A logical approach to model interpretability

Marcelo Arenas

PUC & IMFD Chile and RelationalAI, Berkeley

Joint work with Daniel Báez, Pablo Barceló, Diego Bustamante, José Thomas Caraball, Jorge Pérez, and Bernardo Subercaseaux

Motivation

- A growing interest in developing methods to explain predictions made by machine learning models

- This has led to the development of several notions of explanation

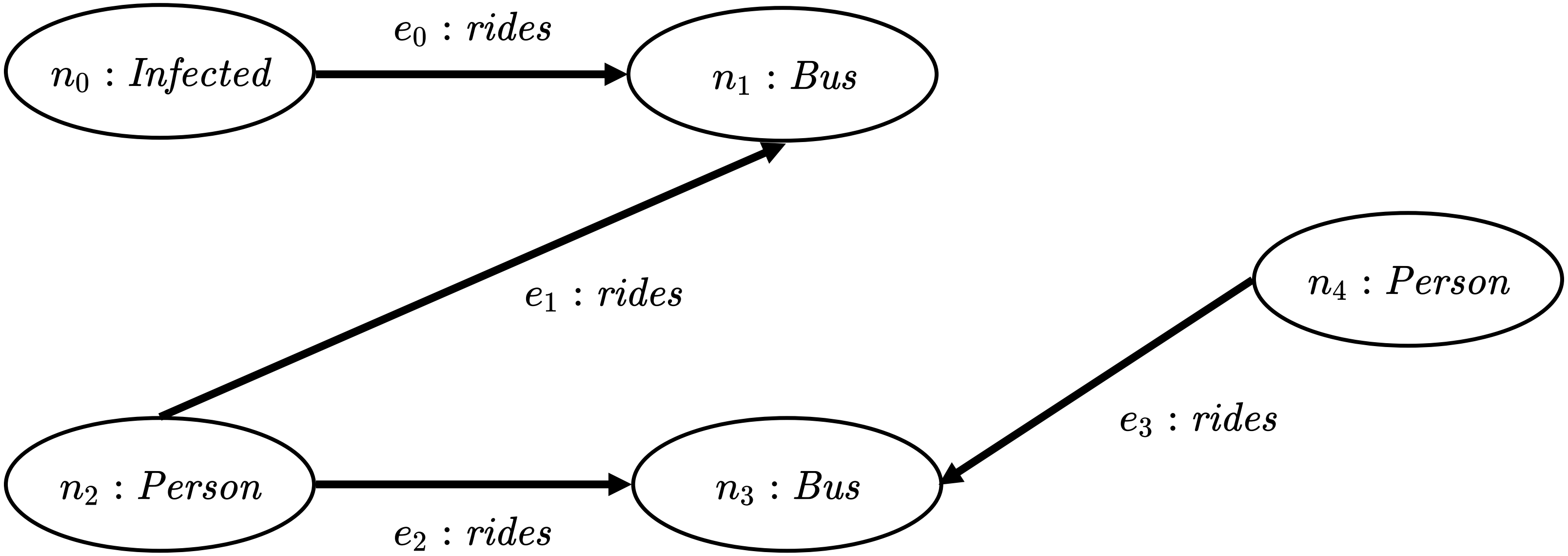
- Instead of struggling with the increasing number of such notions, one can developed a declarative query language for interpretability task

The goal of this talk

To show how such a framework can be developed by interpreting classification models as **labeled graphs**, and by using **first-order logic** as a query language

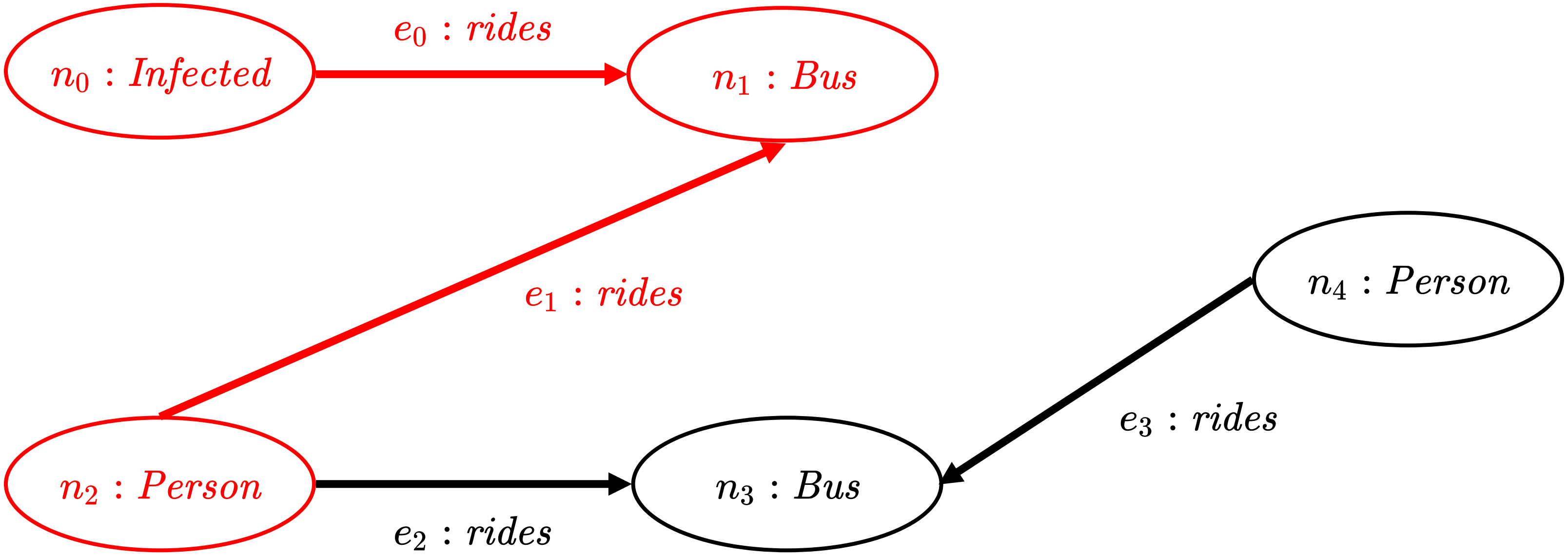
Extracting nodes from a graph: regular paths queries

$$Person/rides/Bus/rides^{-}/Infected$$



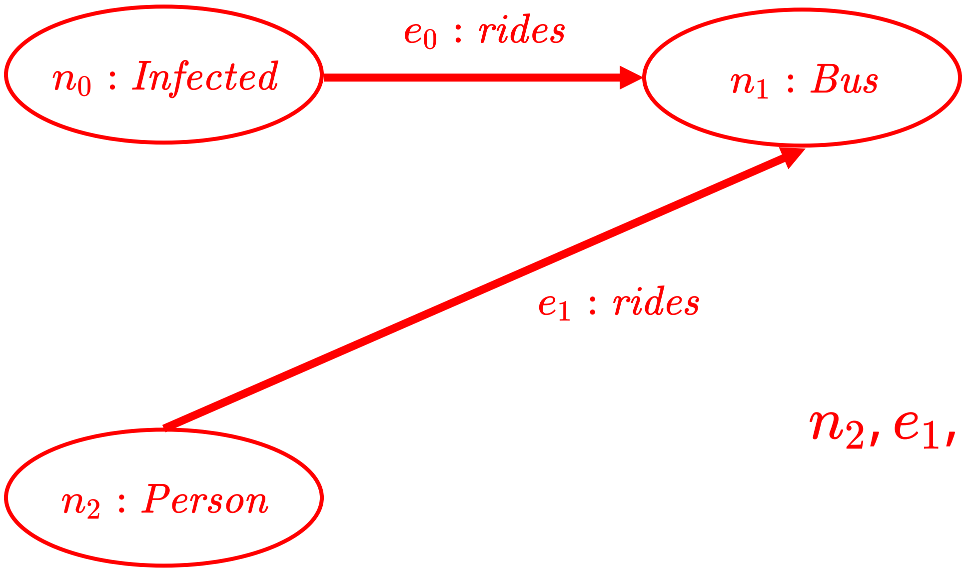
Extracting nodes from a graph: regular paths queries

$$Person/rides/Bus/rides^{-}/Infected$$



Extracting nodes from a graph: regular paths queries

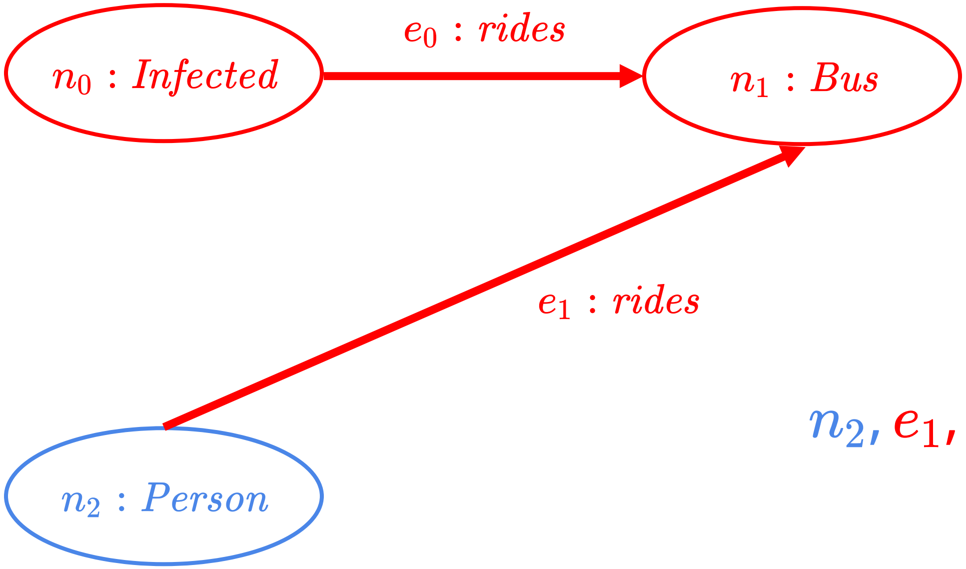
$$Person/rides/Bus/rides^{-}/Infected$$



n_2, e_1, n_1, e_0, n_0

Extracting nodes from a graph: regular paths queries

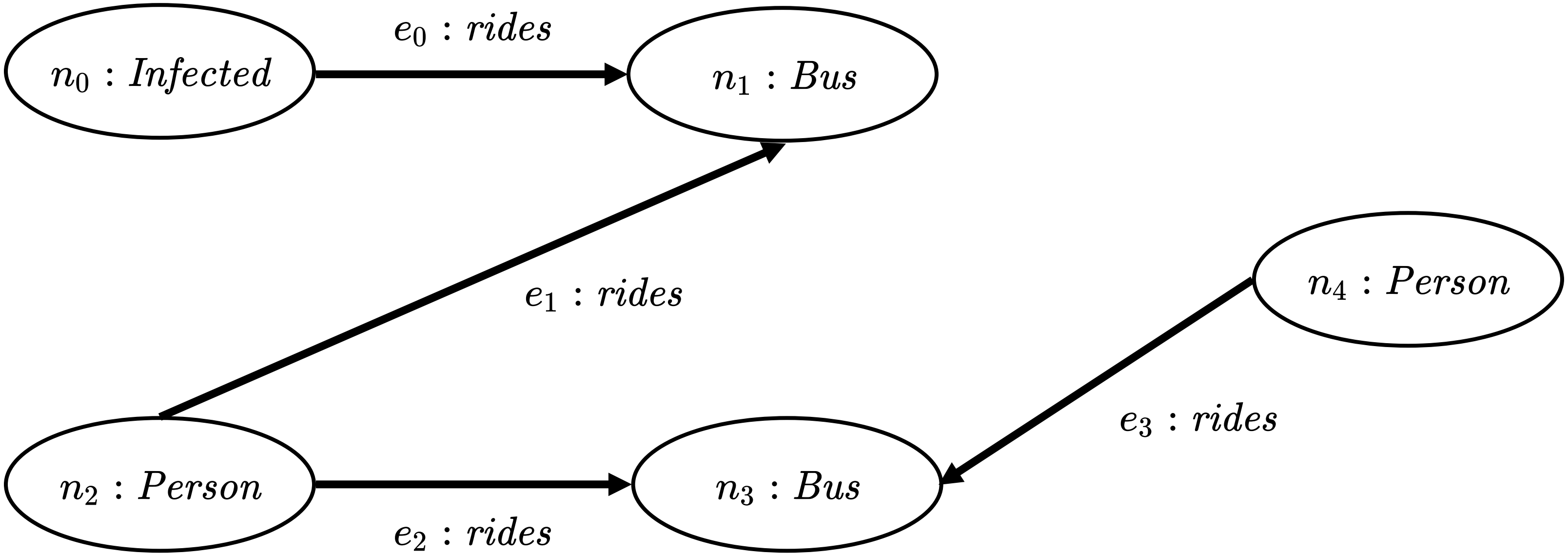
$$Person/rides/Bus/rides^{-}/Infected$$



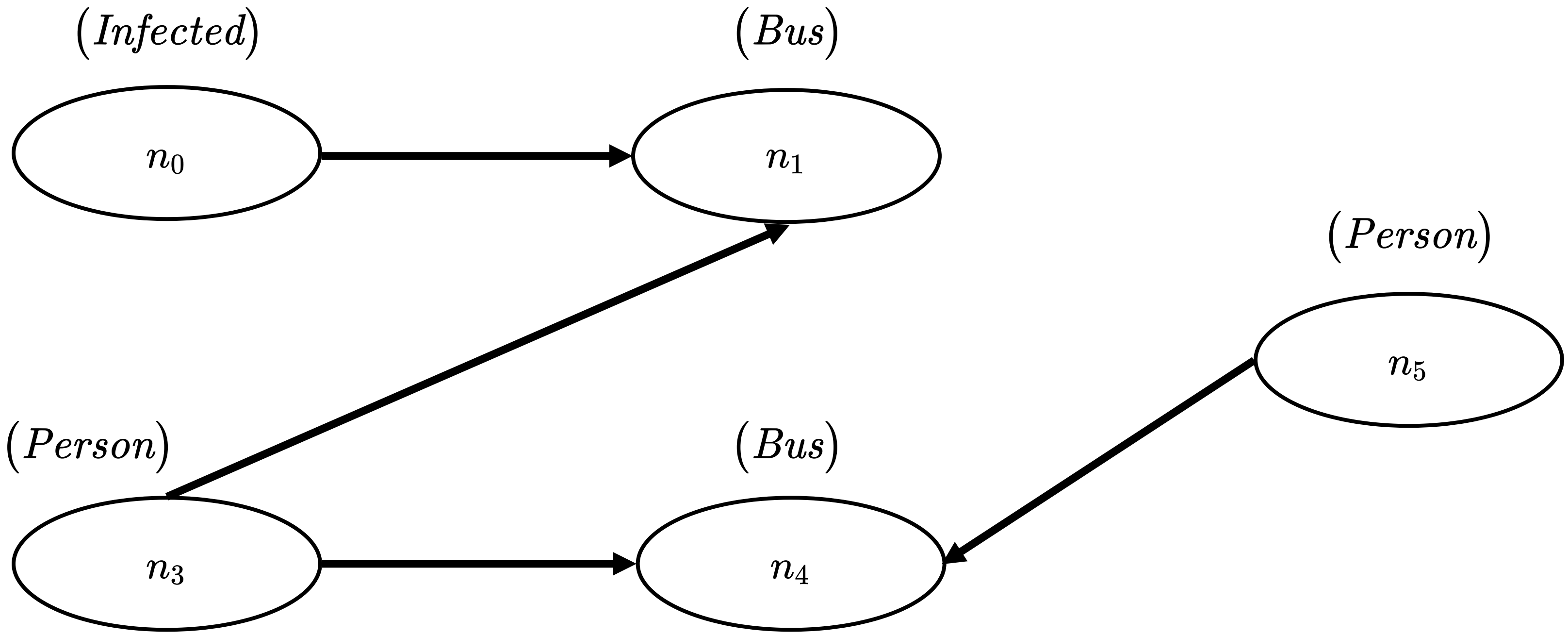
$$n_2, e_1, n_1, e_0, n_0$$

Extracting nodes from a graph: first-order logic

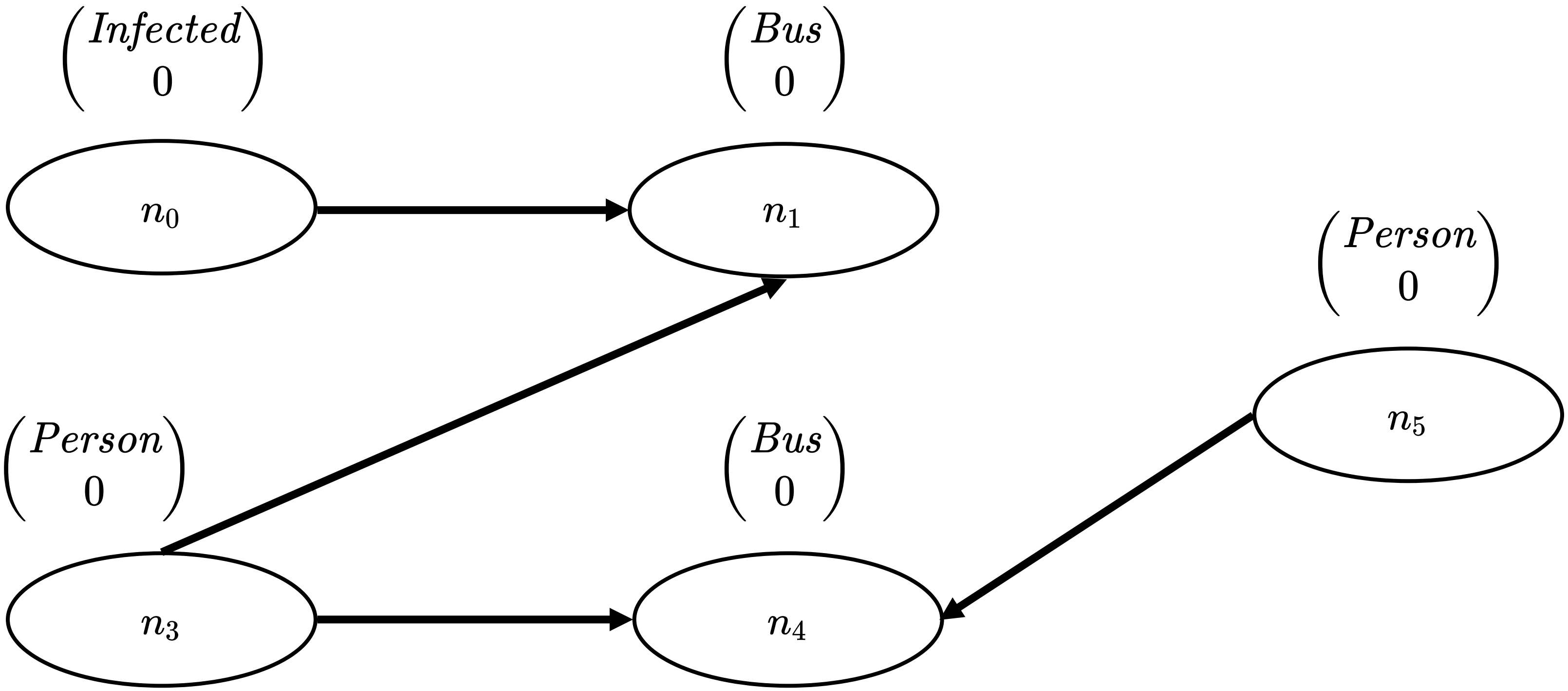
$$Person(x) \land \exists y\, (rides(x, y) \land Bus(y) \land \exists z\, (rides(z, y) \land Infected(z)))$$



