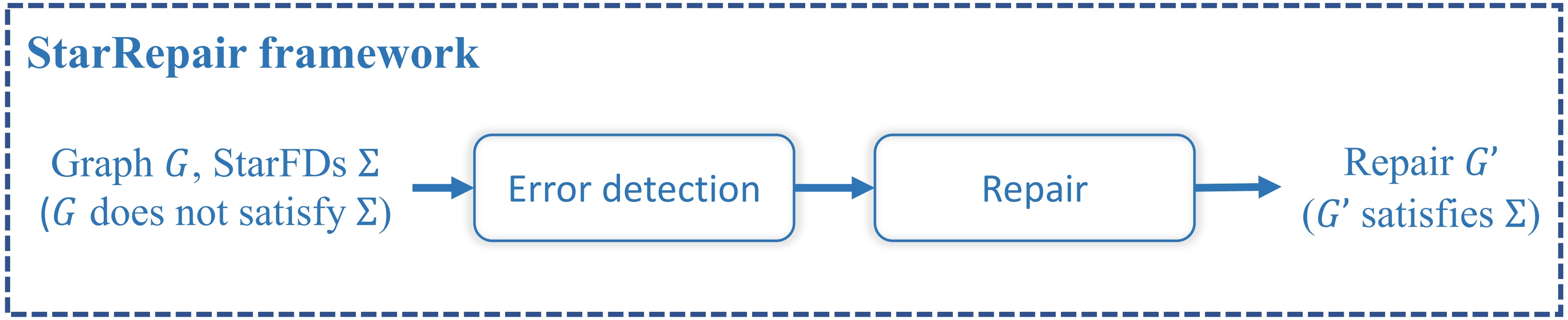
- Paths from **Player** to **Stadium**
- $R_1 = (\text{playsFor} \cdot \text{operates}) \cup (\text{coachedBy} \cdot \text{worksAt})$

- Paths from **Player** to **Facility**
- $R_2 = (\text{playsFor} \cdot \text{operates}) \cup (\text{teammate}^{\leq 1} \cdot \text{trainsAt})$



Graph G: a football database

Contributions



StarRepair framework

Graph G, StarFDs Σ
(G does not satisfy Σ)

Error detection

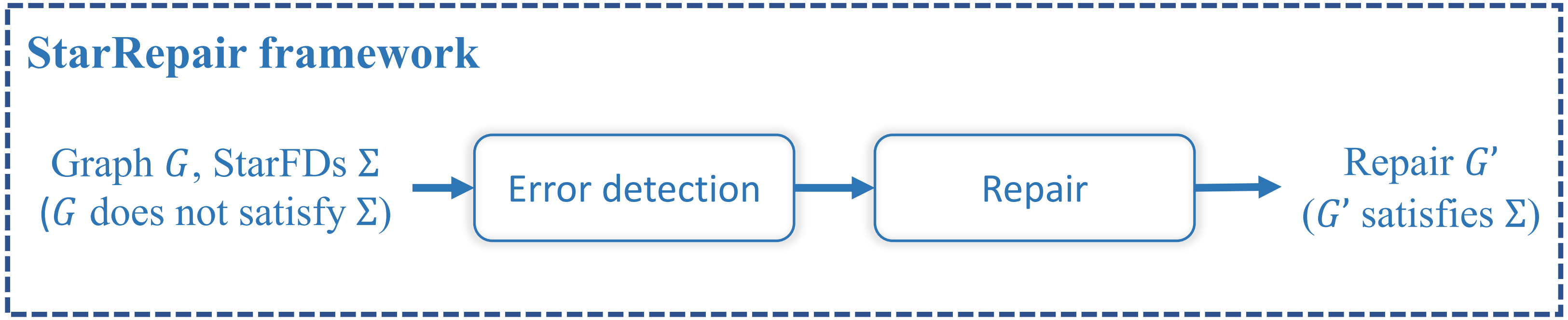
Repair

Repair G'
(G' satisfies Σ)

Contributions

StarFDs: star functional dependencies new constraints for graphs

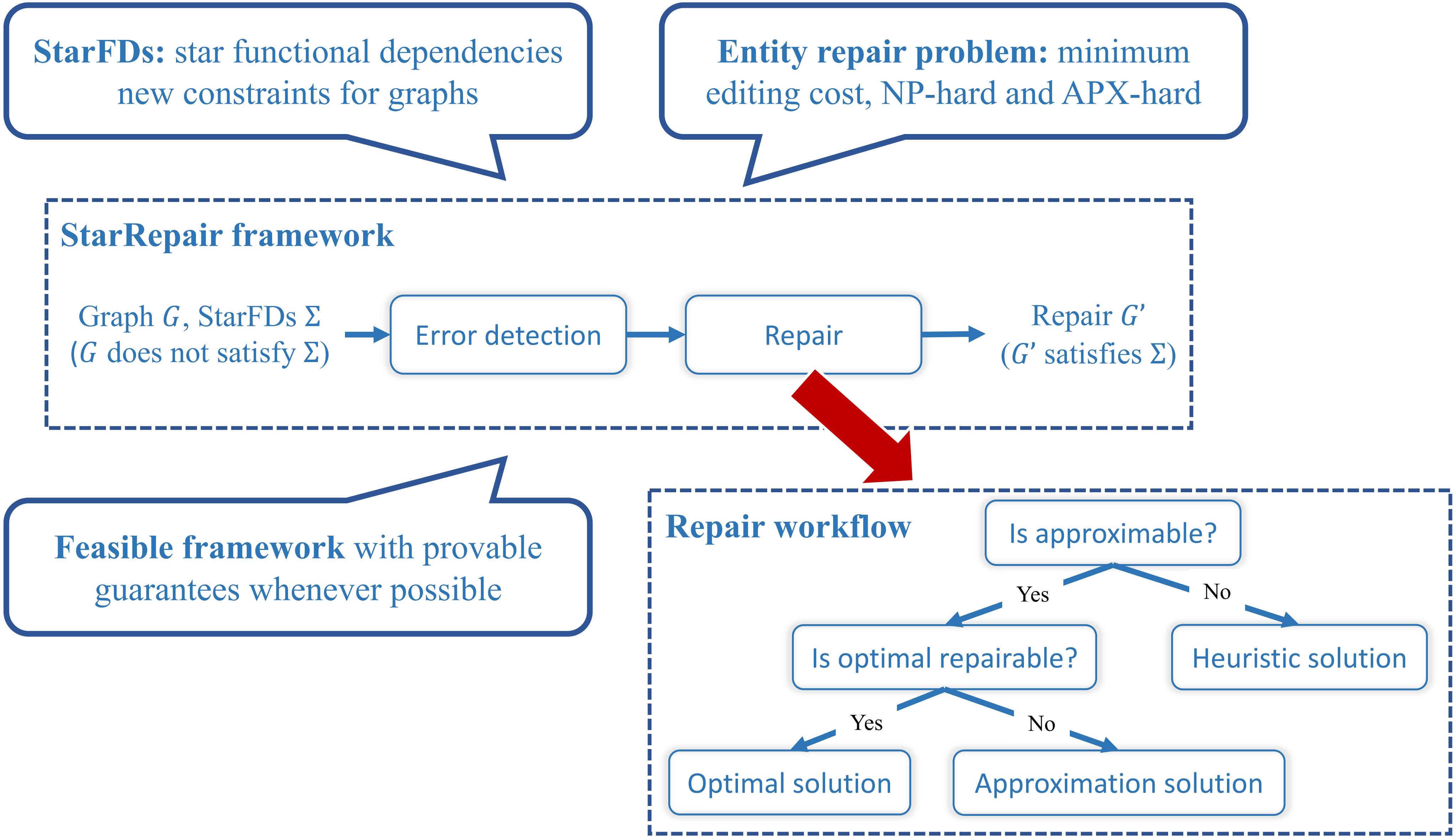
Entity repair problem: minimum editing cost, NP-hard and APX-hard


StarRepair framework

Graph G, StarFDs Σ (G does not satisfy Σ) → Error detection → Repair → Repair G' (G' satisfies Σ)

Feasible framework with provable guarantees whenever possible

Contributions



StarFDs: star functional dependencies new constraints for graphs

Entity repair problem: minimum editing cost, NP-hard and APX-hard

StarRepair framework

Graph G, StarFDs Σ (G does not satisfy Σ)

Error detection

Repair

Repair G' (G' satisfies Σ)

Feasible framework with provable guarantees whenever possible

Repair workflow

Is approximable?

