Logic on MARS: Ontologies for Generalised Property Graphs

Maximilian Marx Markus Krötzsch Veronika Thost

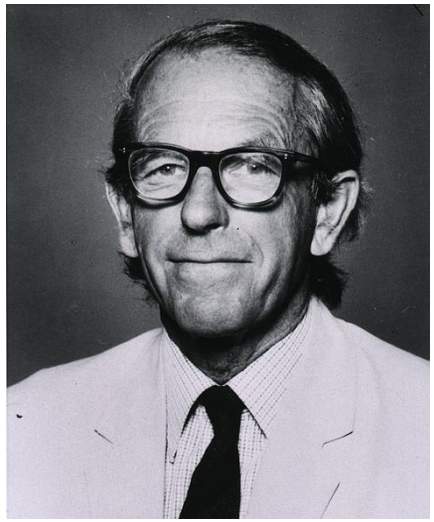
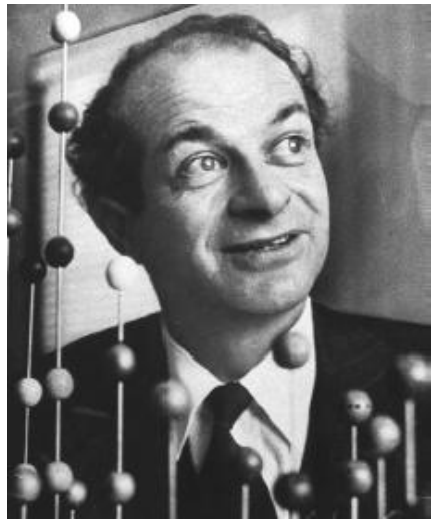
TU Dresden

IJCAI 2017

Full paper: `https://iccl.inf.tu-dresden.de/web/MARS/en`

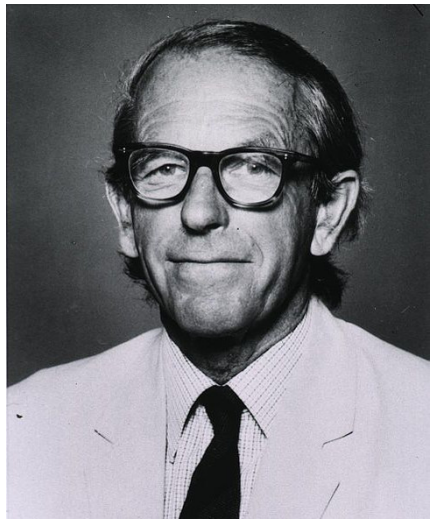
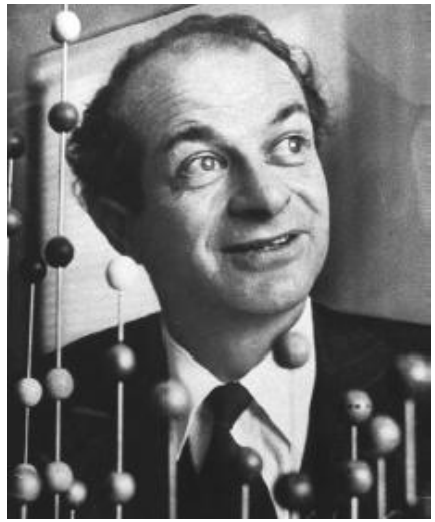
What do these people have in common?



What do these people have in common?