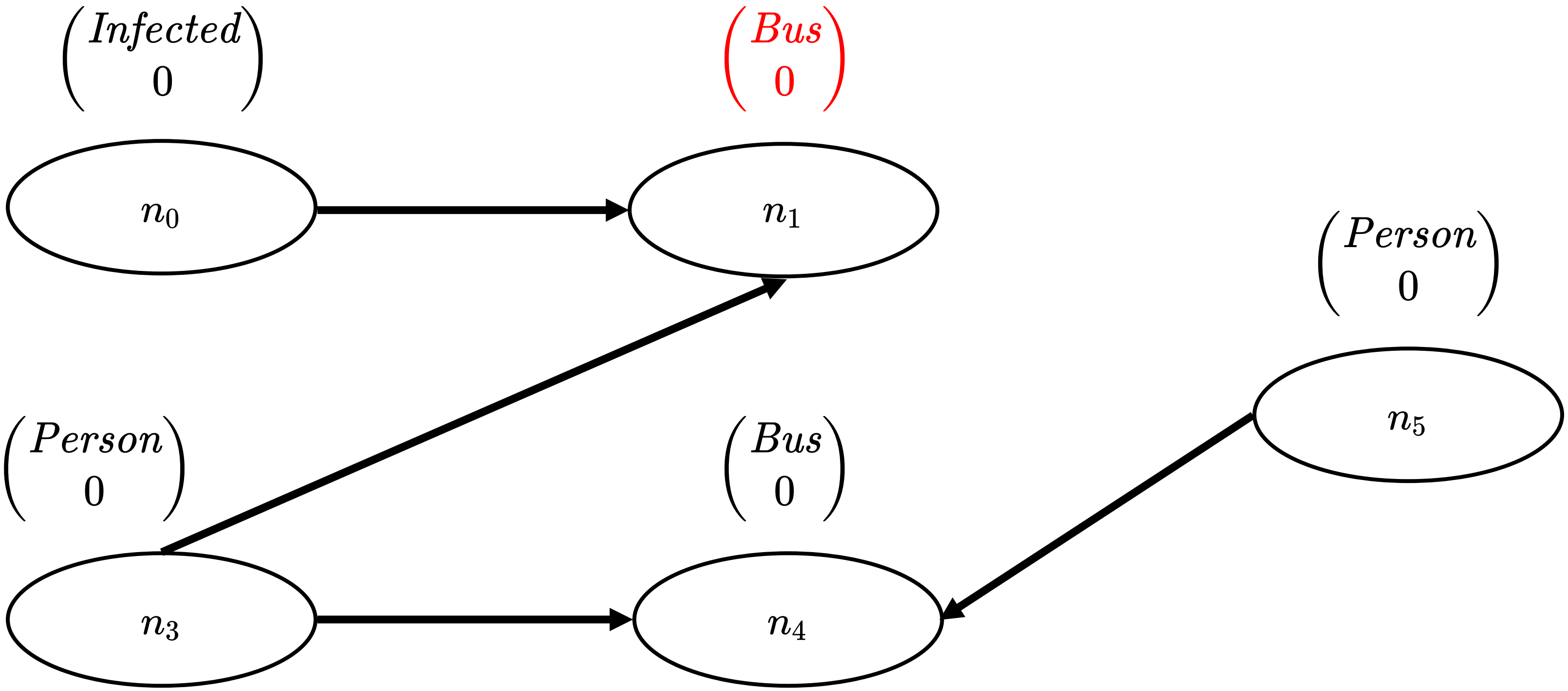
Extracting nodes from a graph: graph neural networks