Yes

No

Is optimal repairable?

Heuristic solution

Yes

No

Optimal solution

Approximation solution

Star constraints

- **StarFDs:** $\varphi = (P(u_o), X \rightarrow Y)$

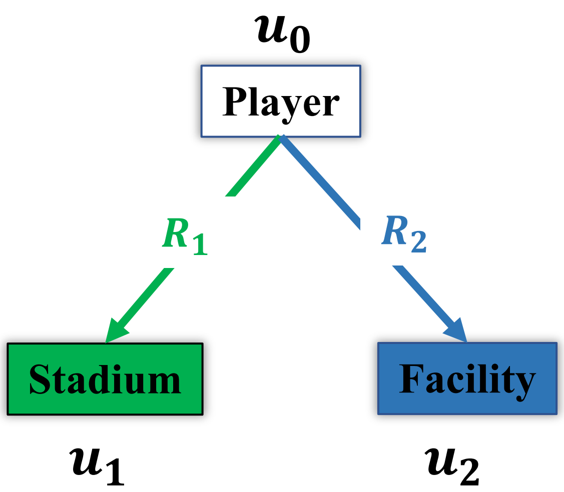
- Star pattern $P(u_o)$:

- Value constraints: $X \rightarrow Y$

Star constraints

- **StarFDs:** $\varphi = (P(u_o), X \rightarrow Y)$

- Star pattern $P(u_o)$:
  - A two-level tree with center node u_o
  - Each branch is a regular expression



$R_1 = (\text{playsFor} \cdot \text{operates}) \cup (\text{coachedBy} \cdot \text{worksAt})$

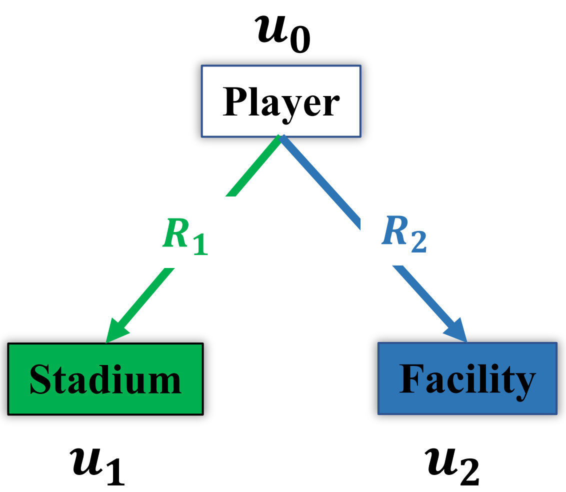
$R_2 = (\text{playsFor} \cdot \text{operates}) \cup (\text{teammate}^{\leq 1} \cdot \text{trainsAt})$

- Value constraints: $X \rightarrow Y$

Star constraints

- **StarFDs:** $\varphi = (P(u_o), X \rightarrow Y)$

- Star pattern $P(u_o)$:
  - A two-level tree with center node u_o
  - Each branch is a regular expression



$R_1 = (\text{playsFor} \cdot \text{operates}) \cup (\text{coachedBy} \cdot \text{worksAt})$

$R_2 = (\text{playsFor} \cdot \text{operates}) \cup (\text{teammate}^{\leq 1} \cdot \text{trainsAt})$

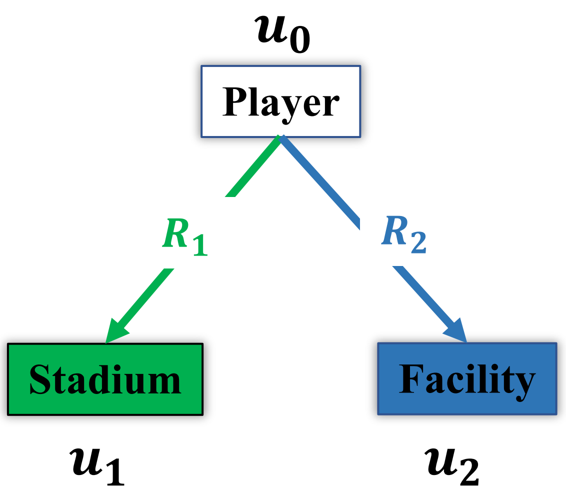
- Value constraints: $X \rightarrow Y$
  - X and Y are two sets of literals
  - Literals: $u.A = c$, or $u.A = u'.A'$