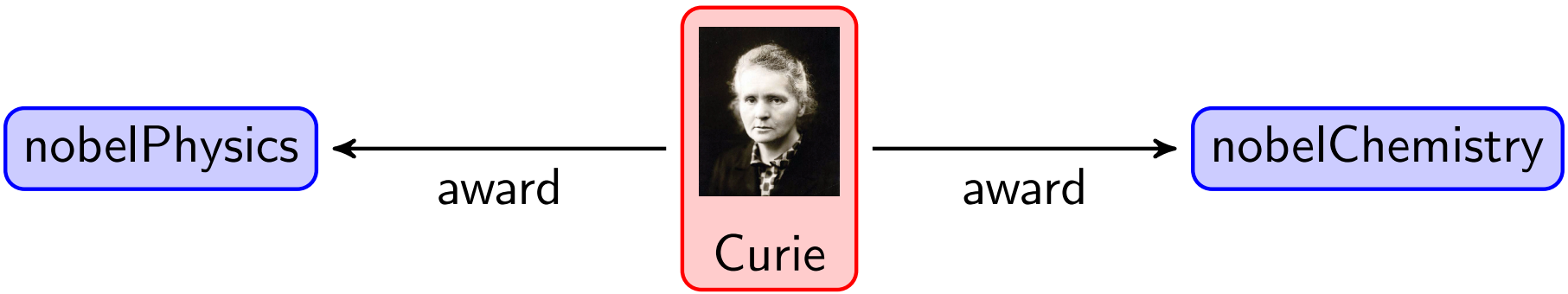


People that have won two Nobel prizes

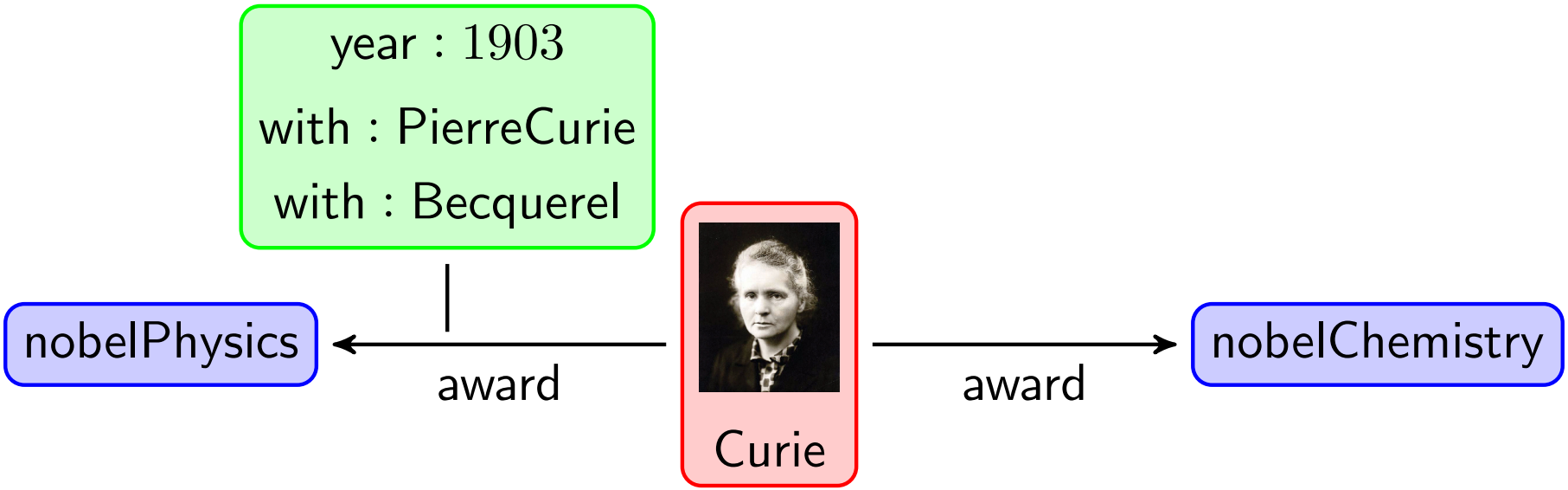
Laureates in Knowledge Graphs

Wikidata: a free and open Knowledge Graph



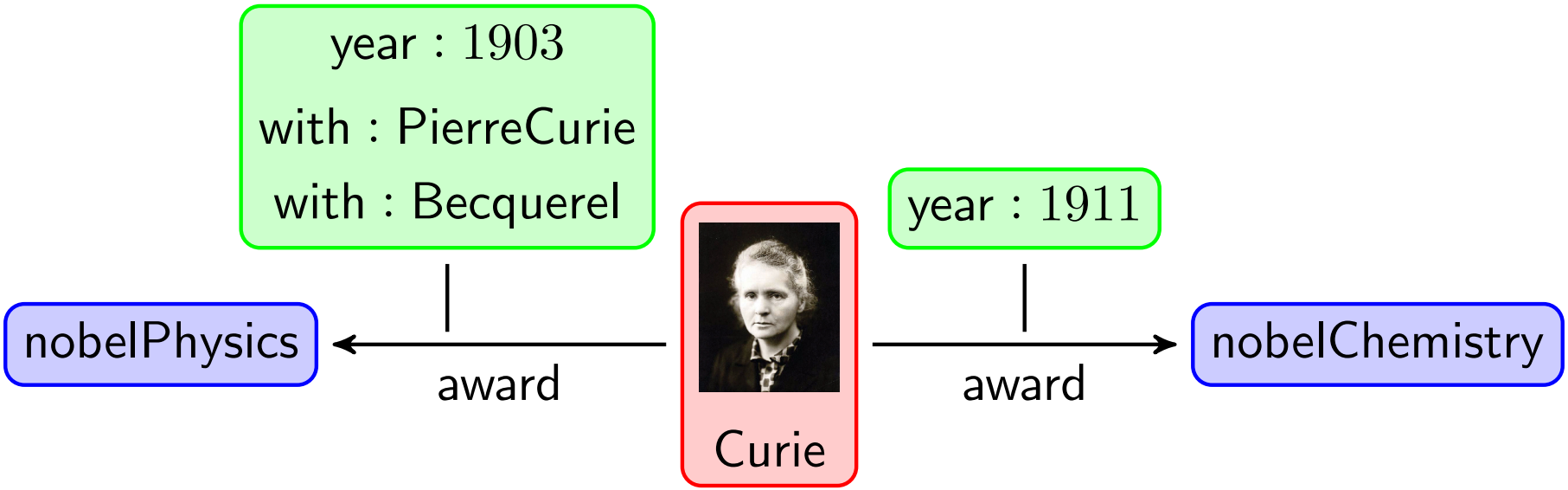
Laureates in Knowledge Graphs

Wikidata: a free and open Knowledge Graph