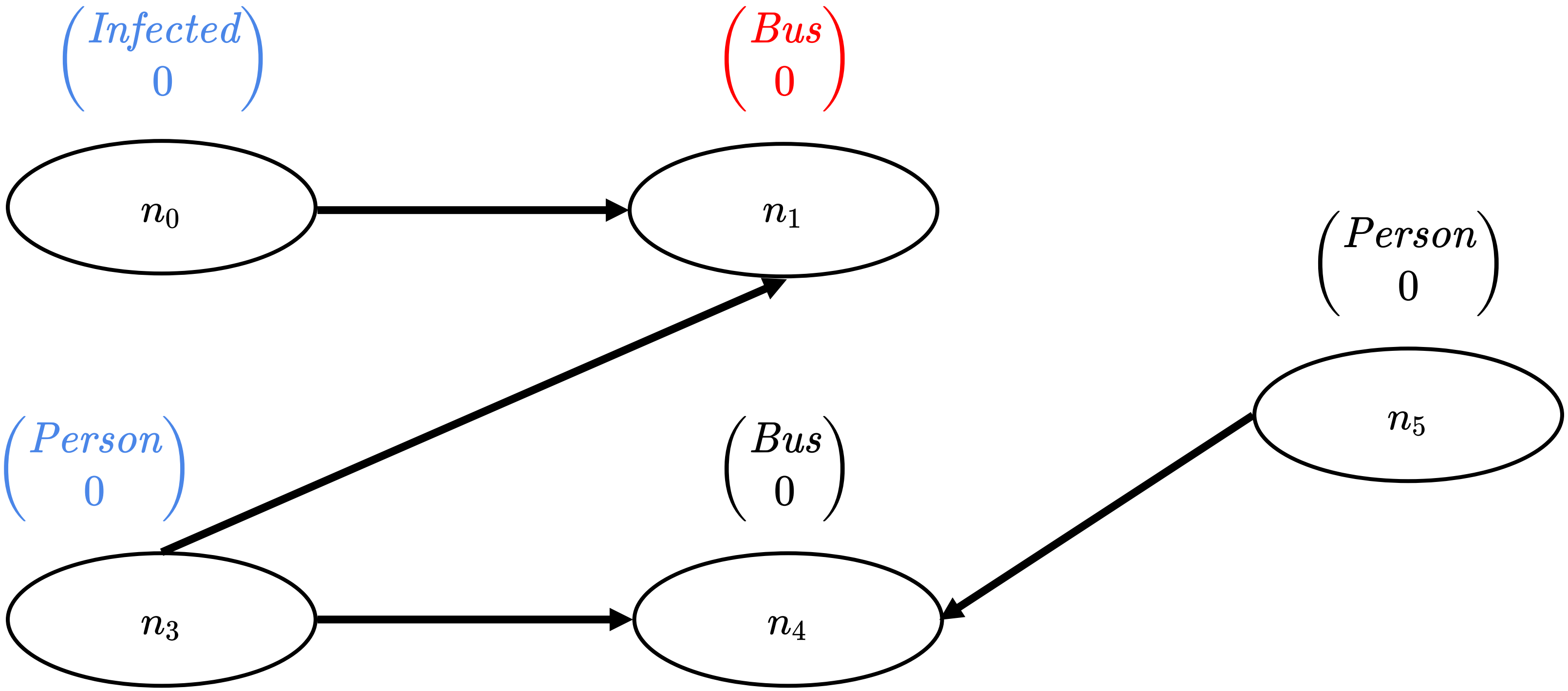
Processing by layers in GNNs: the input



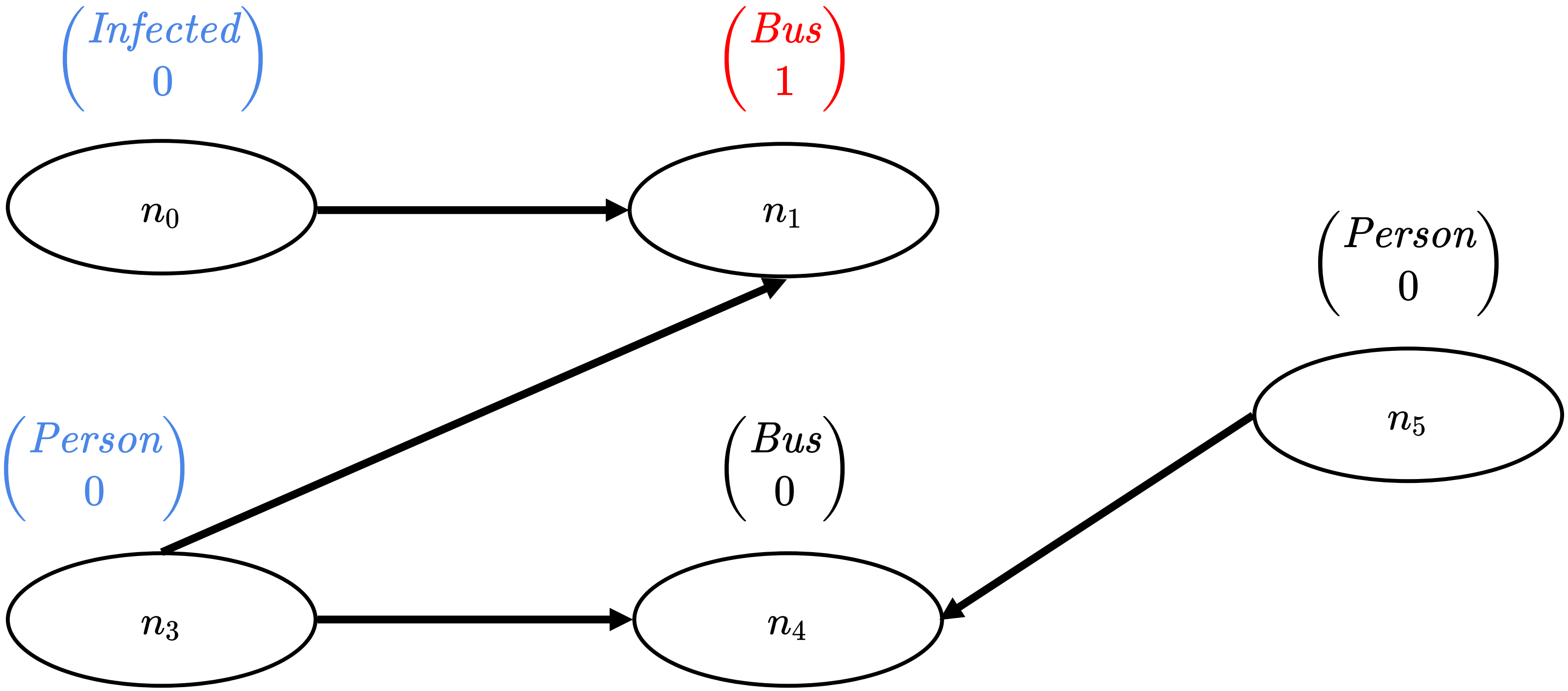
Computing the first layer



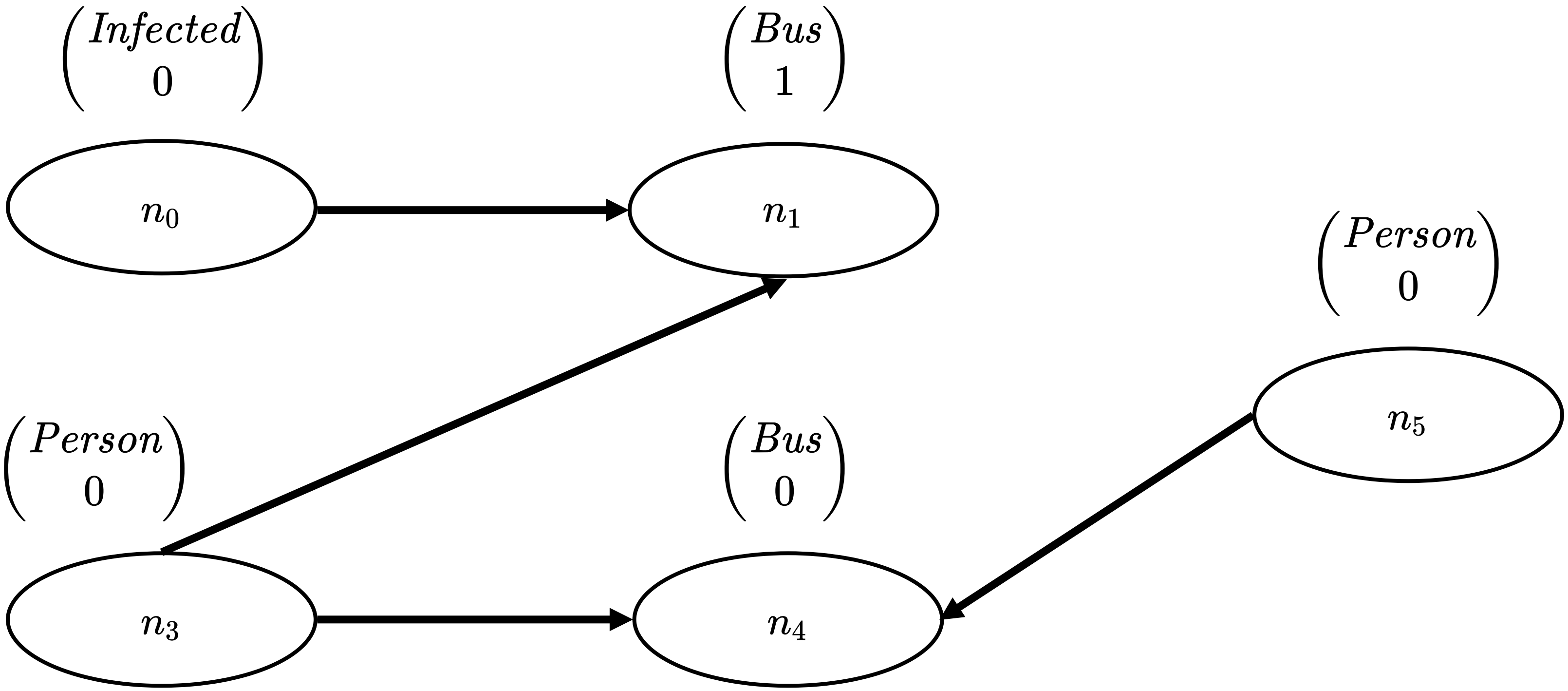
Computing the first layer



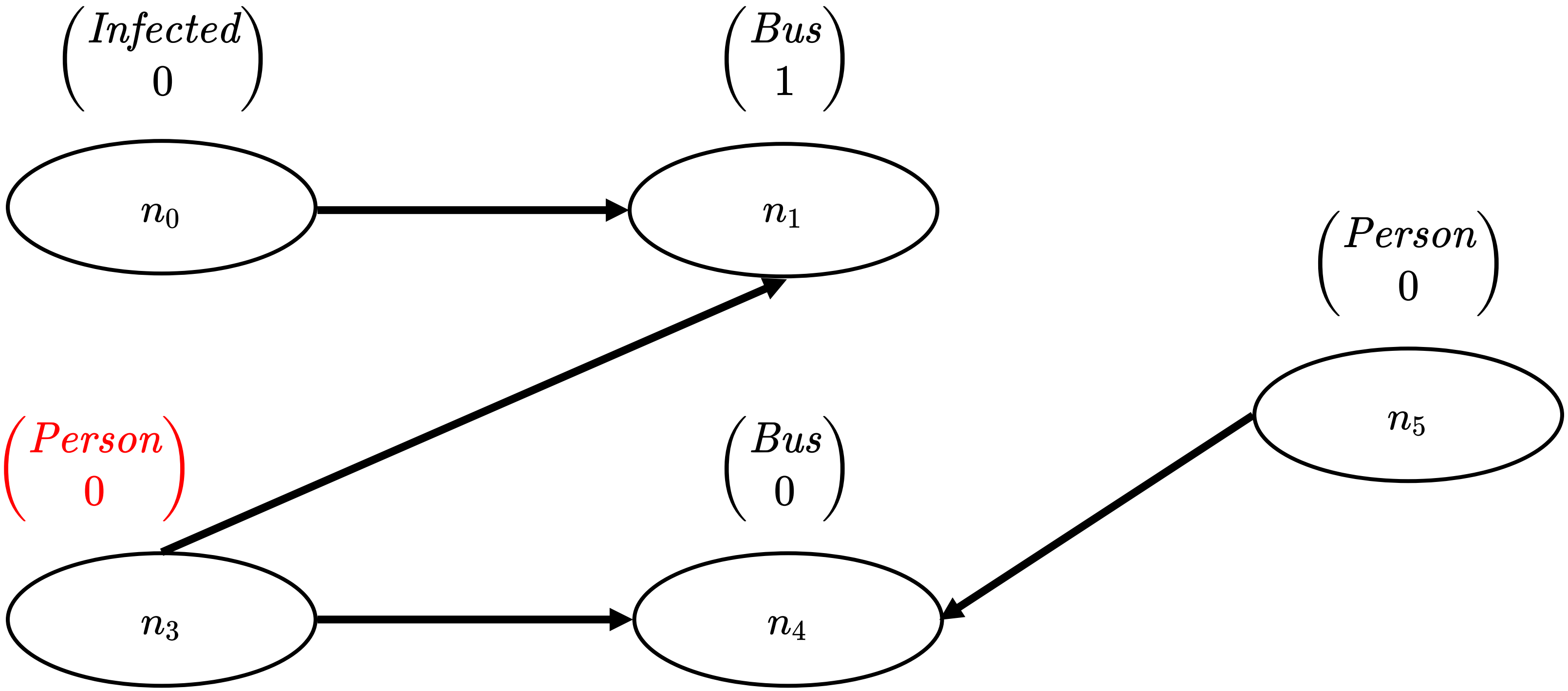
Computing the first layer



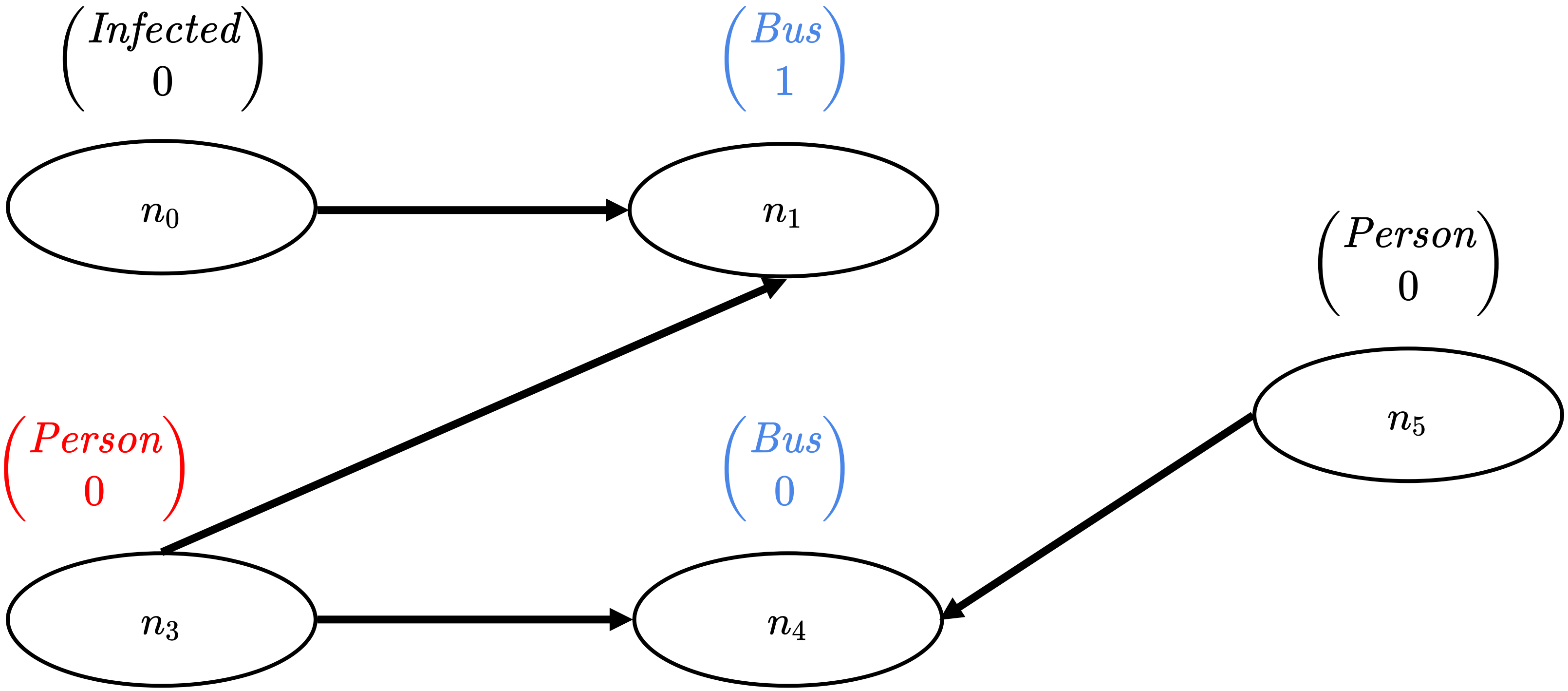
The result of the first layer



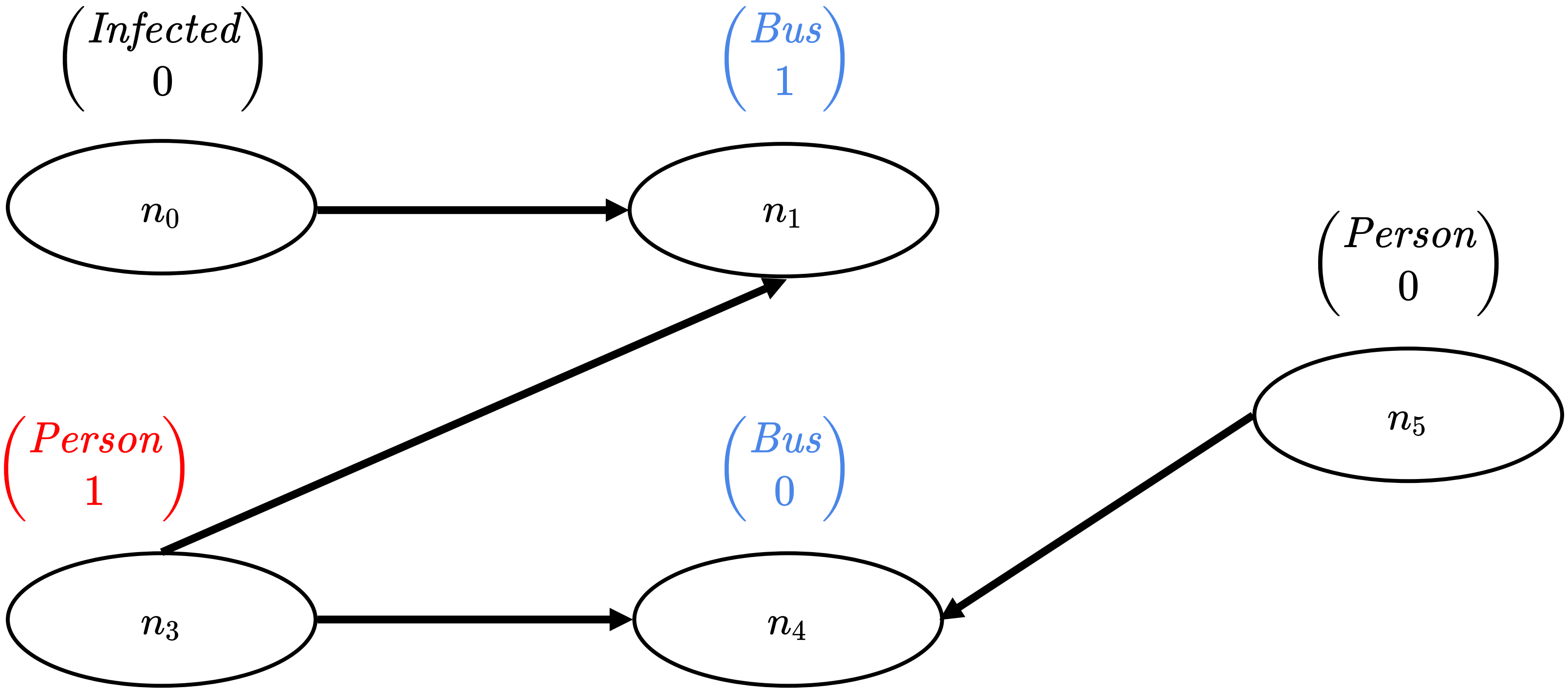
Computing the second layer



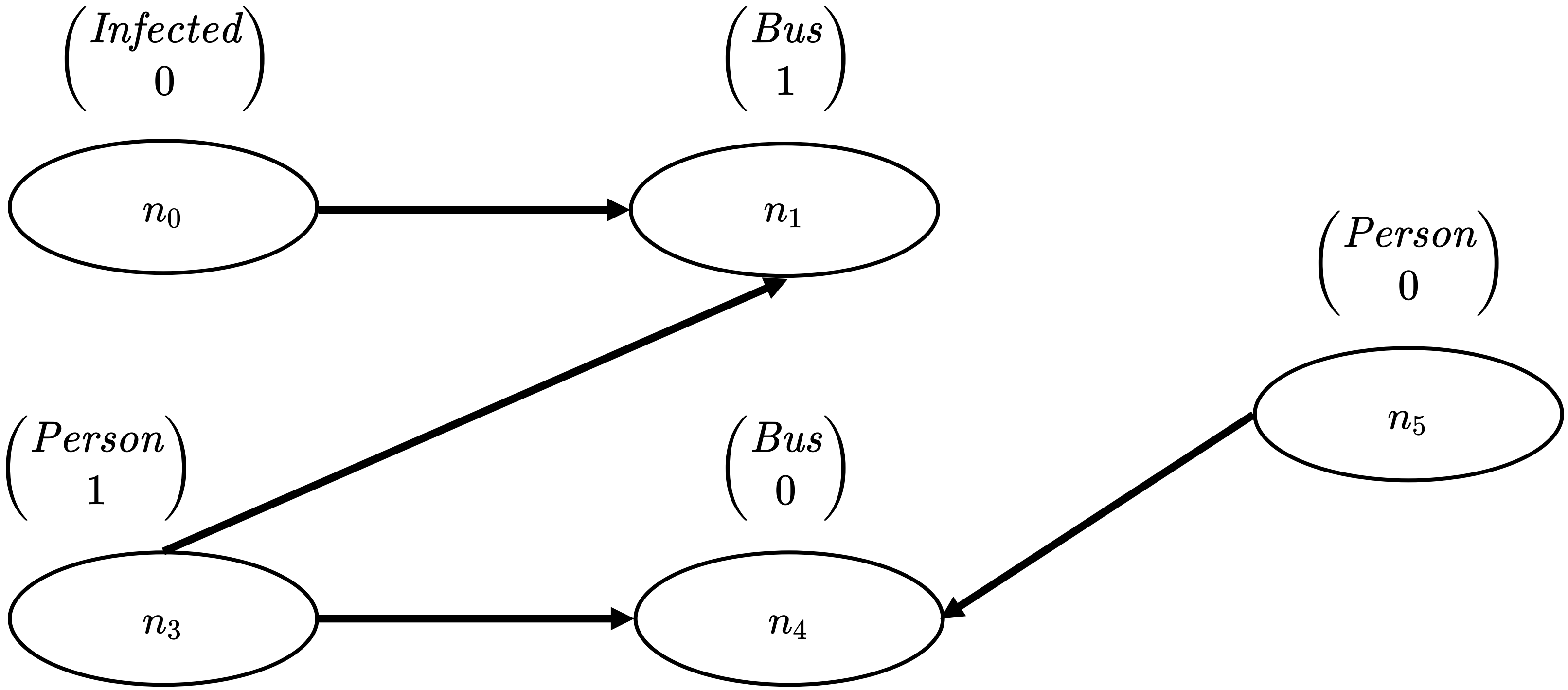
Computing the second layer



Computing the second layer



The result of the second layer



The architecture of GNNs

$u^{(i)}$: vector of features of node u at layer i

- $u^{(0)}$: vector of features from the input graph

$$u^{(i)} = \mathrm{COMB}^{(i)}\Big(u^{(i-1)}, \mathrm{AGG}^{(i)}\big(\{\!\{v^{(i-1)} \mid u \text{ and } v \text{ are neighbors in } G\}\!\}\big)\Big)$$

If k is the last layer: $\mathrm{CSL}(u^{(k)})$ is the result for node u

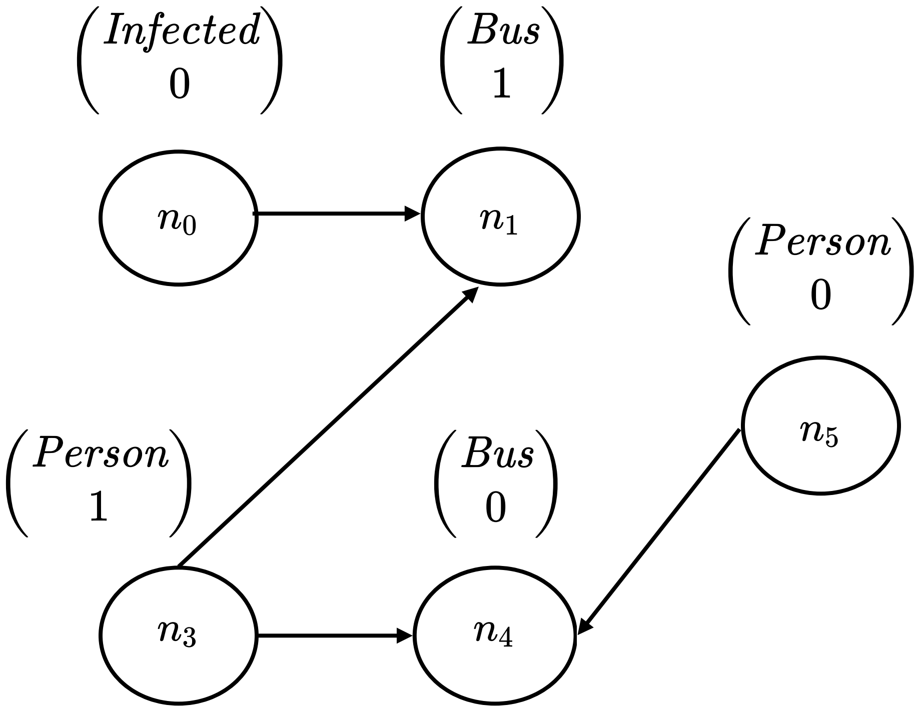
The architecture of GNNs

$$Person/rides/Bus/rides^{-}/Infected$$

$$\mathrm{CSL}\binom{Person}{0} = 0$$

$$\mathrm{CSL}\binom{Person}{1} = 1$$

$$\mathrm{CSL}\binom{Bus}{1} = 0$$



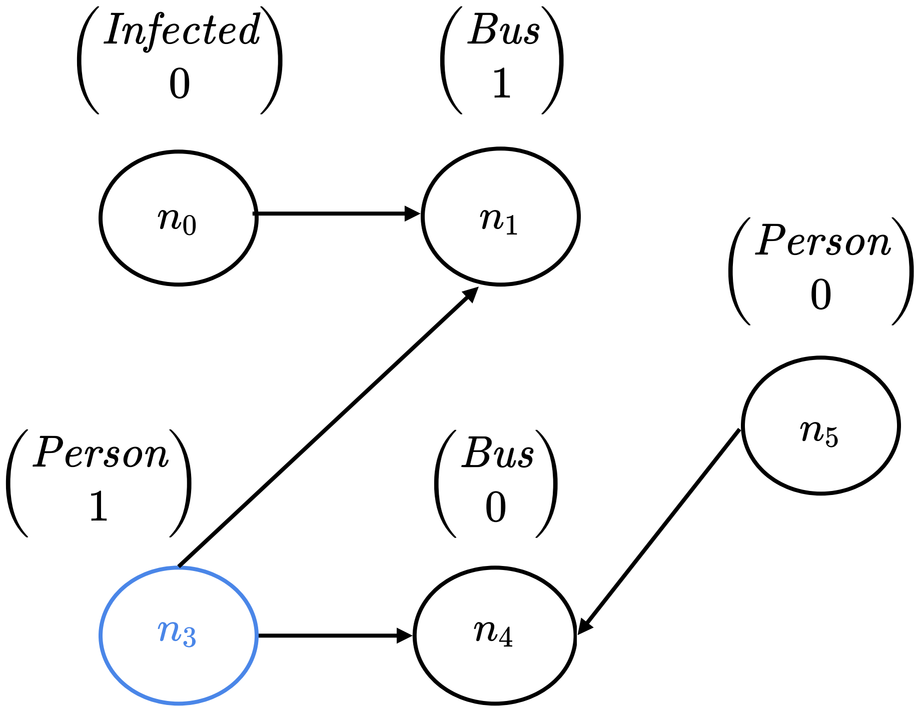
The architecture of GNNs

$$Person/rides/Bus/rides^{-}/Infected$$

$$\mathrm{CSL}\binom{Person}{0} = 0$$

$$\mathrm{CSL}\binom{Person}{1} = 1$$

$$\mathrm{CSL}\binom{Bus}{1} = 0$$



GNNs as a query language

- How do we explain the results of the query?
- Part of a more general issue: explainability or interpretability of black box model results

A call for an interpretability query language

- Several interpretability notions have been studied independently
- Interpretability admits no silver bullet; different contexts require different notions
- Interpretability may require combining different notions; it is better to think of it as an interactive process

A call for an interpretability query language

- This naturally suggests the possibility of interpretability query languages
- These language should de declarative, and should allow to express a wide variety of queries
- This gives control to the end-user to tailor interpretability queries to their particular needs

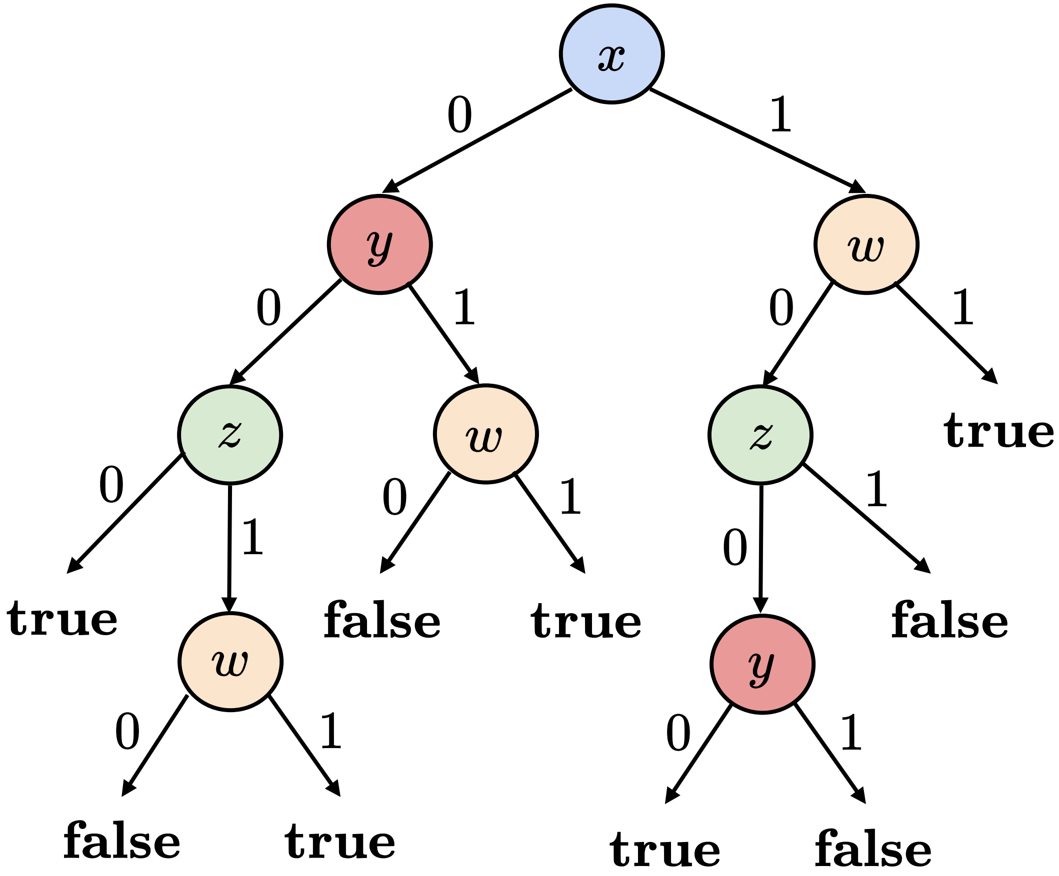
Our main goal is to develop such an interpretability query language