$X : u_o.\text{league} = \text{EPL}, u_1.\text{owner} = u_2.\text{owner}$

$Y : u_1.\text{city} = u_2.\text{city}$

Star constraints

- Matching semantics: maximum set matched by star pattern



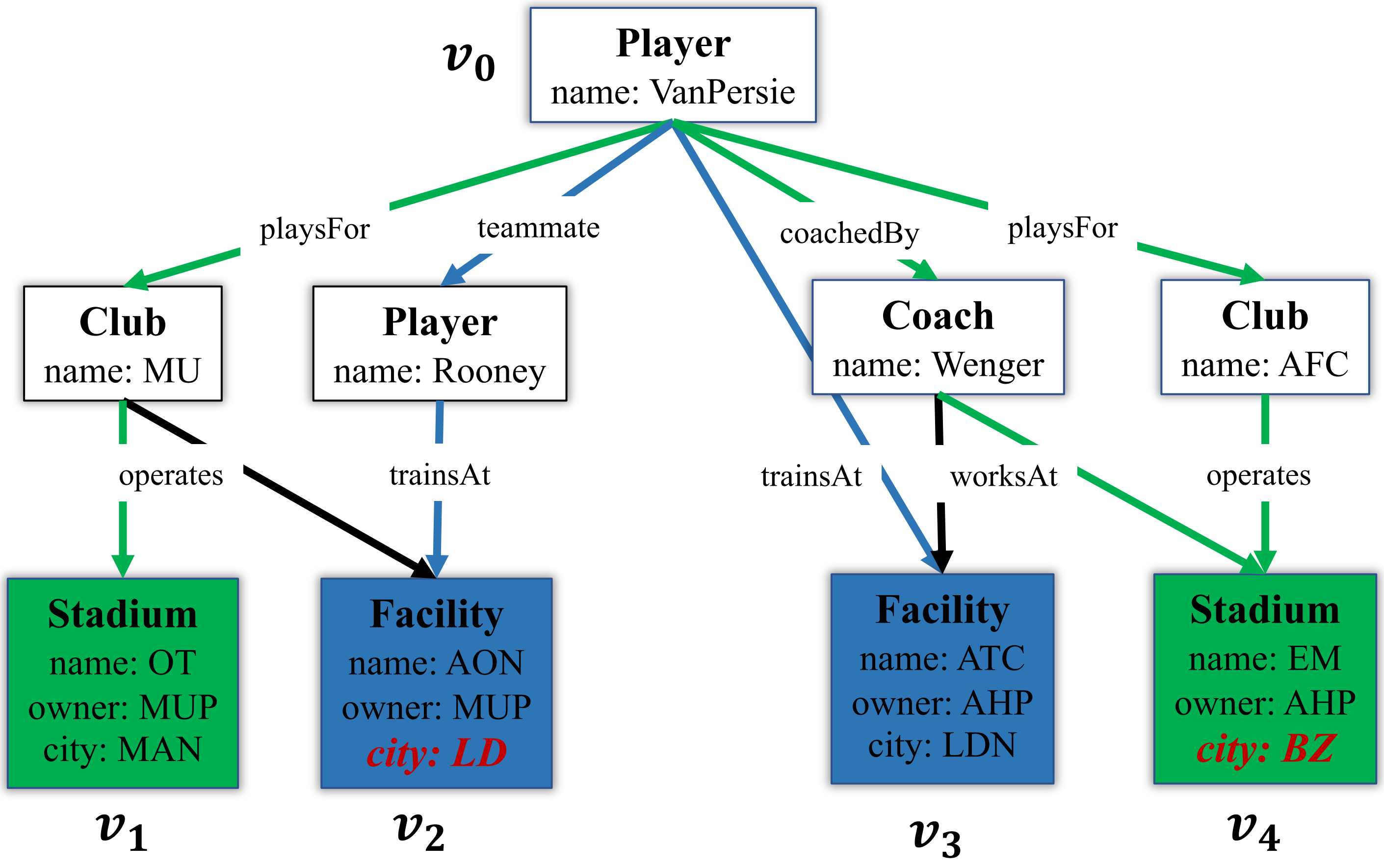
Star pattern $P(u_o)$

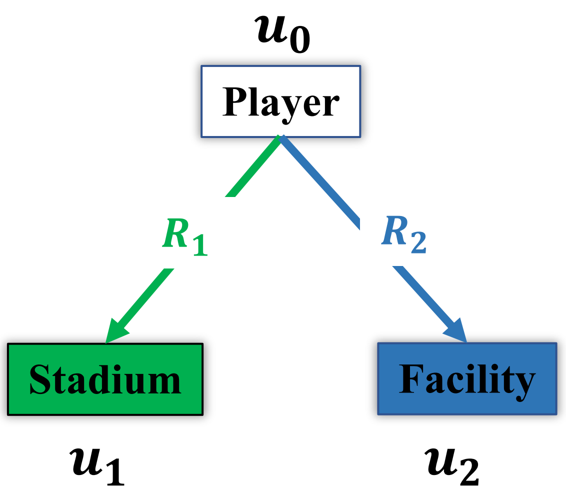
$X : u_o.\text{league} = \text{EPL}, u_1.\text{owner} = u_2.\text{owner}$
$Y : u_1.\text{city} = u_2.\text{city}$

Star constraints

- Matching semantics: maximum set matched by star pattern

$\boldsymbol{u_0}$ matches $\boldsymbol{v_0}$
$\boldsymbol{u_1}$ matches $\boldsymbol{v_1}$ and $\boldsymbol{v_4}$
$\boldsymbol{u_2}$ matches $\boldsymbol{v_2}$ and $\boldsymbol{v_3}$





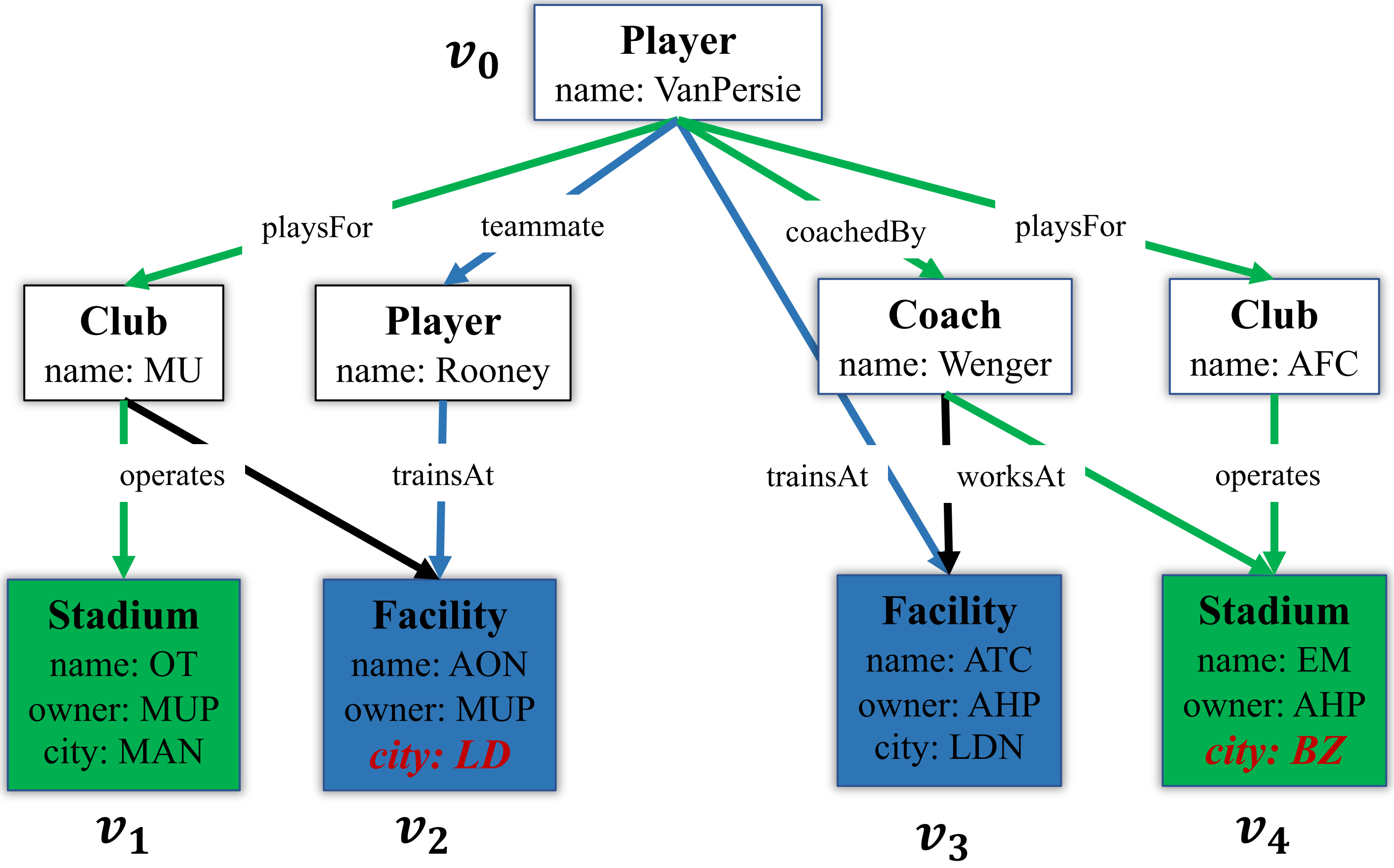
Star pattern $P(u_o)$

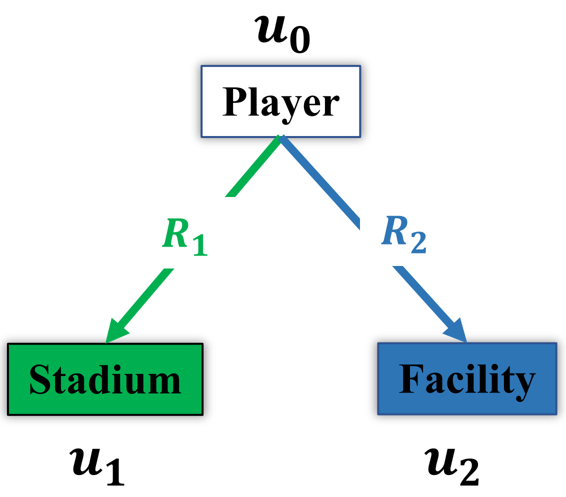
$X : u_o.\text{league} = \text{EPL}, u_1.\text{owner} = u_2.\text{owner}$
$Y : u_1.\text{city} = u_2.\text{city}$