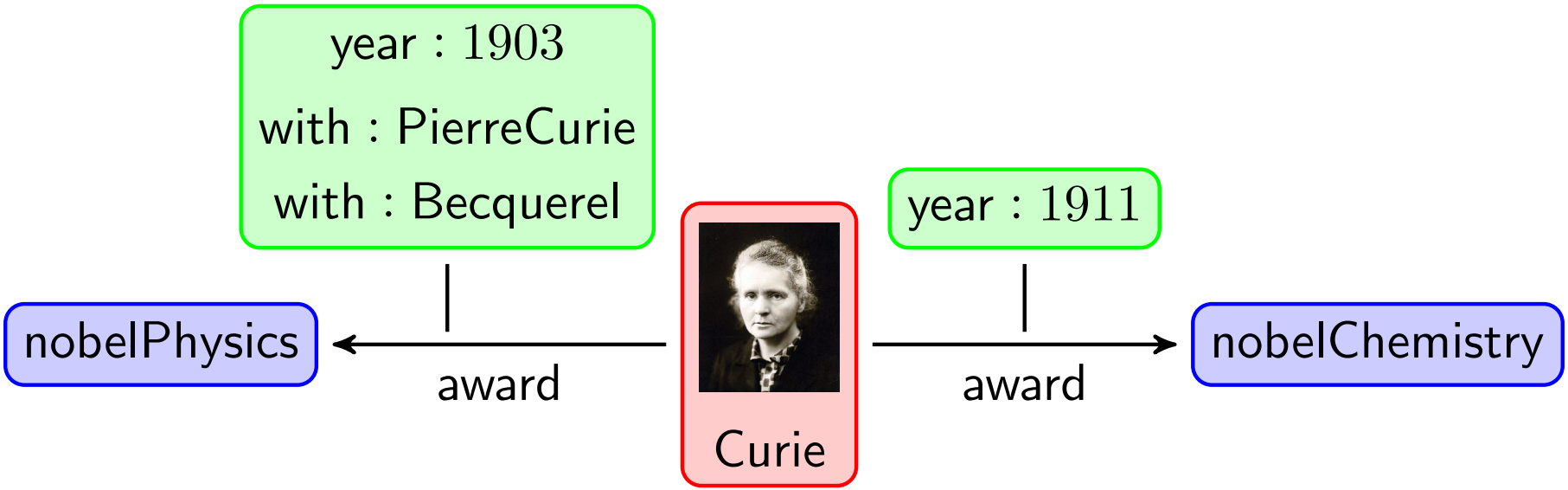
Laureates in Knowledge Graphs

Wikidata: a free and open Knowledge Graph



Laureates in Knowledge Graphs

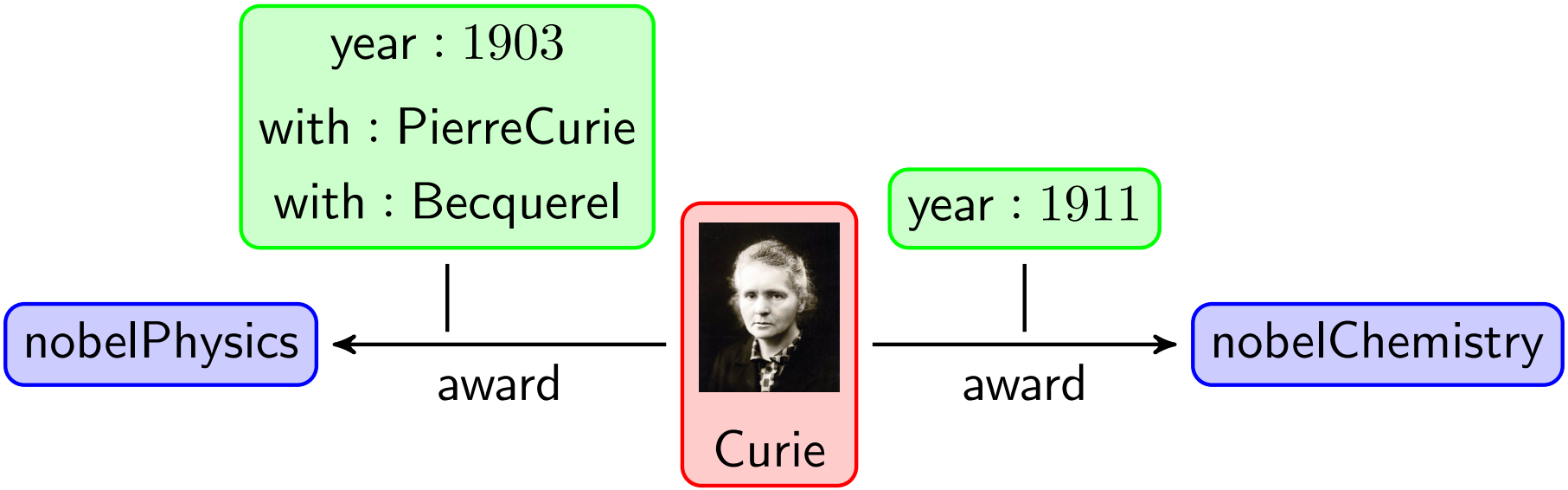
Wikidata: a free and open Knowledge Graph



fact notation: $\mathsf{award}(\mathsf{Curie}, \mathsf{nobelChemistry})@\{\mathsf{year} : 1911\}$