Basic ingredients: classification models are represented as labeled graphs, and first-order logic is used as query language

We start by focusing on a simple but widely used model

- **Decision trees** are widely used, in particular because they are considered *readily* interpretable models
- The main ingredients of our logical approach are already present in this case

A decision tree



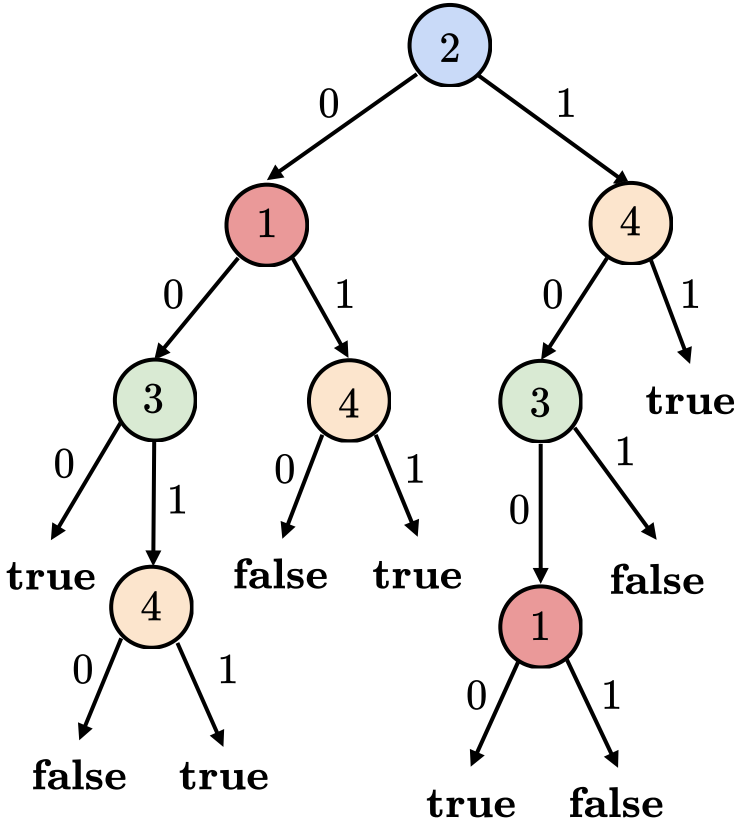
A classification model:

$$\mathcal{M} : \{0,1\}^n \to \{0,1\}$$

- The dimension of $\mathcal{M}$ is n, and each $i \in \{1, \ldots, n\}$ is called a feature
- $\mathbf{e} \in \{0,1\}^n$ is an instance
- $\mathcal{M}$ accepts $\mathbf{e}$ if $\mathcal{M}(\mathbf{e}) = 1$, otherwise $\mathcal{M}$ rejects $\mathbf{e}$

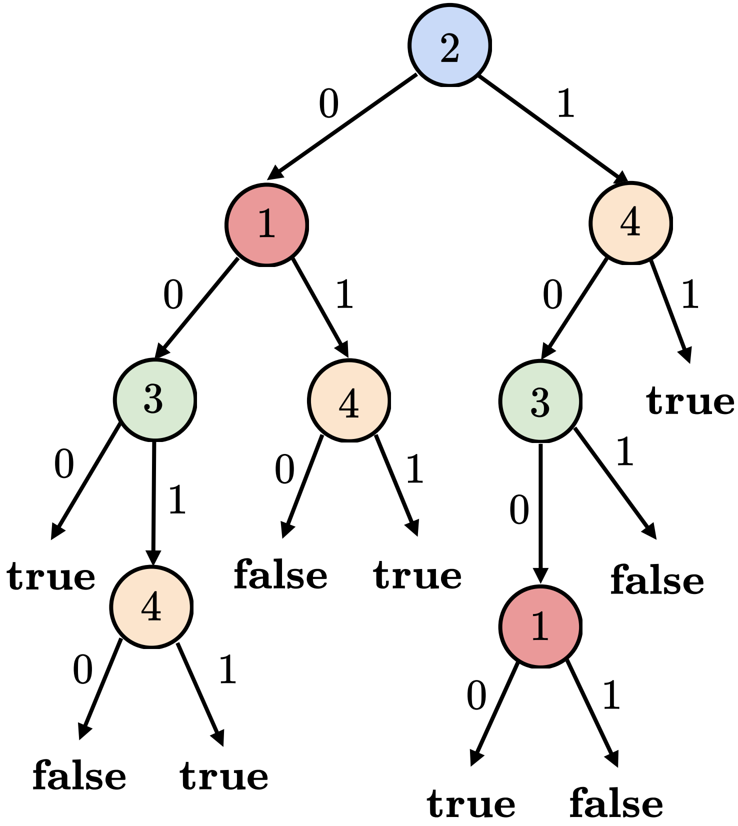
A decision tree $\mathcal{T}$ of dimension n

- Each internal node is labeled with a feature $i \in \{1, \ldots, n\}$, and has two outgoing edges labeled 0 and 1
- Each leaf is labeled **true** or **false**
- No two nodes on a path from the root to a leaf have the same label



A decision tree $\mathcal{T}$ of dimension n

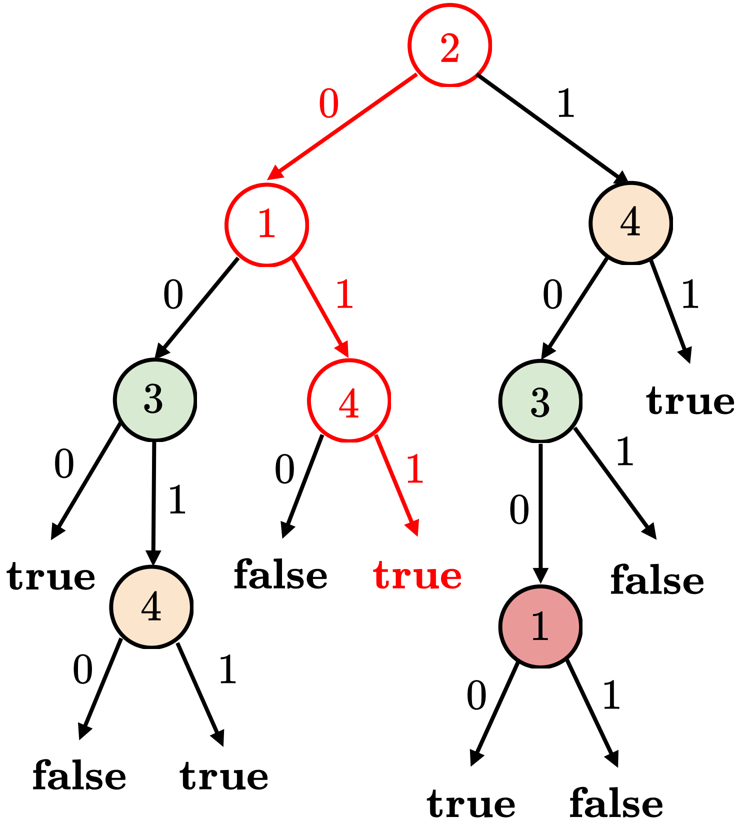
- Every instance $\mathbf{e}$ defines a unique path $n_1, e_1, n_2, \ldots, e_{k-1}, n_k$ from the root to a leaf
- $\mathcal{T}(\mathbf{e}) = 1$ if the label n_k is $\mathbf{true}$



A decision tree $\mathcal{T}$ of dimension n

- Every instance $\mathbf{e}$ defines a unique path $n_1, e_1, n_2, \ldots, e_{k-1}, n_k$ from the root to a leaf
- $\mathcal{T}(\mathbf{e}) = 1$ if the label n_k is $\mathbf{true}$

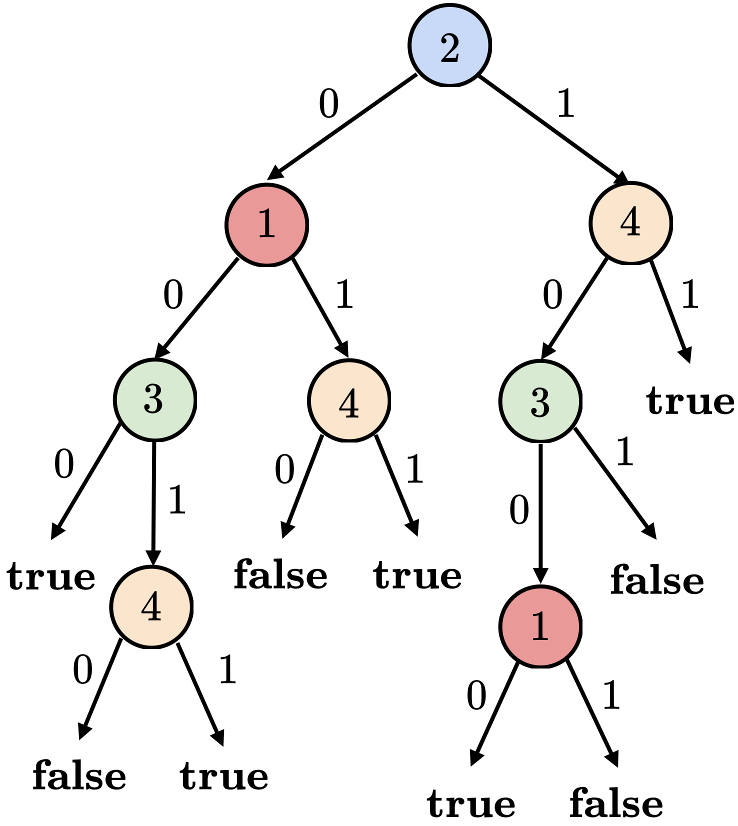
$\mathcal{T}(\mathbf{e_1}) = 1$ for instance $\mathbf{e_1} = (1, 0, 1, 1)$



The evaluation of a model as a query

Is $\mathcal{T}(\mathbf{e_1}) = 1$ for instance $\mathbf{e_1} = (1, 0, 1, 0)$?

$\left(1/1 + 2/0 + 3/1 + 4/0\right)^{*}/\mathbf{true}$



But our problem is to explain the output of a model

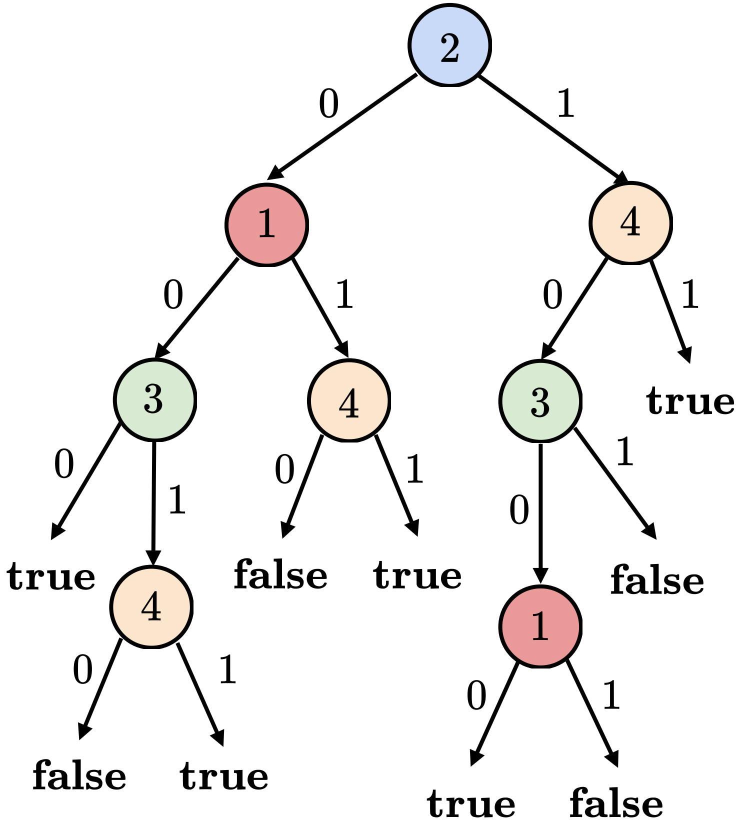
- What are interesting notions of explanation?
- What notions have been studied? What notions are used in practice?
- Can these notions be expressed as queries over decision trees?

But our problem is to explain the output of a model

Is there a completion of $2 \mapsto 0$ that is classified positively?

$$\big(1/(0+1) + {\color{red}2/0} + 3/(0+1) + 4/(0+1)\big)^*/\mathbf{true}$$

Are all the completions of $2 \mapsto 0$ classified positively?

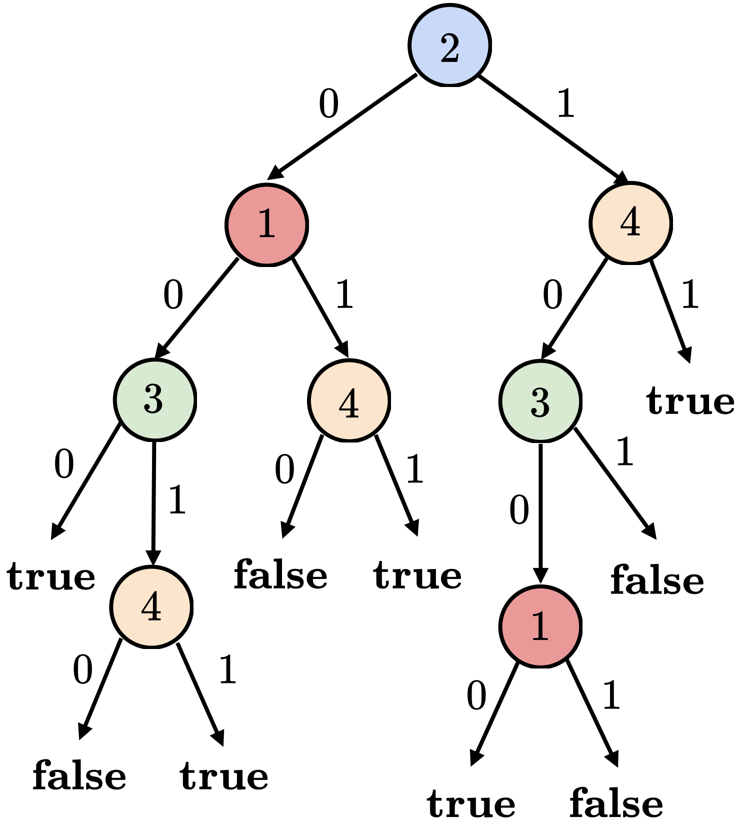


Notions of explanation: sufficient reason

$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1, 1, 1, 1)$

The value of feature 3 is not needed to obtain this result

- $\{1, 2, 4\}$ is a *sufficient reason*

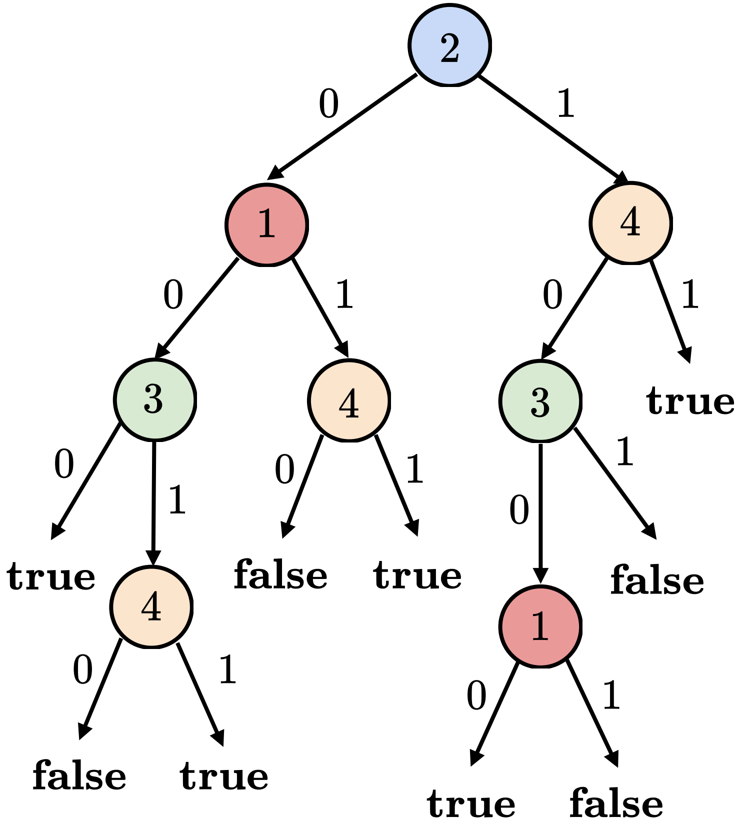


Notions of explanation: minimal sufficient reason

$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1, 1, 1, 1)$

The value of features 1 and 3 are not needed to obtain this result

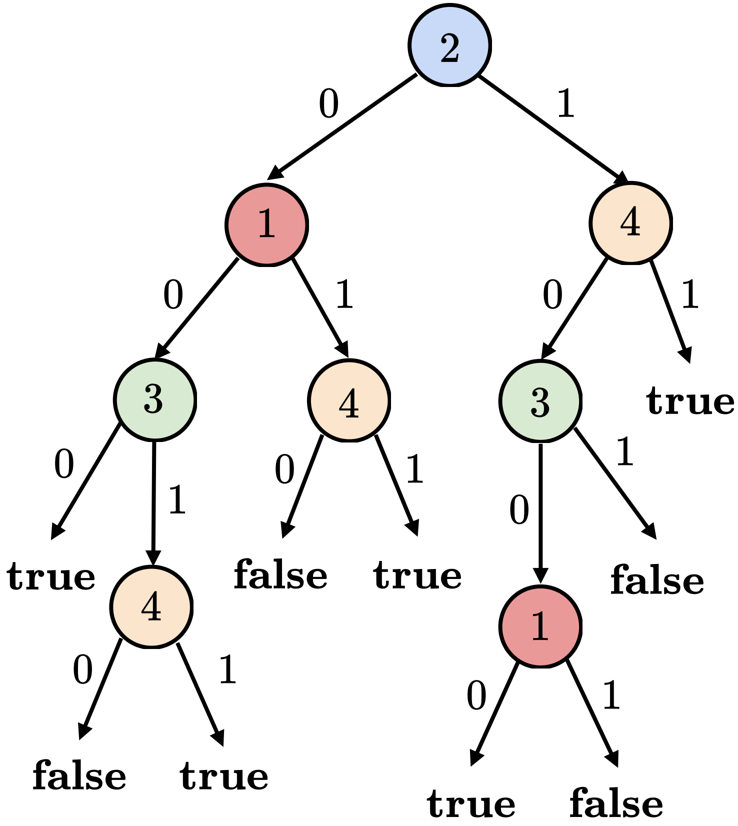
- $\{2, 4\}$ is a *minimal sufficient reason*