Star constraints

- Matching semantics: maximum set matched by star pattern
- Inconsistencies ***I***: matches that X holds but Y does not hold

$\boldsymbol{u_0}$ matches $\boldsymbol{v_0}$
$\boldsymbol{u_1}$ matches $\boldsymbol{v_1}$ and $\boldsymbol{v_4}$
$\boldsymbol{u_2}$ matches $\boldsymbol{v_2}$ and $\boldsymbol{v_3}$





Star pattern $P(u_o)$

$X : u_o.\text{league} = \text{EPL}, u_1.\text{owner} = u_2.\text{owner}$
$Y : u_1.\text{city} = u_2.\text{city}$

Summary of results

Problem	Description	Hardness	Solution
Satisfiability	**Input:** Σ decide whether there exists G that satisfies Σ	NP-complete	
Implication	**Input:** Σ and φ decide whether for all G satisfy Σ, they satisfy φ	coNP-hard	
Error detection (validation)	**Input:** G and Σ **Output:** all inconsistencies $\boldsymbol{I}$	PTIME	Evaluate regular path queries and validate values - time complexity: $O(\|\Sigma\|\|V\| + \|V\|(\|V\| + \|E\|))$
Repair	**Input:** Σ and G that does not satisfy Σ **Ouput:** G' that satisfies Σ with least repair cost	NP-hard APX-hard	Approximable cases (PTIME checkable) - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|\|\Sigma\|))$ - approximation ratio: $\|\boldsymbol{I}\|\|\Sigma\|^2$
			Optimal cases - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|))$
			Heuristic cases - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|\|\Sigma\|))$ - bounded repairable: cost $\leq \|\boldsymbol{I}\|$

- Notations

G: graph V: nodes E: edges

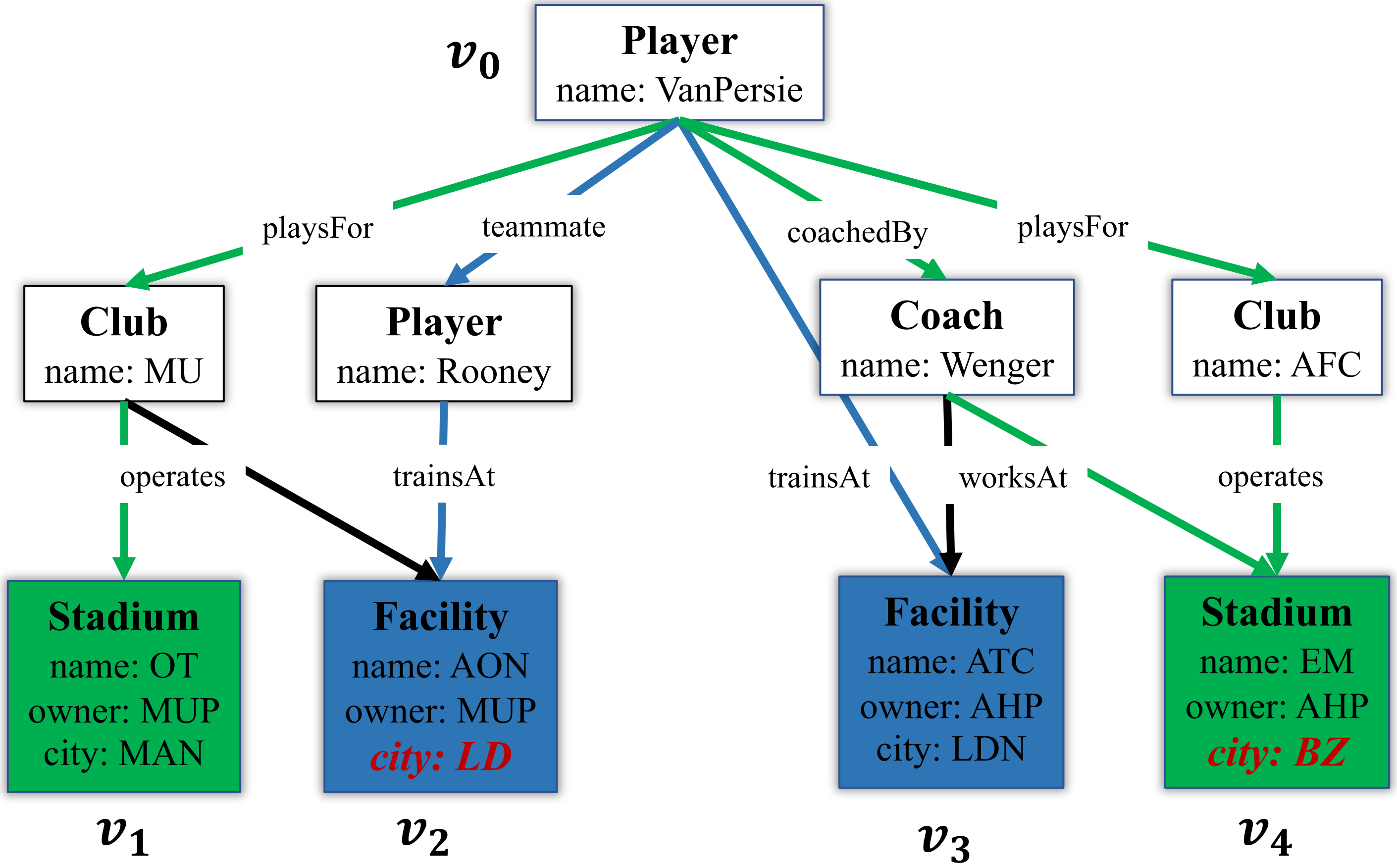
Σ: a set of StarFDs φ: a single StarFD $\boldsymbol{I}$: all inconsistencies.

Updates and repairs

- Updates O: operators $o = (v.A, a, c)$ with editing cost $\mathrm{cost}(O) = \sum_{o \in O} \mathrm{cost}(o)$
- Repair O: applying O to G, such that obtain G' that satisfies Σ

Updates and repairs

- Updates O: operators $o = (v.A, a, c)$ with editing cost $\text{cost}(O) = \sum_{o \in O} \text{cost}(o)$
- Repair O: applying O to G, such that obtain G' that satisfies Σ



Two repairs:
$O_1 = \{(v_2.\text{city, LD, MAN}), (v_4.\text{city, BZ, LDN})\}$
$O_2 = \{(v_2.\text{owner, MUP, CFG}), (v_4.\text{owner, EM, ENIC})\}$

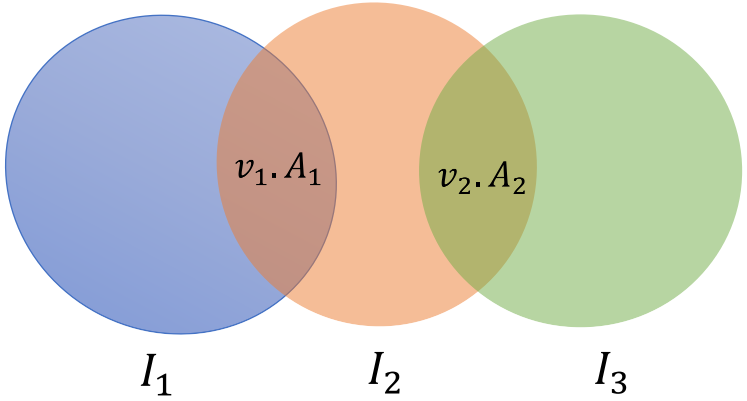
Entity repair problem

- **Input:** StarFDs Σ, and graph G does not satisfy Σ
- **Output:** a repair O, such that
  - obtain G' that satisfies Σ
  - $\text{cost}(O) \leq \text{cost}(O')$ for any O'