Laureates in Knowledge Graphs

Wikidata: a free and open Knowledge Graph

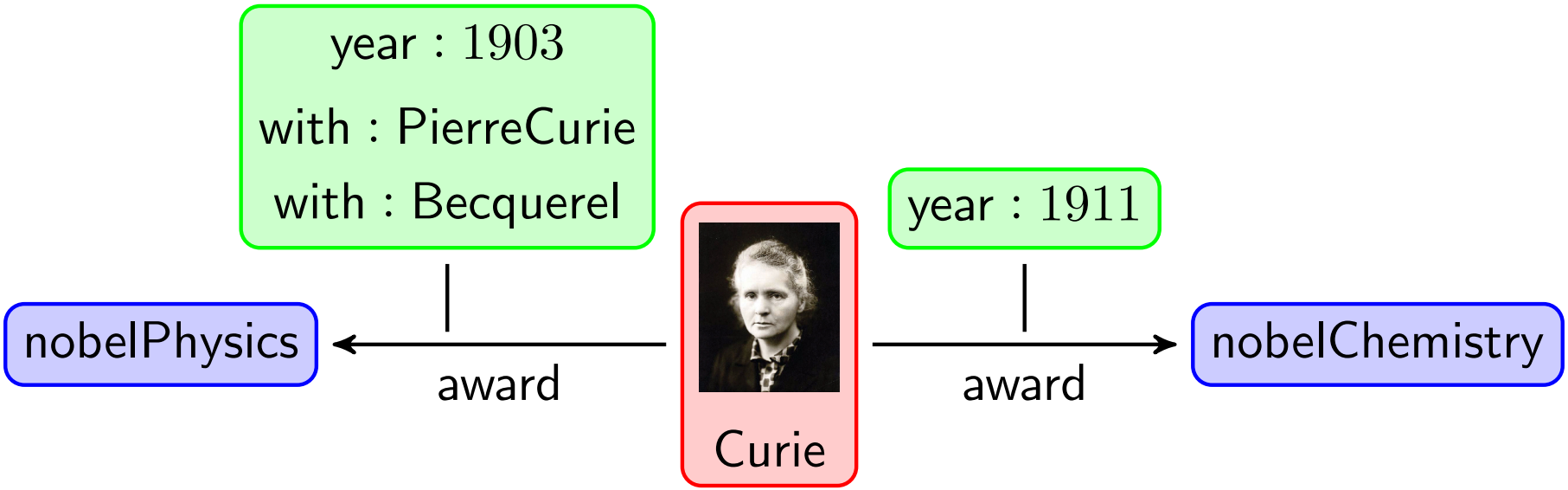


fact notation: $\mathrm{award}(\mathrm{Curie}, \mathrm{nobelChemistry})@\{\mathrm{year} : 1911\}$

Knowledge Graph: labeled graph; edges carry **annotation sets** (finite sets of **attribute–value** pairs)

Laureates in Knowledge Graphs

Wikidata: a free and open Knowledge Graph



fact notation: $\mathrm{award}(\mathrm{Curie}, \mathrm{nobelChemistry})@\{\mathrm{year} : 1911\}$

Knowledge Graph: labeled graph; edges carry **annotation sets** (finite sets of **attribute–value** pairs)

MARS: multi-attributed relational structure (annotated hypergraph)

Annotation-aware reasoning

Example: for sole "award" winners, infer "laureate," i.e.,

$$\text{from} \qquad \text{award}(\text{Curie}, \text{nobelChemistry})@\{\text{year} : 1911\}$$

Annotation-aware reasoning

Example: for sole "award" winners, infer "laureate," i.e.,

$$\begin{aligned} \text{from} \quad & \text{award}(\text{Curie}, \text{nobelChemistry})@\{\text{year} : 1911\}, \\ \text{infer} \quad & \text{laureate}(\text{nobelChemistry}, \text{Curie})@\{\text{year} : 1911\} \end{aligned}$$

Annotation-aware reasoning

Example: for sole "award" winners, infer "laureate," i.e.,

$$\begin{aligned} \text{from} \quad & \mathsf{award}(\mathsf{Curie}, \mathsf{nobelChemistry})@\{\mathsf{year} : 1911\}, \\ \text{infer} \quad & \mathsf{laureate}(\mathsf{nobelChemistry}, \mathsf{Curie})@\{\mathsf{year} : 1911\} \end{aligned}$$