Notions of explanation: relevant feature set

If the values of features $\{1, 3, 4\}$ are fixed, then the output of the model is fixed

The output of the model depends only on these features

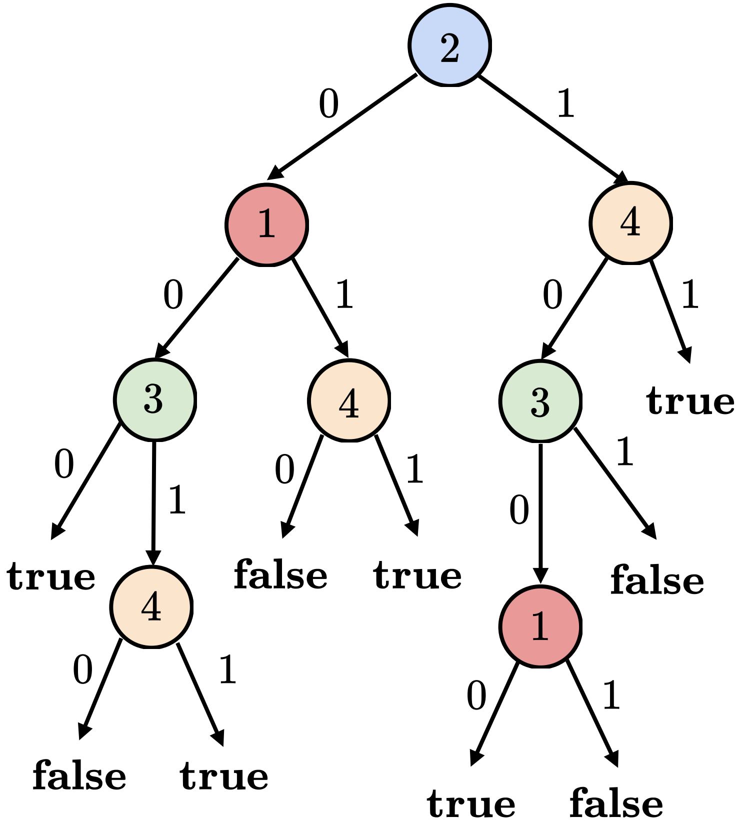


Notions of explanation: relevant feature set

If the values of features $\{1,3,4\}$ are fixed, then the output of the model is fixed

If $\mathbf{e}[1] = \mathbf{e}[3] = \mathbf{e}[4] = 0$:

$$\mathcal{T}(\mathbf{e}) = 1$$

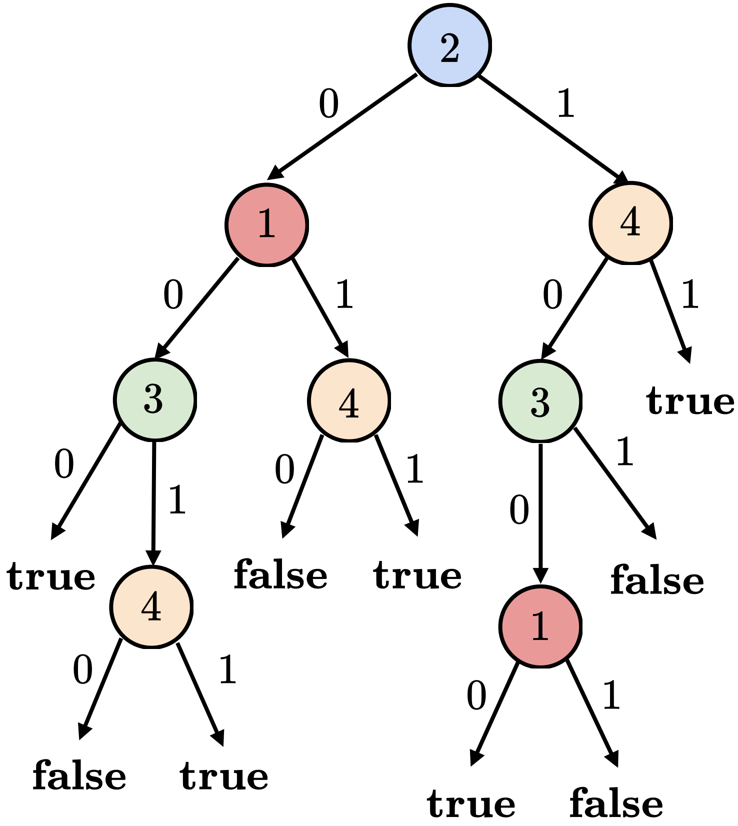


Notions of explanation: relevant feature set

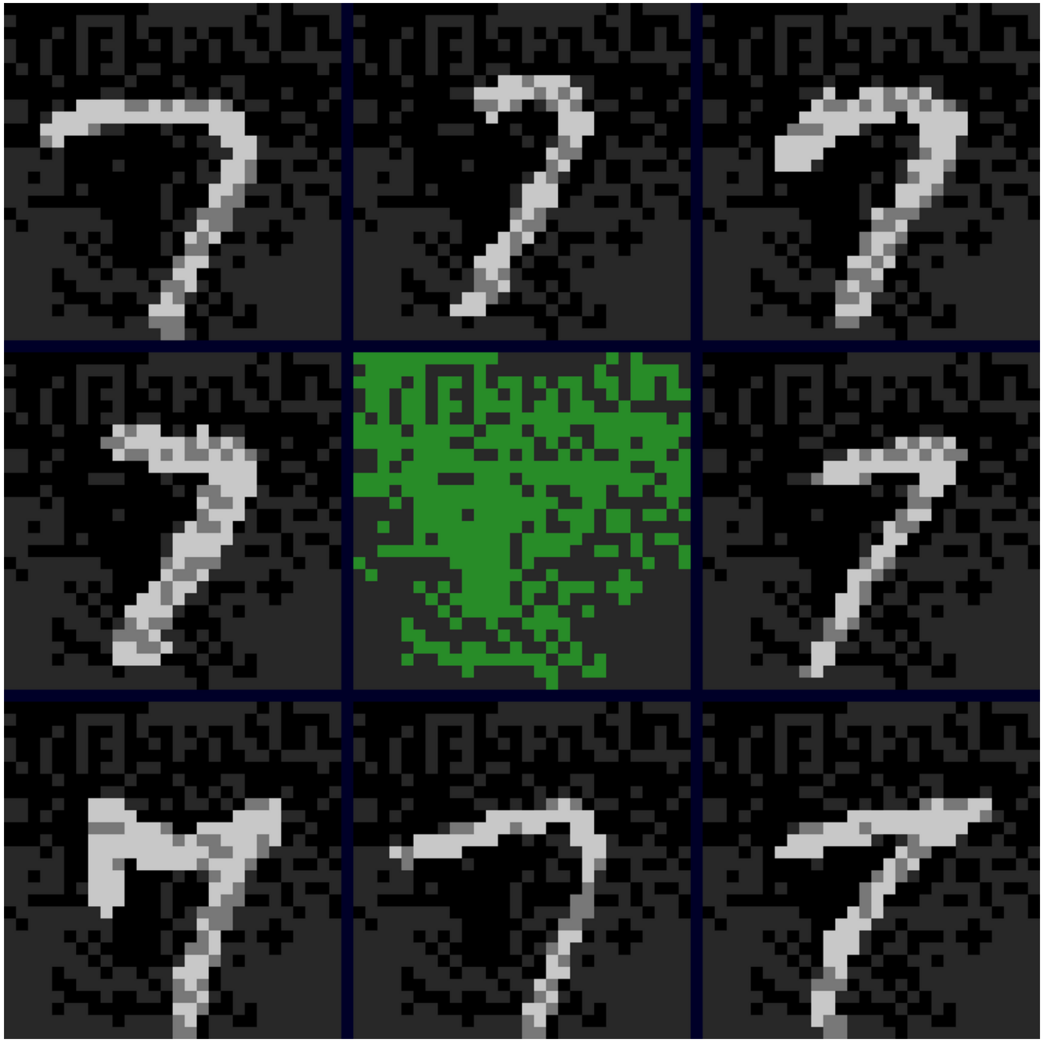
If the values of features $\{1, 3, 4\}$ are fixed, then the output of the model is fixed

If $\mathbf{e}[1] = \mathbf{e}[3] = 1$ and $\mathbf{e}[4] = 0$:

$$\mathcal{T}(\mathbf{e}) = 0$$



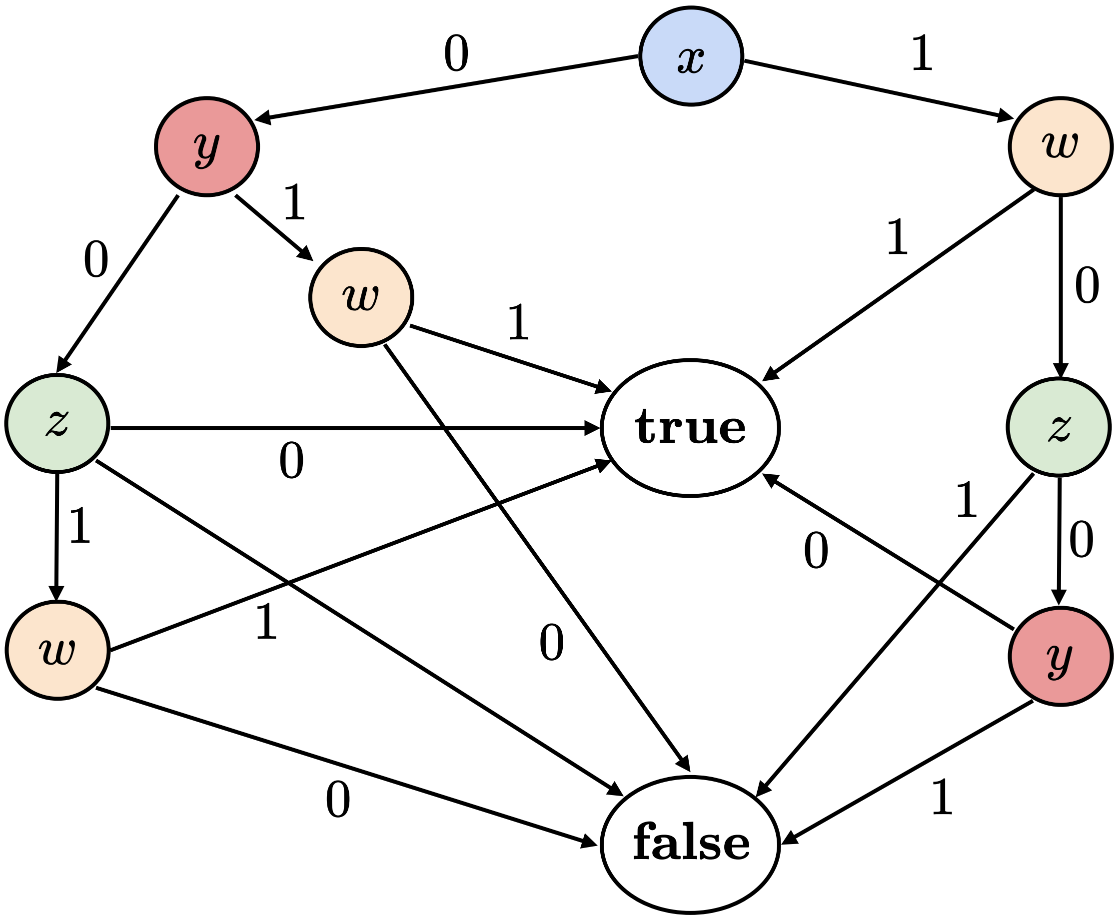
MNIST: relevant feature set



Can these queries be expressed in a graph query language?

- How do we express the previous interpretability queries?
- Is there a common framework for them?
- Is there a *natural* framework based on labeled graphs?

Can these queries be expressed in a graph query language?



Binary decision diagrams (BDDs)

- OBDDs
- FBDDs

A first attempt: FOIL

First-order logic defined on a suitable vocabulary to describe classification models

Key notion: partial instance $\mathbf{e} \in \{0, 1, \perp\}^n$ of dimensión n

$\mathbf{e}_1$ is subsumed by $\mathbf{e}_2$ if $\mathbf{e}_1, \mathbf{e}_2$ are partial instances such that for every $i \in \{1, \ldots, n\}$, if $\mathbf{e}_1[i] \neq \perp$, then $\mathbf{e}_1[i] = \mathbf{e}_2[i]$

$$(1, \perp, 0, \perp) \ \subseteq \ (1, 0, 0, \perp) \ \subseteq \ (1, 0, 0, 1)$$

A first attempt: FOIL

First-order logic defined on a suitable vocabulary to describe classification models: $\{\mathrm{Pos}, \subseteq\}$

A model $\mathcal{M}$ of dimensión n is represented as a structure $\mathfrak{A}_{\mathcal{M}}$:

- The domain of $\mathfrak{A}_{\mathcal{M}}$ is $\{0, 1, \perp\}^n$
- $\mathrm{Pos}(\mathbf{e})$ holds if $\mathbf{e}$ is an instance such that $\mathcal{M}(\mathbf{e}) = 1$
- $\mathbf{e}_1 \subseteq \mathbf{e}_2$ holds if $\mathbf{e}_1, \mathbf{e}_2$ are partial instances such that $\mathbf{e}_1$ is subsumed by $\mathbf{e}_2$

The semantics of FOIL

Given a **FOIL** formula $\Phi(x_1, x_2, \ldots, x_k)$, a classification model $\mathcal{M}$ of dimensión n, and instances $\mathbf{e}_1$, $\mathbf{e}_2$, ..., $\mathbf{e}_k$

$$\mathcal{M} \models \Phi(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k)$$

$$\Longleftrightarrow$$

$$\mathfrak{A}_{\mathcal{M}} \models \Phi(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k)$$

(in the usual sense)

Some examples

$$\mathrm{Full}(x) \;=\; \forall y\,(x \subseteq y \to x = y)$$

$$\mathrm{AllPos}(x) \;=\; \forall y\,\big((x \subseteq y \wedge \mathrm{Full}(y)) \to \mathrm{Pos}(y)\big)$$

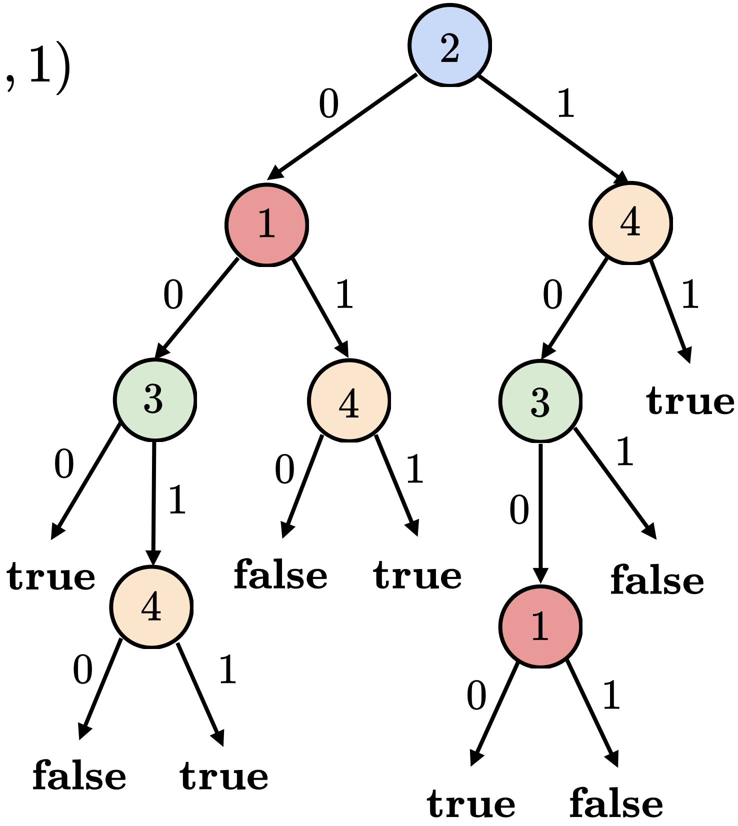
$$\mathrm{AllNeg}(x) \;=\; \forall y\,\big((x \subseteq y \wedge \mathrm{Full}(y)) \to \neg\mathrm{Pos}(y)\big)$$

Notions of explanation: sufficient reason

$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1,1,1,1)$, and $\mathbf{e}_1 = (1,1,\perp,1)$ is a sufficient reason for this