Entity repair problem

- **Input:** StarFDs Σ, and graph G does not satisfy Σ
- **Output:** a repair O, such that
  - obtain G' that satisfies Σ
  - cost(O) ≤ cost(O') for any O'

- Solution overview
  - Connected components (CCs): inconsistencies connected at shared node attributes
  - Isolated CCs: no new inconsistency is introduced when a CC is repaired

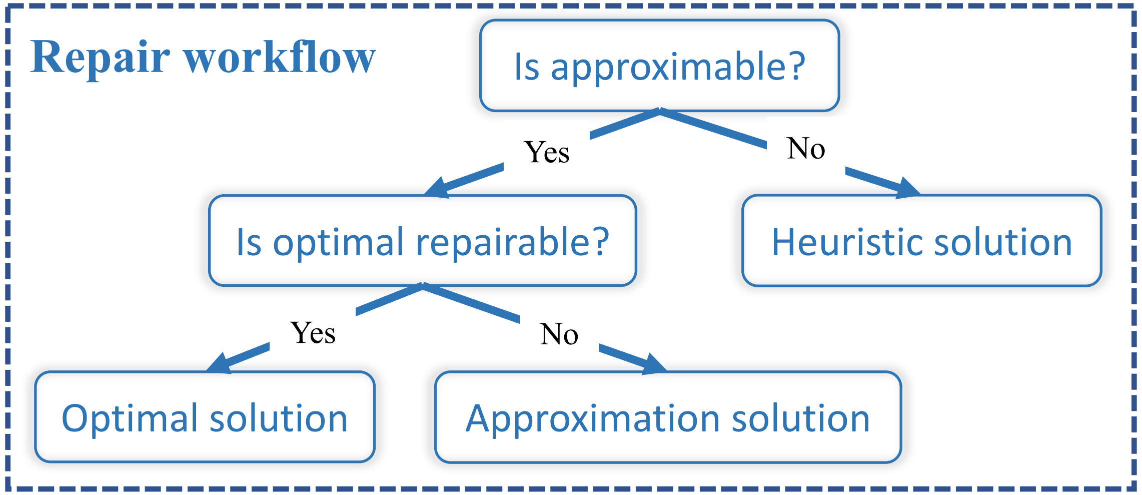


Entity repair problem

- **Input:** StarFDs Σ, and graph G does not satisfy Σ
- **Output:** a repair O, such that
  - obtain G' that satisfies Σ
  - $\mathrm{cost}(O) \leq \mathrm{cost}(O')$ for any O'

Isolated CCs have approximate solutions

- Solution overview
  - Connected components (CCs): inconsistencies connected at shared node attributes
  - Isolated CCs: no new inconsistency is introduced when a CC is repaired



Repair workflow

Is approximable?

Yes

No

Is optimal repairable?

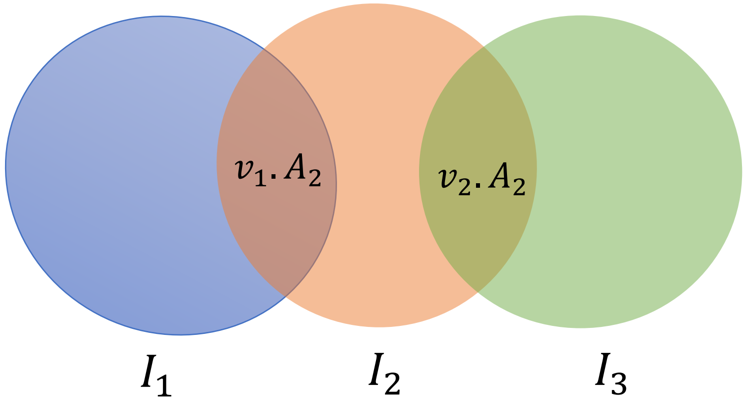
Heuristic solution

Yes

No

Optimal solution

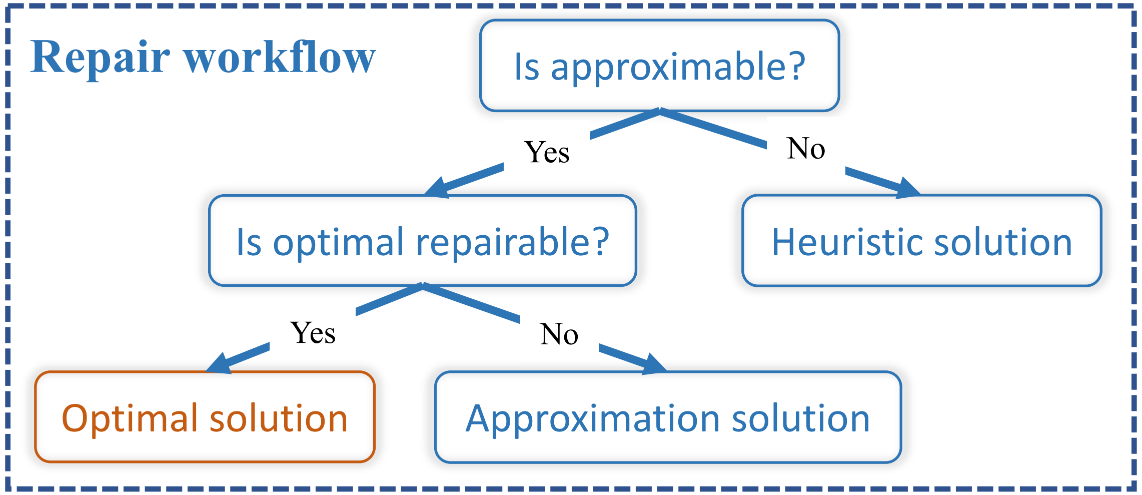
Approximation solution

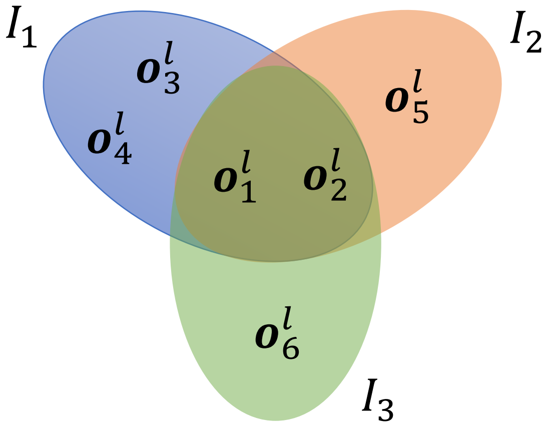


Optimal case

- Updates $\boldsymbol{o}^l$: flip the condition of a literal l in $X \cup Y$
- Optimal solution: hyper star structure
  - Select the $\boldsymbol{o}^*$ with least cost in center
  - Select one $\boldsymbol{o}$ with least cost in each petal, and induce $\boldsymbol{O}$
  - If cost($\boldsymbol{o}^*$) $\leq$ cost($\boldsymbol{O}$), return $\boldsymbol{o}^*$; otherwise, return $\boldsymbol{O}$