$\rightsquigarrow$ **Rule**:

$$\forall x, y. \forall S.\ \mathsf{award}(x, y)@S \wedge (\lfloor\rfloor \setminus \lfloor \mathsf{with} : + \rfloor)(S) \rightarrow \mathsf{laureate}(y, x)@S$$

Annotation-aware reasoning

Example: for sole "award" winners, infer "laureate," i.e.,

$$\begin{aligned} \text{from} \quad & \mathsf{award}(\mathsf{Curie}, \mathsf{nobelChemistry})@\{\mathsf{year} : 1911\}, \\ \text{infer} \quad & \mathsf{laureate}(\mathsf{nobelChemistry}, \mathsf{Curie})@\{\mathsf{year} : 1911\} \end{aligned}$$

$\rightsquigarrow$ **Rule**:

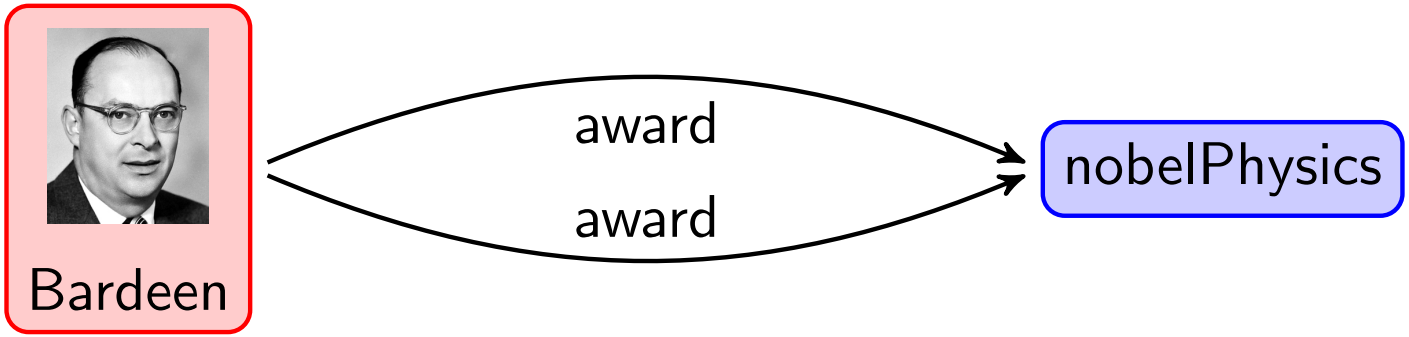
$$\forall x, y. \forall S.\ \mathsf{award}(x, y)@S \wedge (\lfloor\rfloor \setminus \lfloor \mathsf{with} : + \rfloor)(S) \rightarrow \mathsf{laureate}(y, x)@S$$

Specifier $(\lfloor\rfloor \setminus \lfloor \mathsf{with} : + \rfloor)$: any annotation set without attribute "with"