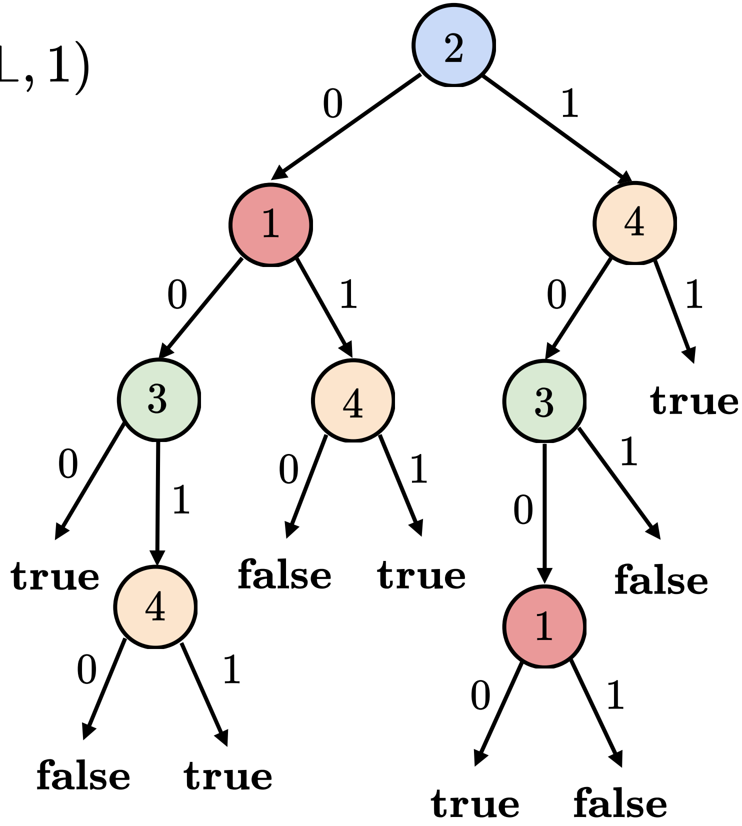
$\mathcal{T} \models \mathrm{SR}(\mathbf{e}, \mathbf{e}_1)$

$$\begin{aligned} \mathrm{SR}(x,y) \;=\; & \mathrm{Full}(x) \wedge y \subseteq x \wedge \\ & (\mathrm{Pos}(x) \to \mathrm{AllPos}(y)) \wedge \\ & (\neg\mathrm{Pos}(x) \to \mathrm{AllNeg}(y)) \end{aligned}$$



Notions of explanation: minimal sufficient reason

$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1,1,1,1)$, and $\mathbf{e}_2 = (\perp, 1, \perp, 1)$ is a minimal sufficient reason for this

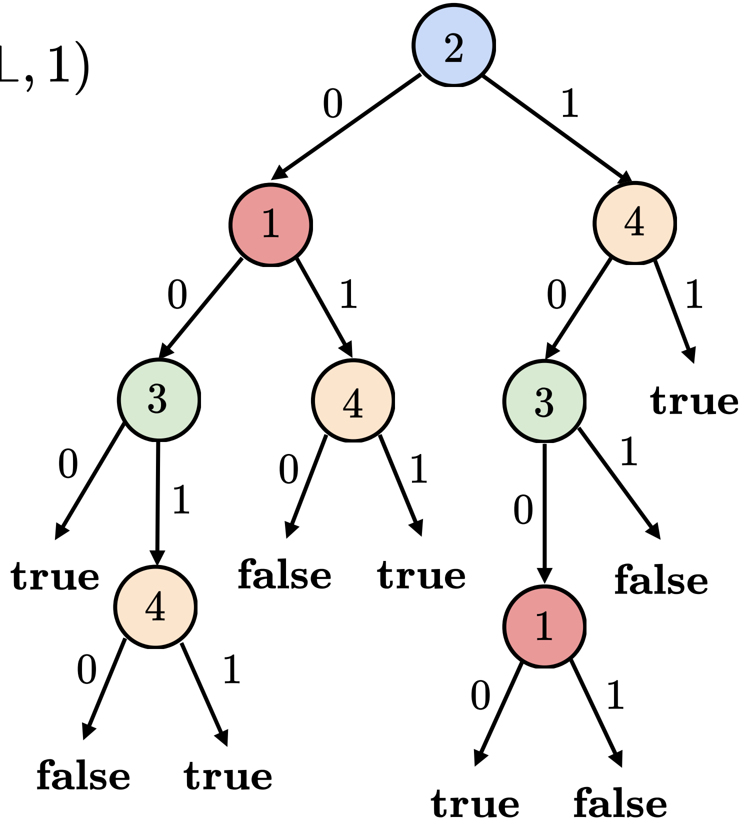


Notions of explanation: minimal sufficient reason

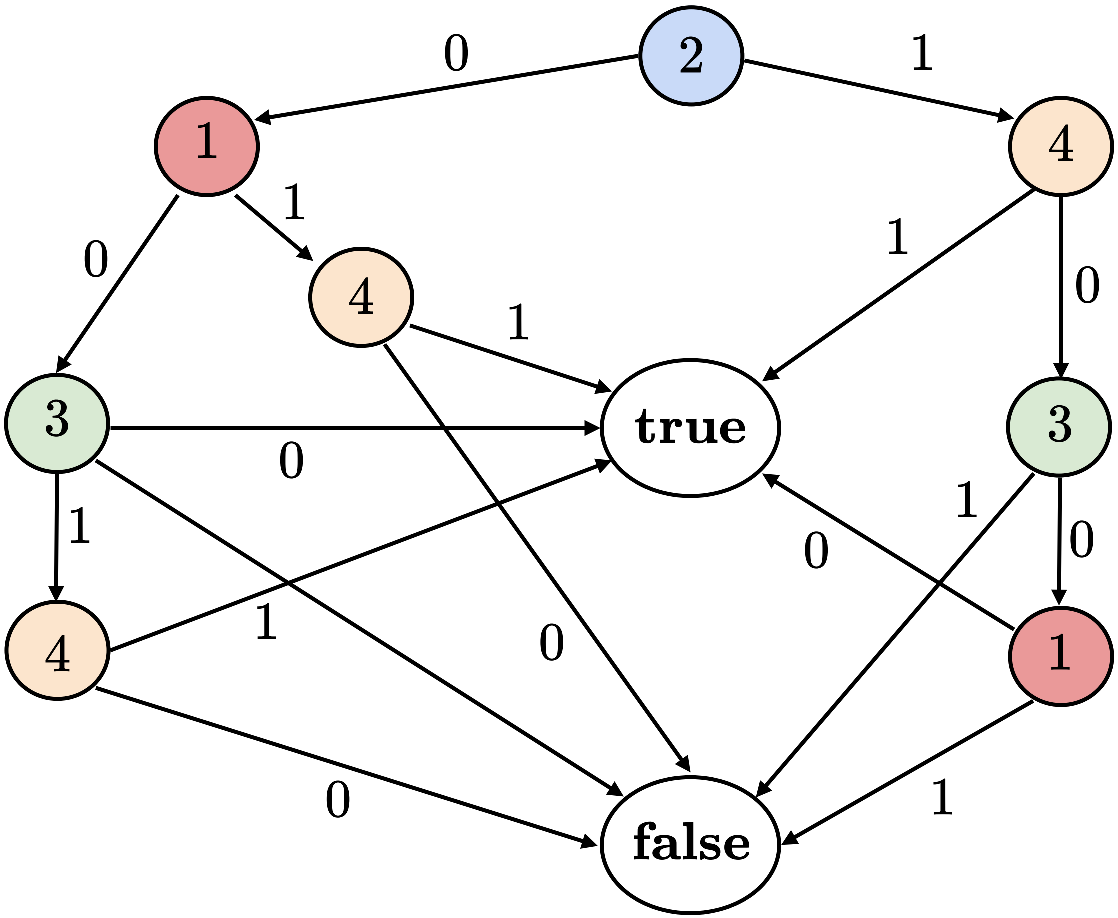
$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1,1,1,1)$, and $\mathbf{e}_2 = (\bot, 1, \bot, 1)$ is a minimal sufficient reason for this

$$\mathcal{T} \models \mathrm{MinimalSR}(\mathbf{e}, \mathbf{e}_2)$$

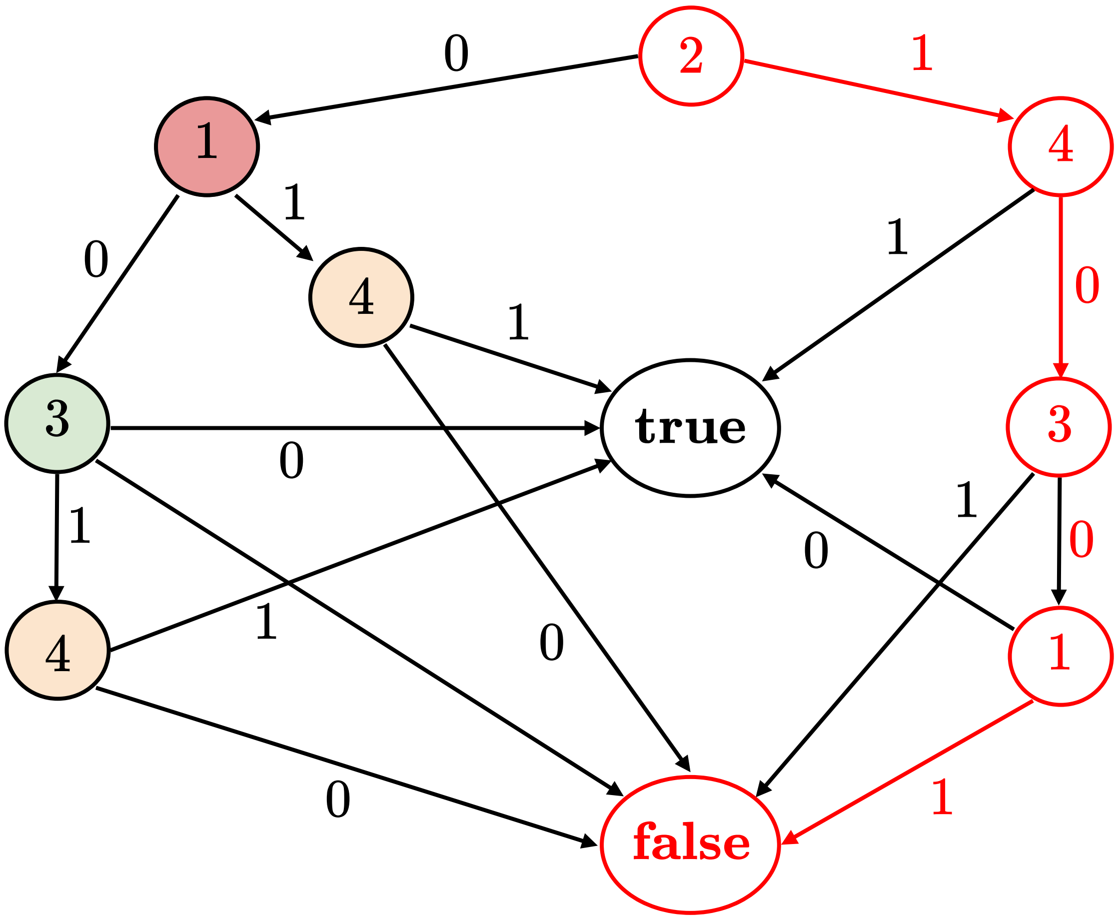
$$\mathrm{MinimalSR}(x, y) \ = \ \mathrm{SR}(x,y) \wedge \forall z \left((\mathrm{SR}(x,z) \wedge z \subseteq y) \rightarrow z = y \right)$$



FOIL is a graph query language

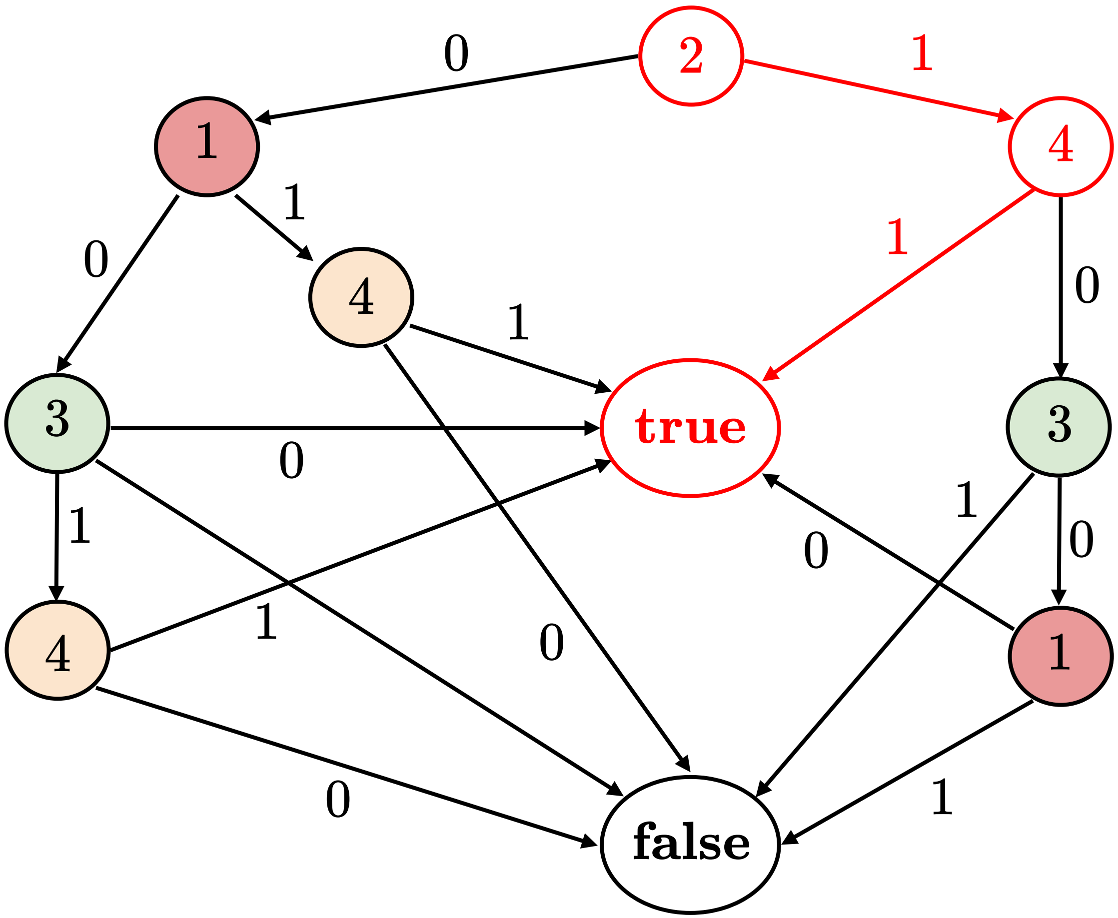


FOIL is a graph query language

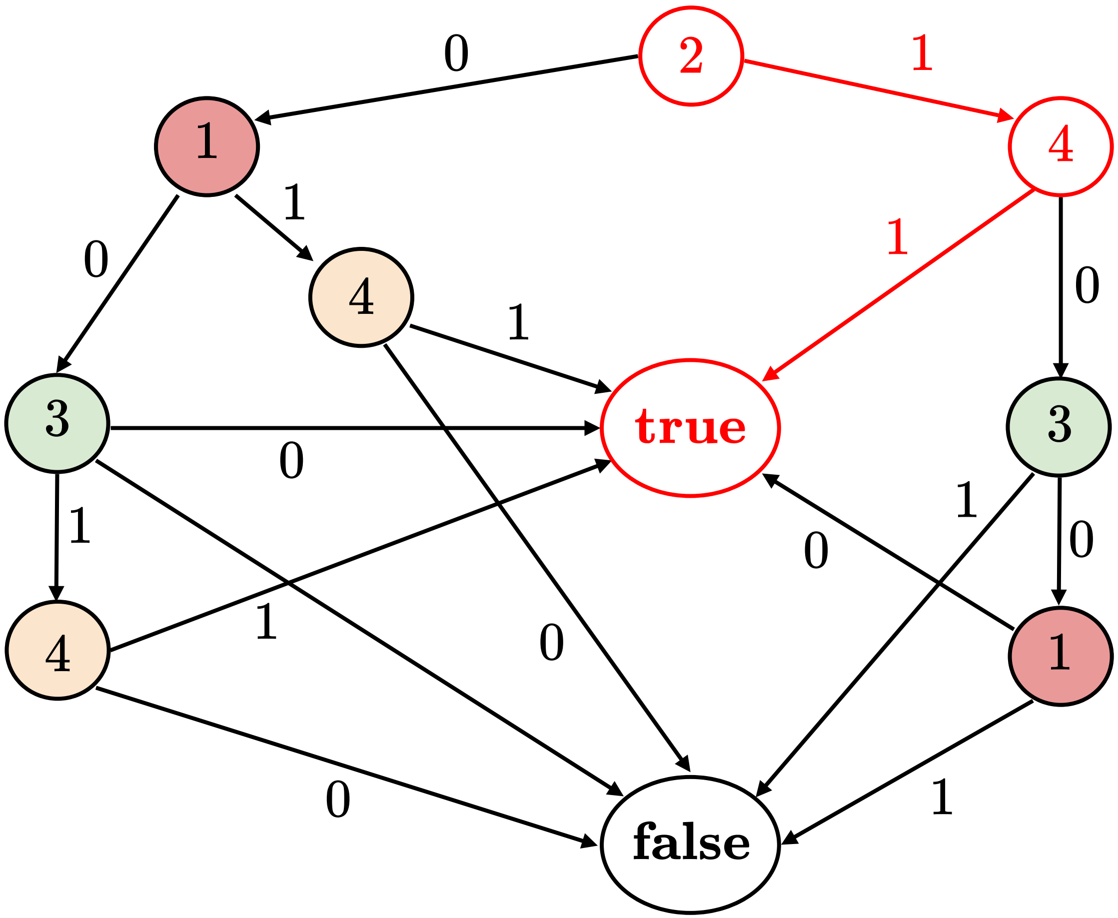


This path represents the instance $(1, 1, 0, 0)$

FOIL is a graph query language



FOIL is a graph query language



This path represents the partial instance $(\perp, 1, \perp, 1)$

Expressiveness and complexity of FOIL

- What notions of explanation can be expressed in **FOIL**?
- What notions of explanation cannot be expressed in **FOIL**?
- What is the complexity of the evaluation problem for **FOIL**?

The evaluation problem for FOIL

We consider the data complexity of the problem, so fix a **FOIL** formula $\Phi(x_1, \ldots, x_k)$

$\mathbf{Eval(\Phi)}$**:**

- **Input:** decision tree $\mathcal{T}$ of dimension n and partial instances $\mathbf{e}_1, \ldots, \mathbf{e}_k$ of dimension n
- **Output:** yes if $\mathcal{T} \models \Phi(\mathbf{e}_1, \ldots, \mathbf{e}_k)$, and no otherwise

The evaluation problem for FOIL

$\mathcal{T} \models \Phi(\mathbf{e}_1, \ldots, \mathbf{e}_k)$ if and ony if $\mathfrak{A}_{\mathcal{T}} \models \Phi(\mathbf{e}_1, \ldots, \mathbf{e}_k)$

But $\mathfrak{A}_{\mathcal{T}}$ could be of exponential size in the size of $\mathcal{T}$

- $\mathfrak{A}_{\mathcal{T}}$ should not be materialized to check whether $\mathcal{T} \models \Phi(\mathbf{e}_1, \ldots, \mathbf{e}_k)$
- $\mathfrak{A}_{\mathcal{T}}$ is used only to define the semantics of **FOIL**

Bad news ...