Example:
- $\boldsymbol{o}^* = \boldsymbol{o}_1^l$
- $\boldsymbol{O} = \boldsymbol{o}_3^l \cup \boldsymbol{o}_5^l \cup \boldsymbol{o}_6^l$
- Return $\boldsymbol{o}^*$ that has less cost





Optimal case

Approximable case

- Updates $\boldsymbol{o}^l$: flip the condition of a literal l in $X \cup Y$

- Approximation solution:
  - Hypergraph vertex cover without forbidden pairs
  - Forbidden pairs

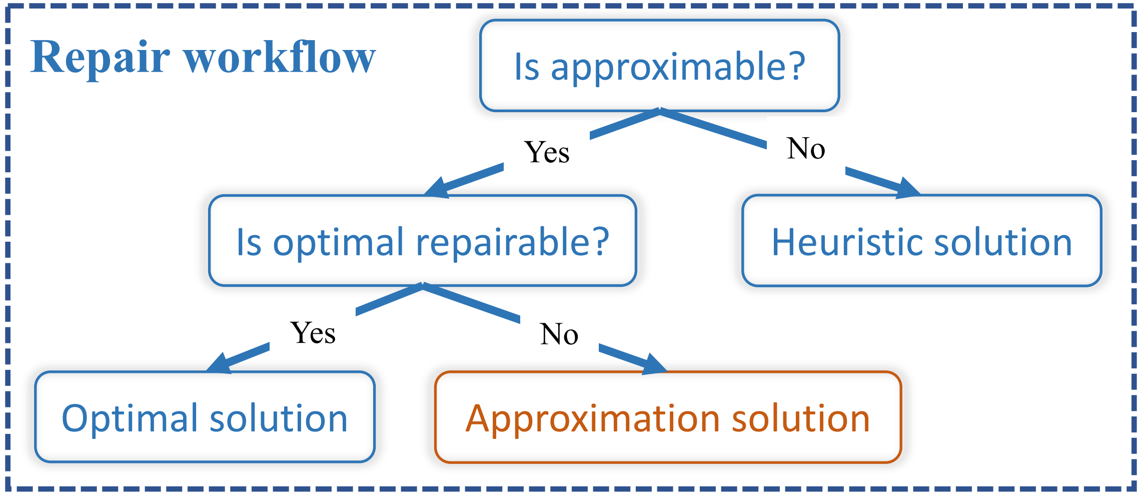
    $\boldsymbol{o}_5^l$ = {(v_2.owner, MUP, CFG), (v_4.owner, EM, ENIC)}

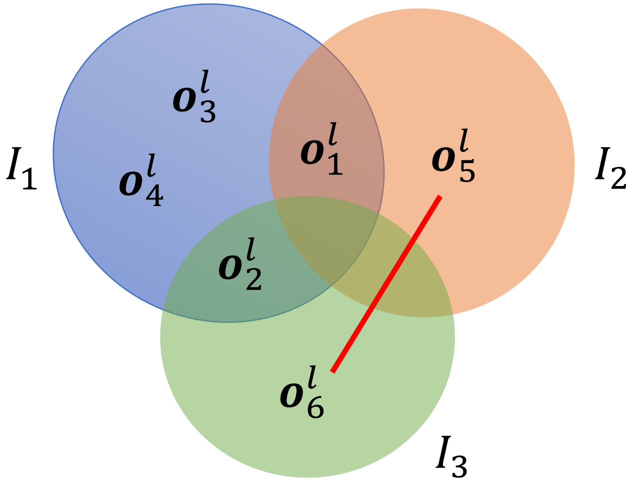
    $\boldsymbol{o}_6^l$ = {(v_2.owner, MUP, FSG), (v_4.owner, EM, ENIC)}

Example:

- Return $\boldsymbol{O} = \boldsymbol{o}_2^l \cup \boldsymbol{o}_5^l$

- $\boldsymbol{o}_6^l$ is pruned



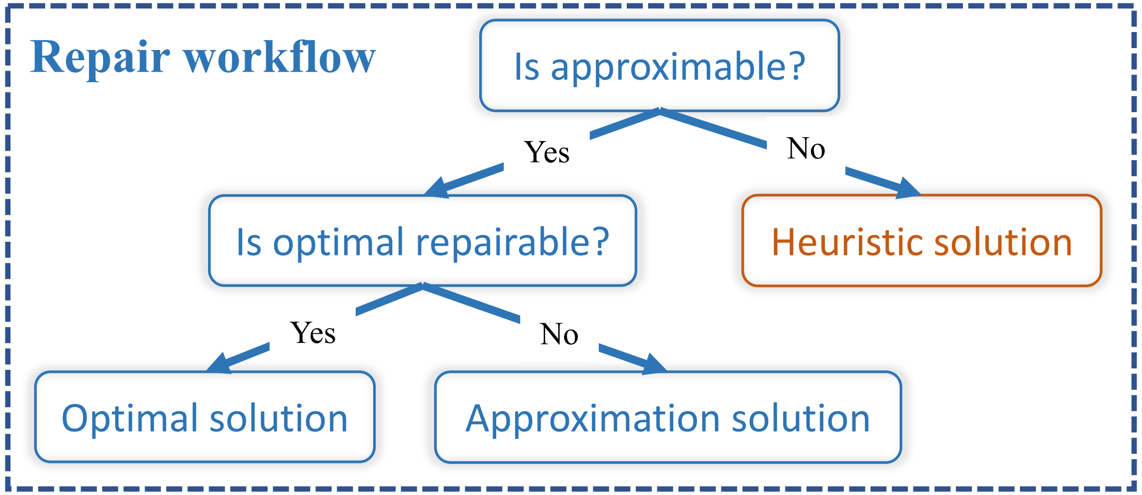


Approximable case

Heuristic case

- Updates $\boldsymbol{o}^l$: flip the condition of a literal l in $X \cup Y$

- Heuristic solution (for *non-isolated* CC):
  - Select CC introducing fewest inconsistencies
  - Invoke approximation/optimal solution
  - Re-detect inconsistencies
  - Repeat until incur a cost bound



Repair workflow

Is approximable?

Yes → Is optimal repairable?

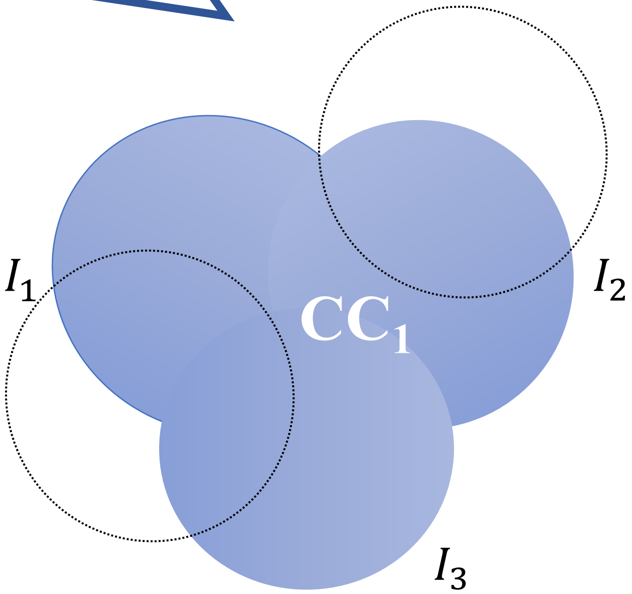
No → Heuristic solution

Is optimal repairable?