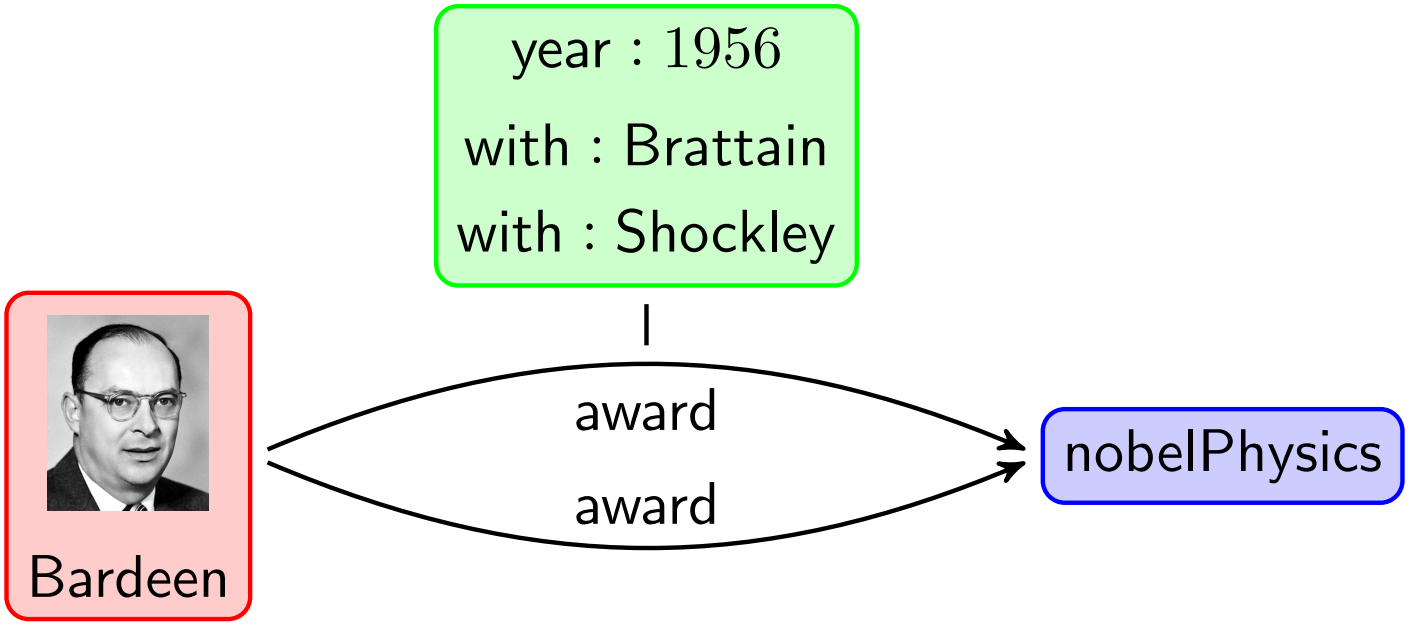
✓ $\{\mathsf{year} : 1911\}$

✗ $\{\mathsf{year} : 1903, \mathsf{with} : \mathsf{PierreCurie}, \mathsf{with} : \mathsf{Becquerel}\}$

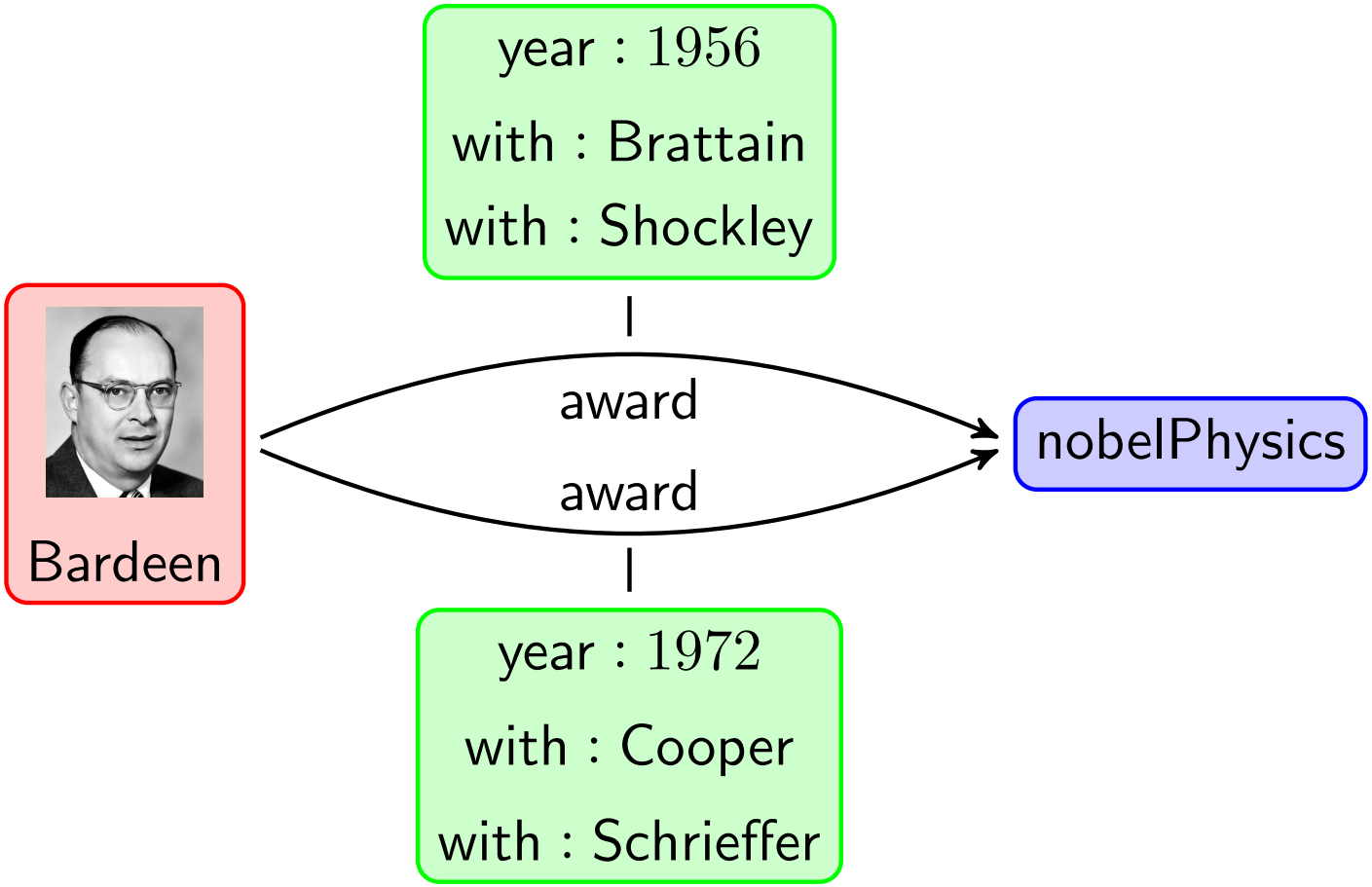
What else do we need?