Theorem:

1. For every **FOIL** formula Φ, there exists $k \geq 0$ such that $\mathrm{Eval}(\Phi)$ is in Σ_k^{P}
2. For every $k \geq 0$, there exists a **FOIL** formula Φ such that $\mathrm{Eval}(\Phi)$ is Σ_k^{P}-hard

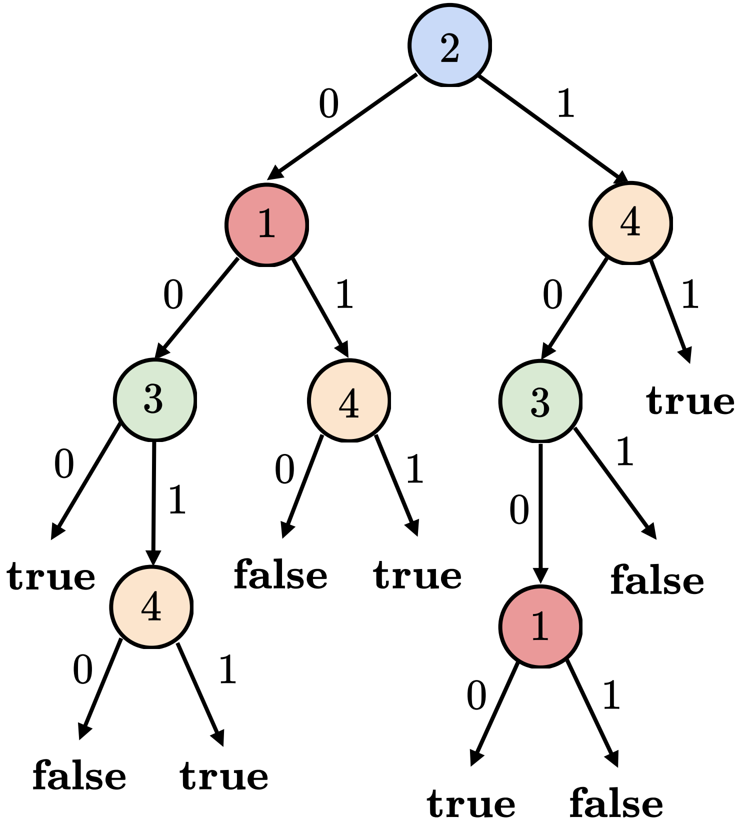
More bad news ...

$\mathcal{T}(\mathbf{e}) = 1$ for $\mathbf{e} = (1, 1, 1, 1)$

$\{2, 4\}$ is a minimum sufficient reason for $\mathbf{e}$ over $\mathcal{T}$

- There is no sufficient reason for $\mathbf{e}$ over $\mathcal{T}$ with a smaller number of features

$\mathbf{e}_2 = (\perp, 1, \perp, 1)$ is a minimum sufficient reason for $\mathbf{e}$ over $\mathcal{T}$



More bad news ...

Theorem:

There is no **FOIL** formula $\mathrm{MinimumSR}(x, y)$ such that, for every decision tree $\mathcal{T}$, instance $\mathbf{e}_1$ and partial instance $\mathbf{e}_2$:

$$\mathcal{T} \models \mathrm{MinimumSR}(\mathbf{e}_1, \mathbf{e}_2)$$

$$\iff$$

$\mathbf{e}_2$ is a minimum sufficient reason for $\mathbf{e}_1$ over $\mathcal{T}$

How do we overcome these limitations?

- We use first-order logic, over a larger vocabulary but with some syntactic restrictions
- We continue using some common notions for graphs
- Our goal is to find languages with polynomial or even NP data complexity, since the latter allows the use of SAT solvers

The StratiFOILed Logic

The logic **StratiFOILed** is defined by considering three layers

1. Atomic formulas
2. Guarded formulas
3. The formulas from **StratiFOILed** itself

The first layer

$\subseteq$ can be considered as a *syntactic* predicate, it does not refer to the models

We need another predicate like that. Given partial instances $\mathbf{e}_1, \mathbf{e}_2$ of dimension n:

$\mathbf{LEL}(\mathbf{e}_1, \mathbf{e}_2)$ holds

if and only if

$$|\{i \in \{1, \ldots n\} \mid \mathbf{e}_1[i] = \bot\}| \geq |\{i \in \{1, \ldots n\} \mid \mathbf{e}_1[i] = \bot\}|$$

Why do we need another syntactic predicate?

$$\begin{aligned}\mathrm{MinimumSR}(x,y) \;=\; & \mathrm{SR}(x,y) \wedge \\ & \forall z\, \big((\mathrm{SR}(x,z) \wedge \mathrm{LEL}(z,y)) \rightarrow \mathrm{LEL}(y,z)\big)\end{aligned}$$

How many more predicates do we need to include?

Atomic formulas

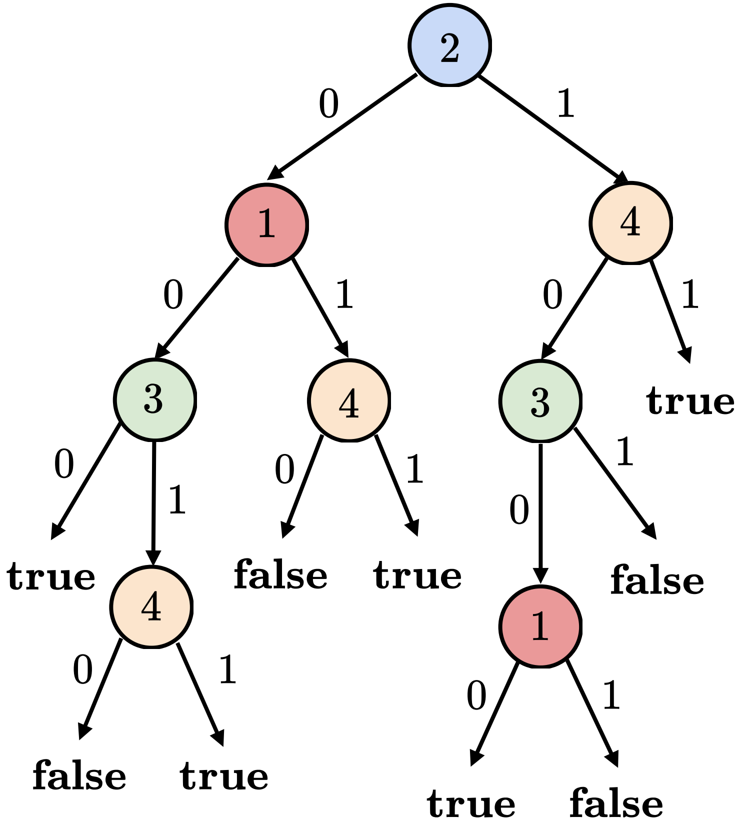
All the syntactic predicates needed in our formalism can be expressed as first-order queries over $\{\subseteq, \mathrm{LEL}\}$

Theorem: if Φ is a first-order formula defined over $\{\subseteq, \mathrm{LEL}\}$, then $\mathrm{Eval}(\Phi)$ can be solved in polynomial time

Atomic formulas of StratiFOILed: the set of first-order formulas defined over $\{\subseteq, \mathrm{LEL}\}$

The second layer

Node($\mathbf{e}$):



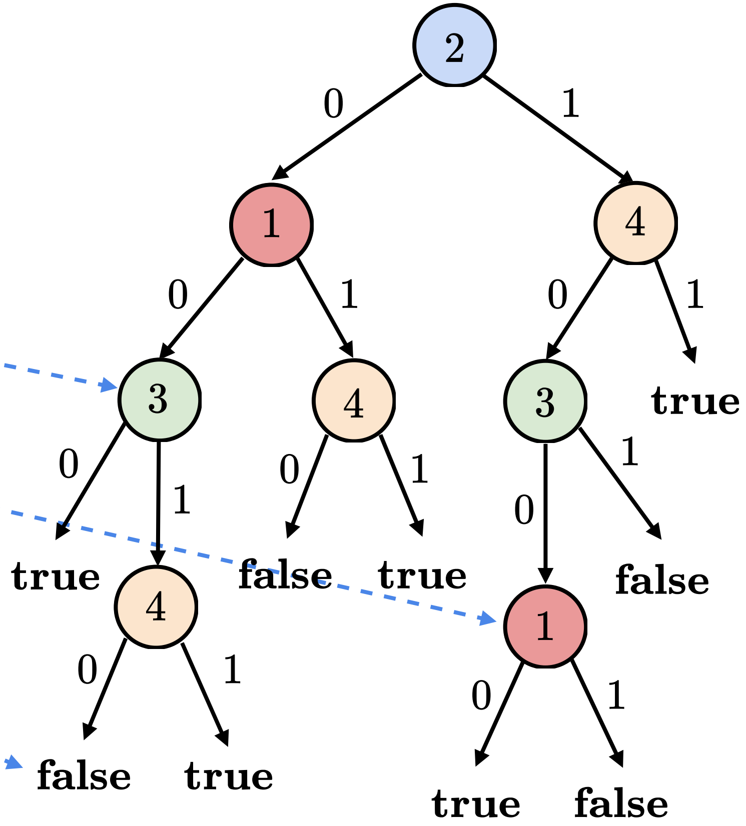
The second layer

Node($\mathbf{e}$):

$(0, 0, \perp, \perp)$

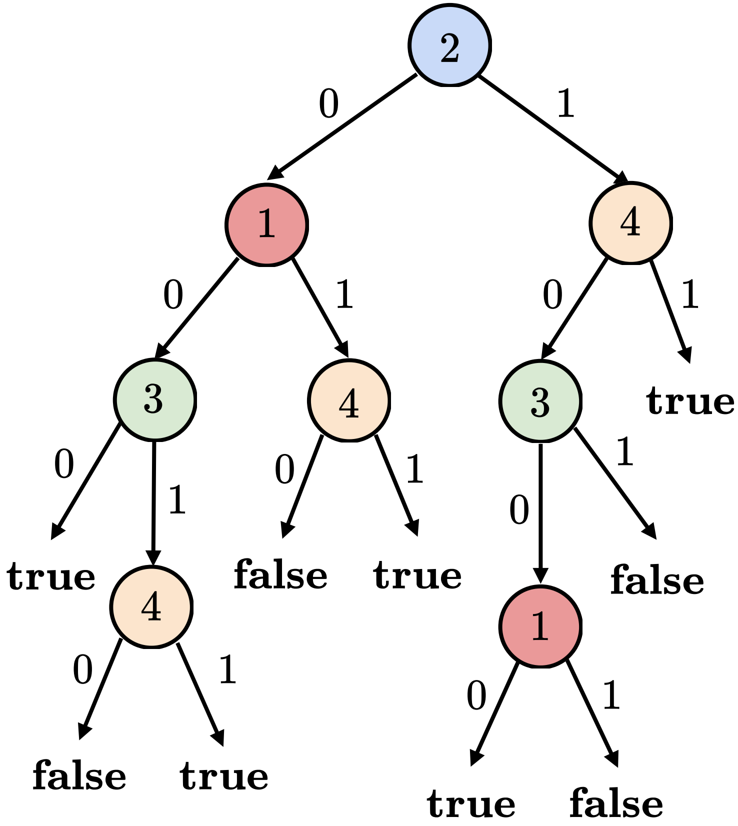
$(\perp, 1, 0, 0)$

$(0, 0, 1, 0)$



The second layer

$\mathrm{PosLeaf}(\mathbf{e})$:

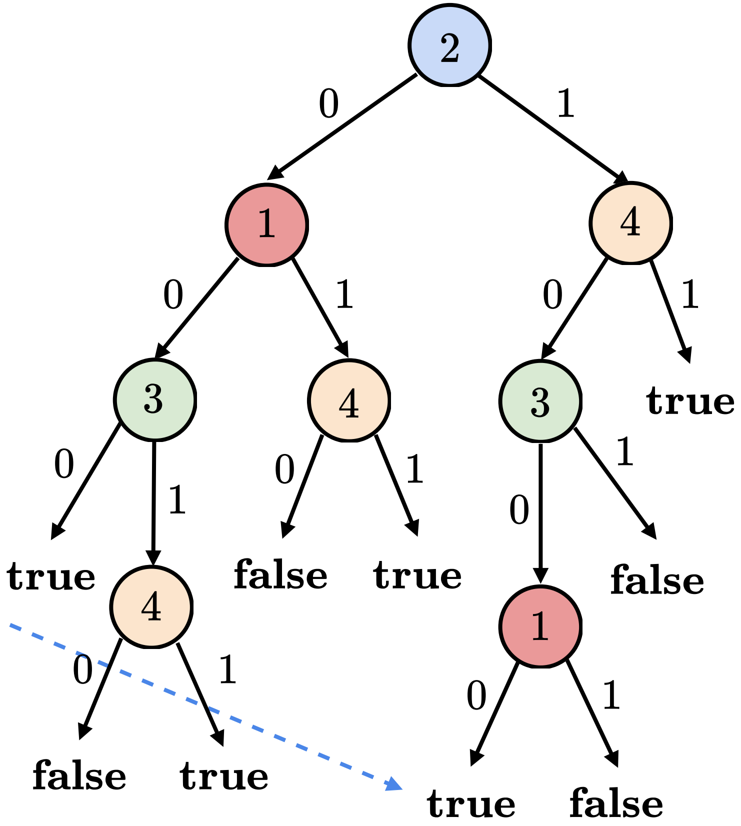


The second layer

$\mathrm{PosLeaf}(\mathbf{e})$:

$(0, 0, 0, \perp)$

$(0, 1, 0, 0)$



Guarded formulas

1. Each atomic formula is a guarded formula
2. Boolean combinations of guarded formulas are guarded formulas
3. If Φ is a guarded formula, then so are

$$\exists x\ (\mathrm{Node}(x) \wedge \Phi) \qquad \forall x\ (\mathrm{Node}(x) \rightarrow \Phi)$$

$$\exists x\ (\mathrm{PosLeaf}(x) \wedge \Phi) \qquad \forall x\ (\mathrm{PosLeaf}(x) \rightarrow \Phi)$$

An example of a guarded formula

$$
\begin{aligned}
\mathrm{FRS}(x) \;=\; & \forall y \left[\mathrm{Node}(y) \to (\mathrm{AllPos}(y) \;\to \right. \\
& \quad \forall z \,(\mathrm{Node}(z) \to (\mathrm{AllNeg}(z) \to \\
& \qquad \left. \neg\exists w\,(\mathrm{Suf}(x,w) \wedge \mathrm{Cons}(w,y) \wedge \mathrm{Cons}(w,z)))))\right]
\end{aligned}
$$

guarded formula

An example of a guarded formula

$$\begin{aligned}\mathrm{FRS}(x) \;=\; \forall y \Big[\mathrm{Node}(y) \to (\mathrm{AllPos}(y) \;\to\; & \\ \forall z\, (\mathrm{Node}(z) \to (\mathrm{AllNeg}(z) \to & \\ \neg\exists w\, (\mathrm{Suf}(x,w) \wedge \mathrm{Cons}(w,y) \wedge \mathrm{Cons}(w,z))))) & \Big]\end{aligned}$$

guarded formula

An example of a guarded formula

$$\begin{aligned}\mathrm{FRS}(x) \;=\; \forall y \Big[\mathrm{Node}(y) \to (\mathrm{AllPos}(y) \;\to\; & \\ \forall z\,(\mathrm{Node}(z) \to (\mathrm{AllNeg}(z) \to & \\ \neg\exists w\,(\mathrm{Suf}(x,w) \wedge \mathrm{Cons}(w,y) \wedge \mathrm{Cons}(w,z))))\Big] &\end{aligned}$$

atomic formulas

An example of a guarded formula

$$
\begin{aligned}
\mathrm{FRS}(x) \;=\; \forall y \Big[\mathrm{Node}(y) \to (\mathrm{AllPos}(y) \;\to\; & \\
\forall z \,(\mathrm{Node}(z) \to (\mathrm{AllNeg}(z) \to & \\
\neg\exists w \,(\mathrm{Suf}(x, w) \wedge \mathrm{Cons}(w, y) \wedge \mathrm{Cons}(w, z))))) & \Big]
\end{aligned}
$$

atomic formula

The third layer: StratiFOILed

1. Each guarded formula is a **StratiFOILed** formula
2. If Φ is a guarded formula, then $\exists x_1 \cdots \exists x_k \, \Phi$ and $\forall x_1 \cdots \forall x_k \, \Phi$ are **StratiFOILed** formulas
3. Boolean combinations of **StratiFOILed** formulas are **StratiFOILed** formulas

Examples of StratiFOILed formulas

$\mathrm{SR}(x,y)$, $\mathrm{MinimalSR}(x,y)$, $\mathrm{MinimumSR}(x,y)$ can be expressed as **StratiFOILed** formulas

$$\begin{aligned}\mathrm{FRS}(x) \;=\; \forall y \,\Big[\mathrm{Node}(y) \to (\mathrm{AllPos}(y) \;\to \\ \forall z \,(\mathrm{Node}(z) \to (\mathrm{AllNeg}(z) \to \\ \neg\exists w \,(\mathrm{Suf}(x,w) \wedge \mathrm{Cons}(w,y) \wedge \mathrm{Cons}(w,z)))))\Big]\end{aligned}$$

$\mathrm{MinimalFRS}(x)$, $\mathrm{MinimumFRS}(x)$ can be expressed as a **StratiFOILed** formulas

The evaluation problem for StratiFOILed

BH: Boolean Hierarchy over NP

Theorem:

1. For each **StratiFOILed** formula Φ, there exists $k \geq 1$ such that $\mathrm{Eval}(\Phi)$ is in BH_k
2. For every $k \geq 1$, there exists a **StratiFOILed** formula Φ such that $\mathrm{Eval}(\Phi)$ is in BH_k-hard