Yes → Optimal solution

No → Approximation solution

Repair CC_1 consisting of I_1, I_2, and I_3

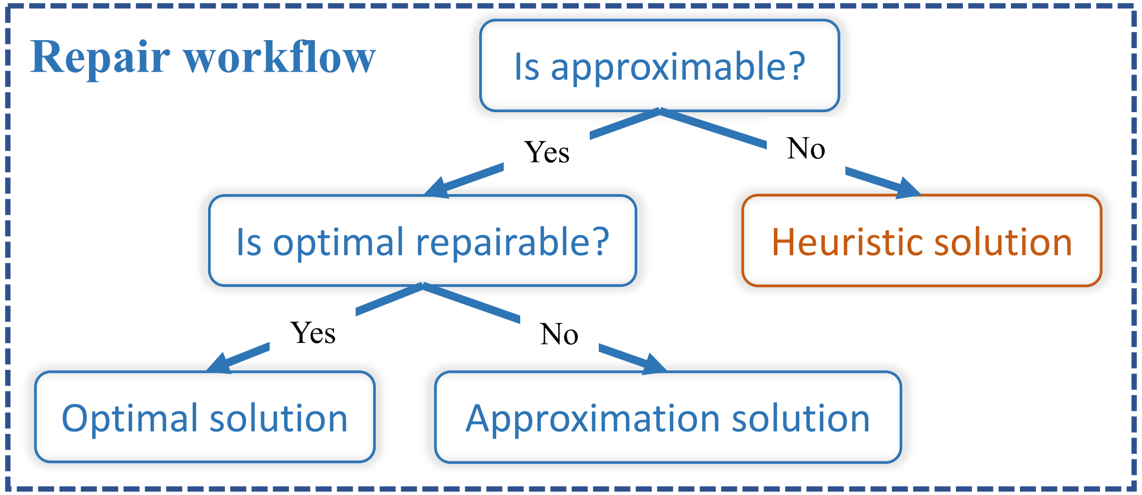


Heuristic case

Heuristic case

- Updates $\boldsymbol{o}^l$: flip the condition of a literal l in $X \cup Y$

- Heuristic solution (for *non-isolated* CC):
  - Select CC introducing fewest inconsistencies
  - Invoke approximation/optimal solution
  - Re-detect inconsistencies
  - Repeat until incur a cost bound



Repair workflow

Is approximable?

Yes

No

Is optimal repairable?

Heuristic solution

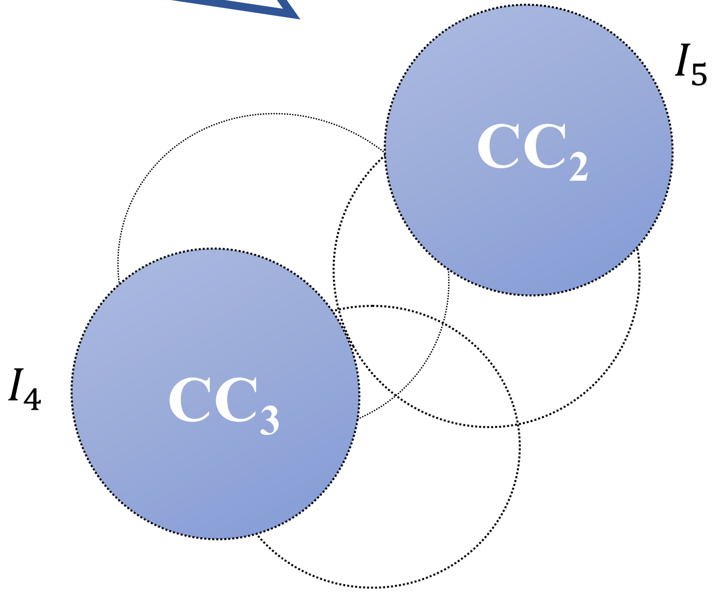
Yes

No

Optimal solution

Approximation solution

Two new inconsistencies I_4 and I_5



Heuristic case

Experiment settings

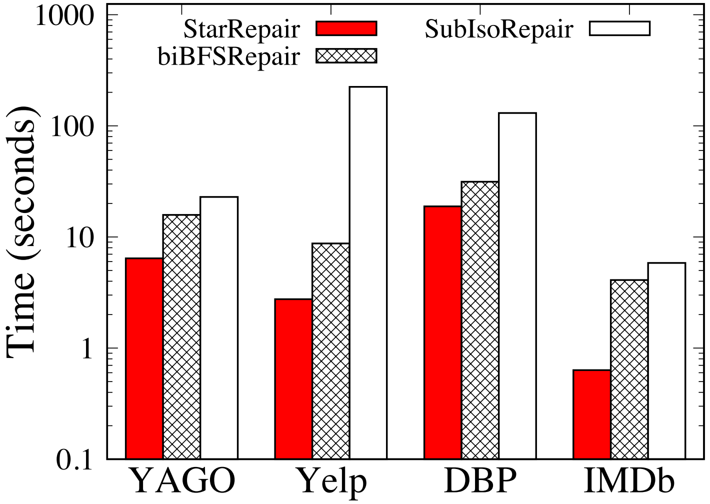
- Datasets

Data	Description	# of nodes	# of edges	avg. # of attributes per node
Yago	Knowledge graph	2.1M	4.0M	3
DBPedia	Knowledge graph	2.2M	7.4M	4
Yelp	Business reviews	1.5M	1.6M	5
IMDb	Movie network	5.9M	3.2M	3

- Error generation: adopt silver standard and an error generation benchmark (Arocena et al. 2015)
- StarFD generation: discovered from silver standard (first star patterns and then value constraints)
- Algorithms:
  - **StarRepair:** use bidirectional search for regular path queries with incremental error detection
  - **biBFSRepair:** use bidirectional search *without incremental error detection*
  - **SubIsoRepair:** *use subgraph isomorphism* as matching semantics with incremental error detection

Experiment results

- StarFD repairs: efficiency and effectiveness

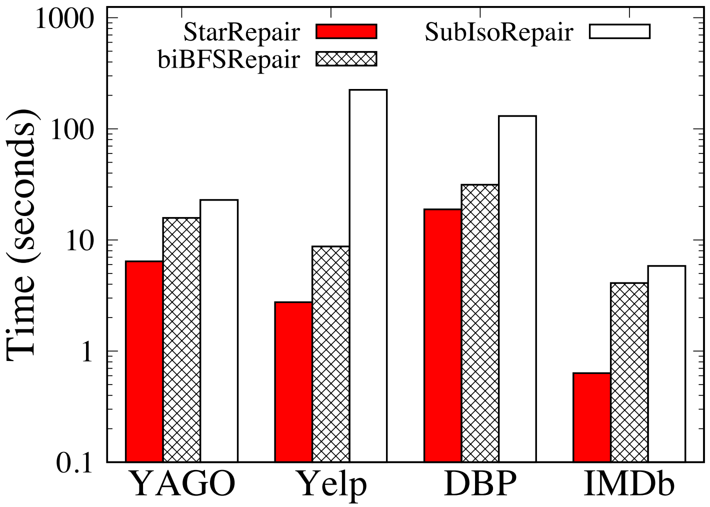


StarRepair outperforms biBFSRepair and SubIsoRepair by 3.4 and 7.1 times respectively