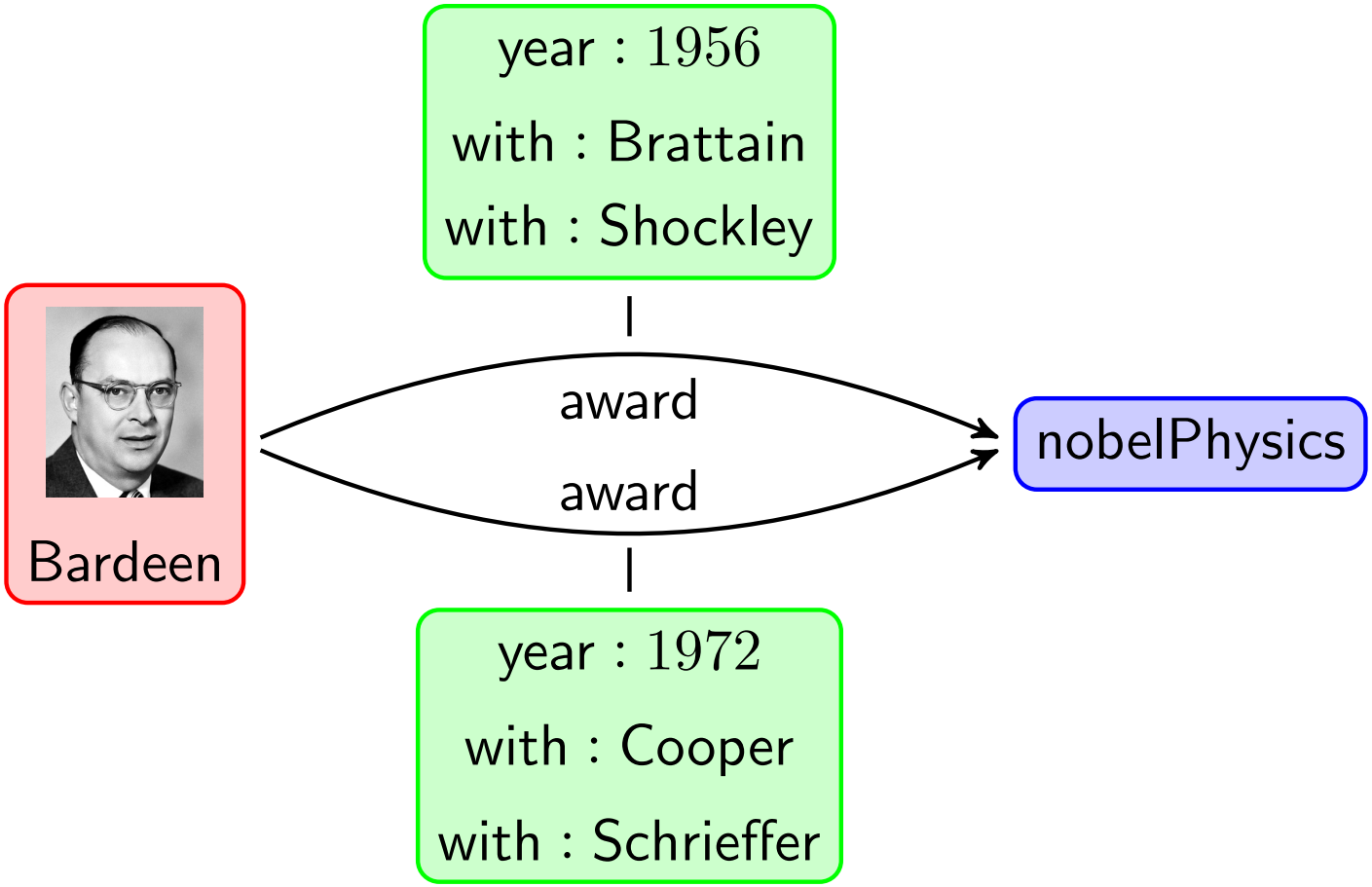
What else do we need?



What else do we need?



What else do we need?



Goal: derive "award" for co-laureates $\rightsquigarrow$ copying annotation sets is not enough, we need to compute a new annotation set

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

MAPL: multi-attributed predicate logic

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

MAPL: multi-attributed predicate logic

- enrich edges with finite binary relations (annotation sets)

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

MAPL: multi-attributed predicate logic

- enrich edges with finite binary relations (annotation sets)
- quantification over annotation sets

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

MAPL: multi-attributed predicate logic

- enrich edges with finite binary relations (annotation sets)
- quantification over annotation sets

Theorem

MAPL has the same expressivity as weak second-order logic.
$\rightsquigarrow$ *Entailment for MAPL theories is not semi-decidable.*