- Case study

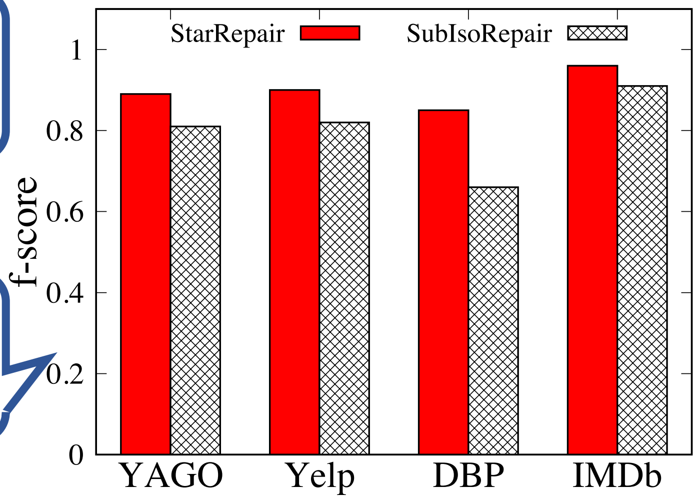
Experiment results

- StarFD repairs: efficiency and effectiveness



StarRepair outperforms biBFSRepair and SubIsoRepair by 3.4 and 7.1 times respectively

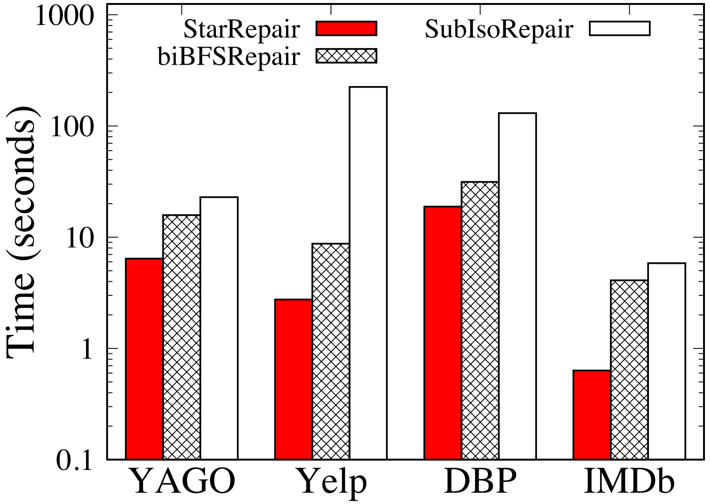
StarRepair outperforms SubIsoRepair by 10% in f-score (9% in precision and 14% in recall)



- Case study

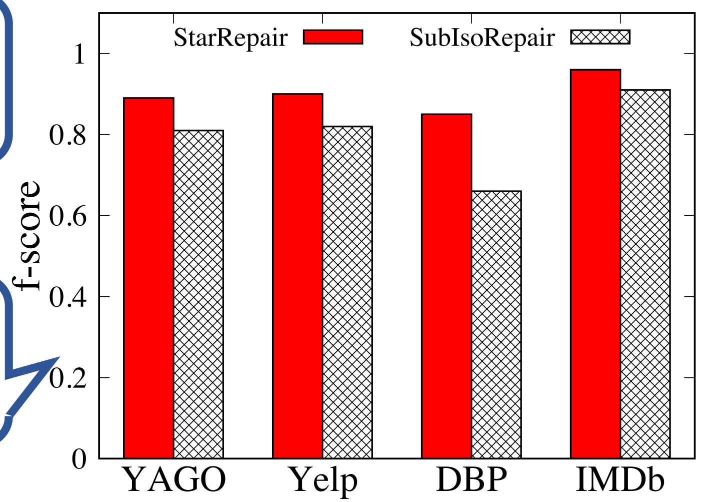
Experiment results

- StarFD repairs: efficiency and effectiveness



StarRepair outperforms biBFSRepair and SubIsoRepair by 3.4 and 7.1 times respectively

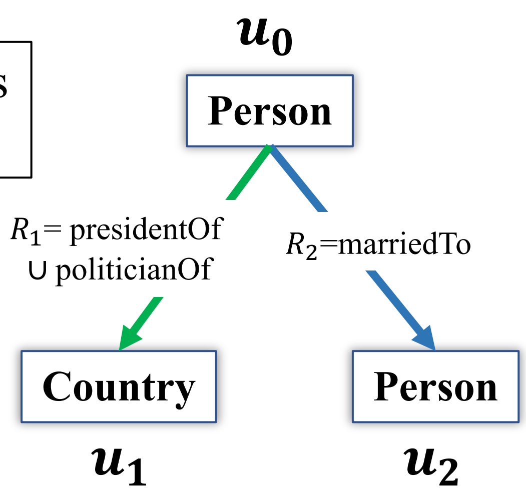
StarRepair outperforms SubIsoRepair by 10% in f-score (9% in precision and 14% in recall)

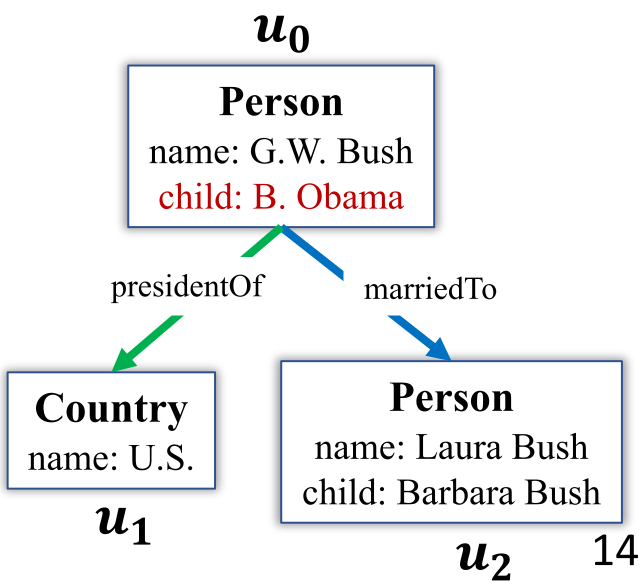


- Case study

StarFD: If a person u_0 is a politician or president of U.S., and is married to another person u_1, then u_1's child is u_0's child.

We found more than 100 such errors in Yago.





Compare with GFDs (Fan et al. 2016)

- StarFDs: star functional dependencies
  - Definition: $\varphi = (P(u_o), X \rightarrow Y)$

- GFDs: graph functional dependencies
  - Definition: $\varphi = (P, X \rightarrow Y)$

Problem	StarFDs	GFDs
Semantic	star patterns with regex queries	subgraph isomorphism
Satisfiability	NP-complete	coNP-complete
Implication	coNP-hard	NP-complete
Error detection (validation)	PTIME	coNP-complete

Repairing Entities using Star Constraints in Multi-relational Graphs

Problem	Description	Hardness	Solution
Satisfiability	**Input:** Σ decide whether there exists G that satisfies Σ	NP-complete	
Implication	**Input:** Σ and φ decide whether for all G satisfy Σ, they satisfy φ	coNP-hard	
Error detection (validation)	**Input:** G and Σ **Output:** all inconsistencies $\boldsymbol{I}$	PTIME	Evaluate regular path queries and validate values - time complexity: $O(\|\Sigma\|\|V\| + \|V\|(\|V\| + \|E\|))$
Repair	**Input:** Σ and G that does not satisfy Σ **Ouput:** G' that satisfies Σ with least repair cost	NP-hard APX-hard	Approximable cases (PTIME checkable) - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|\|\Sigma\|))$ - approximation ratio: $\|\boldsymbol{I}\|\|\Sigma\|^2$
			Optimal cases - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|)$
			Heuristic cases - time complexity $O(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|(\|\boldsymbol{I}\|\|\Sigma\|^2 + \|\boldsymbol{I}\|\|\Sigma\|))$ - bounded repairable: cost $\leq \|\boldsymbol{I}\|$

- Notations G: graph V: nodes E: edges
  Σ: a set of StarFDs φ: a single StarFD $\boldsymbol{I}$: all inconsistencies.

Thank you!

Kronos: Lightweight Knowledge-based Event Analysis in Cyber-Physical Data Streams
To appear in Demo Session