A Logic for Knowledge Graphs

Goal: annotation-aware reasoning

- derived knowledge may depend on annotation sets
- compute new annotation sets

MAPL: multi-attributed predicate logic

- enrich edges with finite binary relations (annotation sets)
- quantification over annotation sets

Theorem

MAPL has the same expressivity as weak second-order logic.
$\rightsquigarrow$ *Entailment for MAPL theories is not semi-decidable.*

Idea: Encode arbitrary arity predicates in annotation sets

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets
- **Function definitions** derive new annotation sets

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets
- **Function definitions** derive new annotation sets

Example: from

$$\mathsf{award}(\mathsf{Bardeen}, \mathsf{nobelPhysics})@\{\mathsf{year} : 1956,\ \mathsf{with} : \mathsf{Shockley},\ \mathsf{with} : \mathsf{Brattain}\}$$

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets
- **Function definitions** derive new annotation sets

Example: from

$$\mathsf{award}(\mathsf{Bardeen}, \mathsf{nobelPhysics})@\{\mathsf{year} : 1956,\ \mathsf{with} : \mathsf{Shockley},\ \mathsf{with} : \mathsf{Brattain}\}$$

infer

$$\mathsf{award}(\mathsf{Shockley}, \mathsf{nobelPhysics})@\{\mathsf{year} : 1956,\ \mathsf{with} : \mathsf{Bardeen},\ \mathsf{with} : \mathsf{Brattain}\}$$

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets
- **Function definitions** derive new annotation sets

Example: from

$\mathsf{award}(\mathsf{Bardeen}, \mathsf{nobelPhysics})@\{\mathsf{year} : 1956,\ \mathsf{with} : \mathsf{Shockley},\ \mathsf{with} : \mathsf{Brattain}\}$

infer

$\mathsf{award}(\mathsf{Shockley}, \mathsf{nobelPhysics})@\{\mathsf{year} : 1956,\ \mathsf{with} : \mathsf{Bardeen},\ \mathsf{with} : \mathsf{Brattain}\}$

$\rightsquigarrow$ **Rule**

$\mathsf{award}(x, y)@S \wedge \lfloor \mathsf{with} : z \rfloor(S) \rightarrow \mathsf{award}(z, y)@\mathbf{CoLaureate}(S, x, z)$

MAPL Rules

A decidable fragment

- **Specifiers** express constraints on annotation sets
- **Function definitions** derive new annotation sets

Example: from

$$\mathsf{award}(\mathsf{Bardeen}, \mathsf{nobelPhysics})@\{\mathsf{year}:1956,\ \mathsf{with}:\mathsf{Shockley},\ \mathsf{with}:\mathsf{Brattain}\}$$

infer

$$\mathsf{award}(\mathsf{Shockley}, \mathsf{nobelPhysics})@\{\mathsf{year}:1956,\ \mathsf{with}:\mathsf{Bardeen},\ \mathsf{with}:\mathsf{Brattain}\}$$

$\rightsquigarrow$ **Rule**

$$\mathsf{award}(x, y)@S \wedge \lfloor \mathsf{with} : z \rfloor(S) \rightarrow \mathsf{award}(z, y)@\mathbf{CoLaureate}(S, x, z)$$

with function definition $\mathbf{CoLaureate}(U, v, w)$:

$$\Rightarrow \mathrm{insert}(\mathsf{with} : v)$$

$$\lfloor \mathsf{with} : o \rfloor(U), o \not\approx w \Rightarrow \mathrm{insert}(\mathsf{with} : o)$$

$$\lfloor \mathsf{year} : d \rfloor(U) \Rightarrow \mathrm{insert}(\mathsf{year} : d)$$

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**
- Idea: never touch existing annotation sets, only derive new ones

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**
- Idea: never touch existing annotation sets, only derive new ones
- Function definitions are evaluated during rule application

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**
- Idea: never touch existing annotation sets, only derive new ones
- Function definitions are evaluated during rule application
- exponentially many possible annotation sets ensure termination

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**
- Idea: never touch existing annotation sets, only derive new ones
- Function definitions are evaluated during rule application
- exponentially many possible annotation sets ensure termination

Theorem

MARPL entailment is ExpTime*-complete for data & combined complexity.*

Reasoning in MARPL

Overview

- Bottom-up materialisation: **MARS chase**
- Idea: never touch existing annotation sets, only derive new ones
- Function definitions are evaluated during rule application
- exponentially many possible annotation sets ensure termination

Theorem

MARPL entailment is ExpTime*-complete for data & combined complexity.*

Theorem

MARPL entailment is PTime*-complete for data complexity if the size of annotation sets is bounded.*

Conclusion & Outlook

Summary:

MAPL general, second-order based framework for attributed logics; not semi-decidable

MARPL decidable, rule-shaped fragment; ExpTime-complete for data & combined complexity

Future Work:

- Create attributed ontologies, e.g., for Wikidata
- Implement a MARPL reasoner
- Identify more expressive decidable fragments of MAPL
- Study attributed versions of other KR formalisms
- Classify data complexities, identify tractable fragments