The evaluation problem for StratiFOILed

$\mathrm{Eval}(\Phi)$ can be solved with a fixed number of calls to a SAT solver, for each **StratiFOILed** formula Φ

Implementation based on SAT solvers

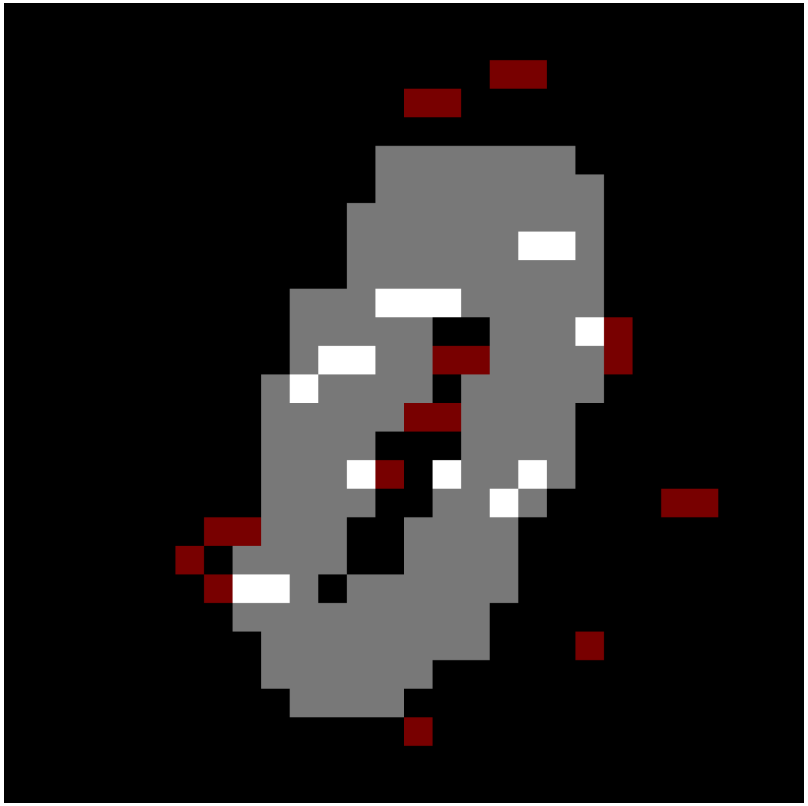
Any SAT solver can be used

Given the complexity of the evaluation problem for **StratiFOILed**, we use:

- **YalSAT:** to find a truth assignment that satisfies a propositional formula
- **Kissat:** to prove that a propositional formula is not satisfiable

MNIST: sufficient reason

$\mathbf{e}$

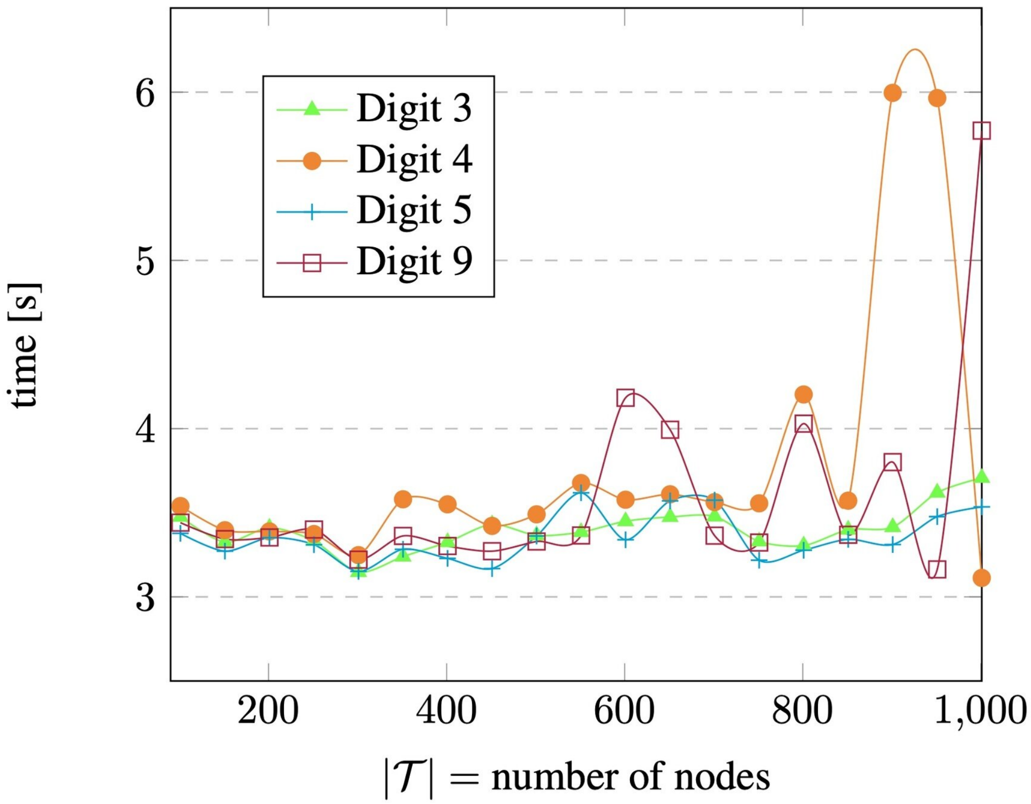


$$\alpha(x, z) = \exists y\,(\mathrm{SR}(x, y) \wedge \mathrm{LEL}(y, z))$$

Evaluate whether $\alpha(\mathbf{e}, \mathbf{u}_{730})$ holds

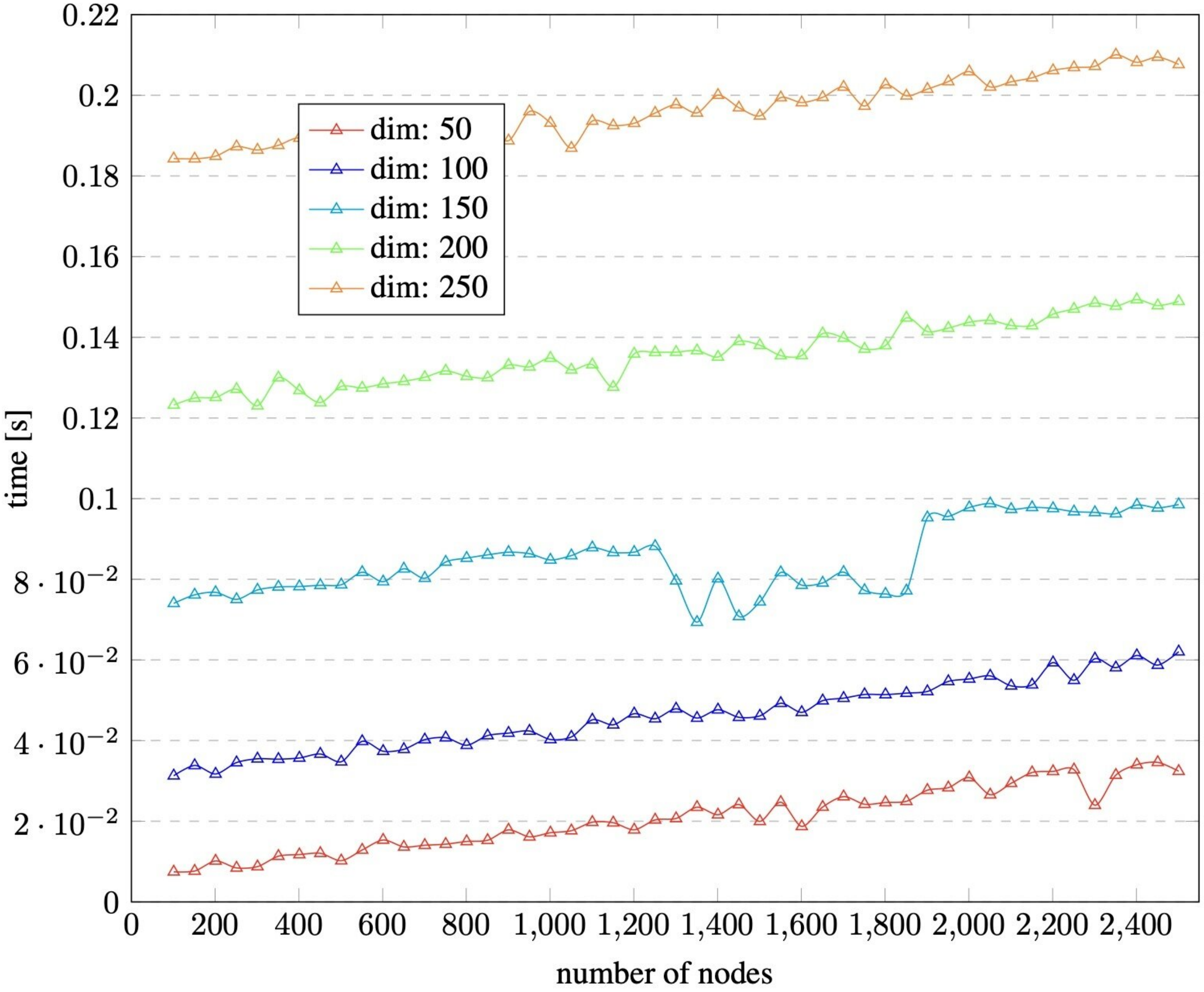
$\mathbf{u}_{730} \in \{0, 1, \perp\}^{784}$ satisfies that $|\{i \in \{1, \ldots, 784\} \mid \mathbf{u}_{730}[i] = \perp\}| = 730$

MNIST: sufficient reason



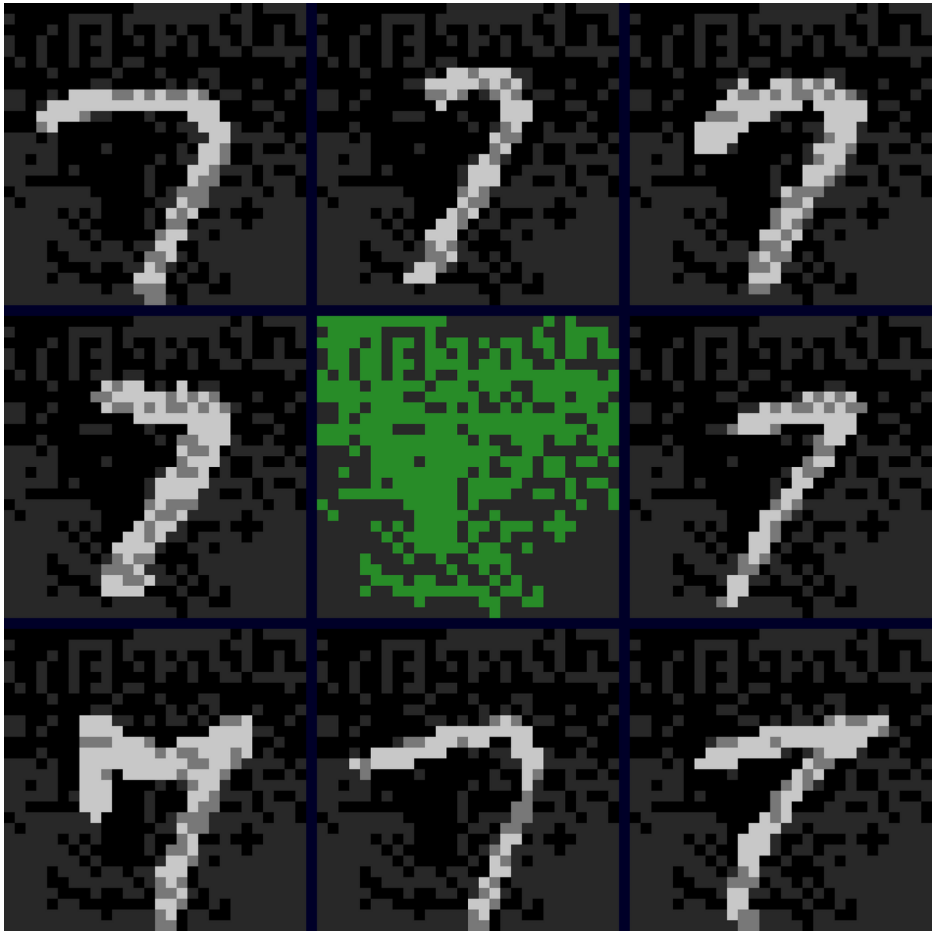
Evaluation time of $\alpha(x, \mathbf{u}_{720})$

Synthetic data: sufficient reason



Evaluation time of $\alpha(x, \mathbf{u}_\ell)$ with $\ell = 0.05 \times \mathrm{dim}$

MNIST: relevant feature set



$$\beta(y) = \exists x\,(\mathrm{RFS}(x) \wedge \mathrm{LEL}(x, y))$$

Evaluate whether $\alpha(\mathbf{u}_{392})$ holds

Concluding remarks

StratiFOILed is a model-specific interpretability query language

- How can the definition of **StratiFOILed** be extended to OBDDs and FBDDs?
- What are the right definitions of $\mathrm{Node}(x)$ and $\mathrm{PosLeaf}(x)$ in these cases?

Concluding remarks

FOIL is a model-agnostic interpretability query language

- The evaluation problem for some fragments of **FOIL** can be solved in polynomial time for decision trees and OBDDs
- What is an appropriate fragment of **FOIL** to be evaluated using SAT solvers?
- What is an appropriate interpretability query language for FBDDs that is based on **FOIL**?

Concluding remarks

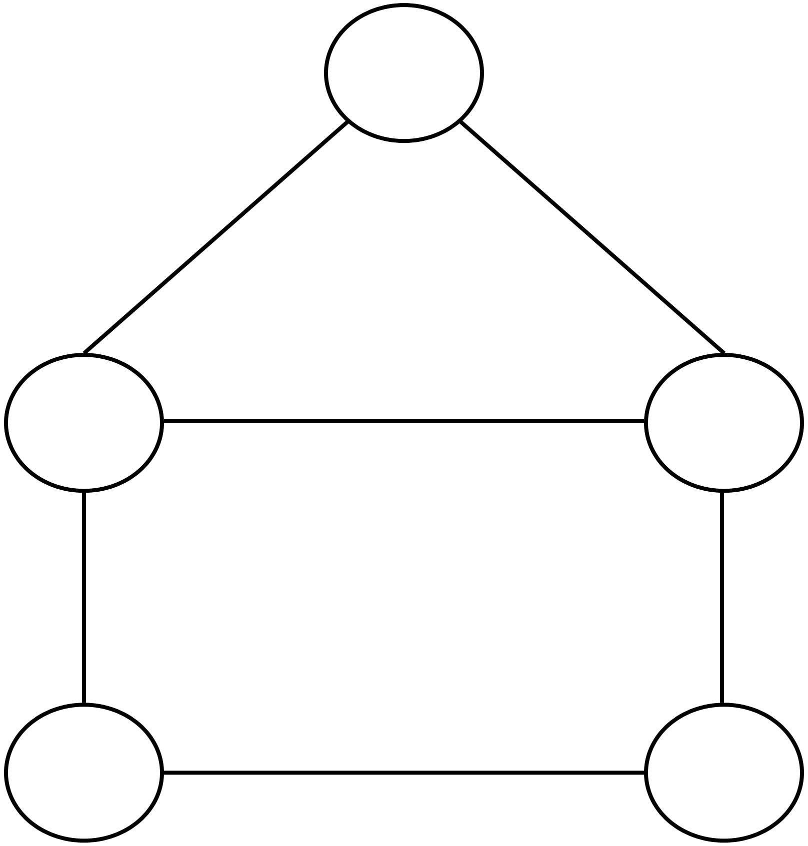
What is an appropriate interpretability query language for **labeled graphs** and **GNNs**?

- Which can be evaluated in polynomial time, or using SAT solvers
- What is an appropriate model-agnostic interpretability query language?
- What is an appropriate model-specific interpretability query language?

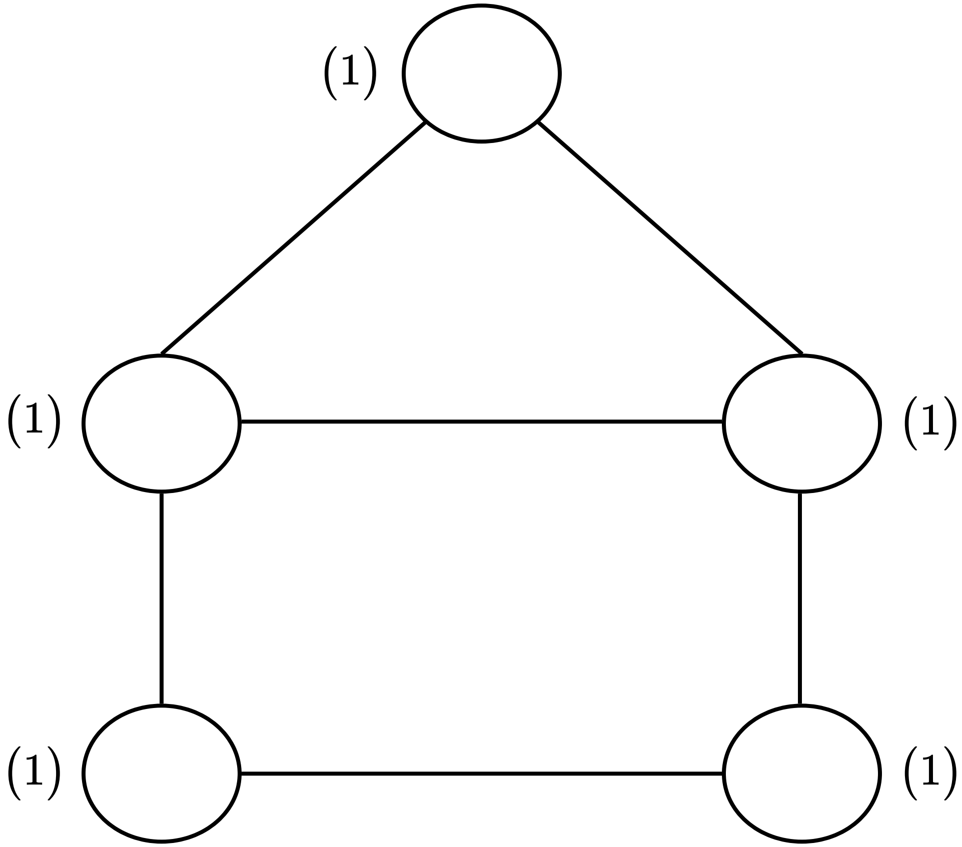
Thanks!

Backup slides

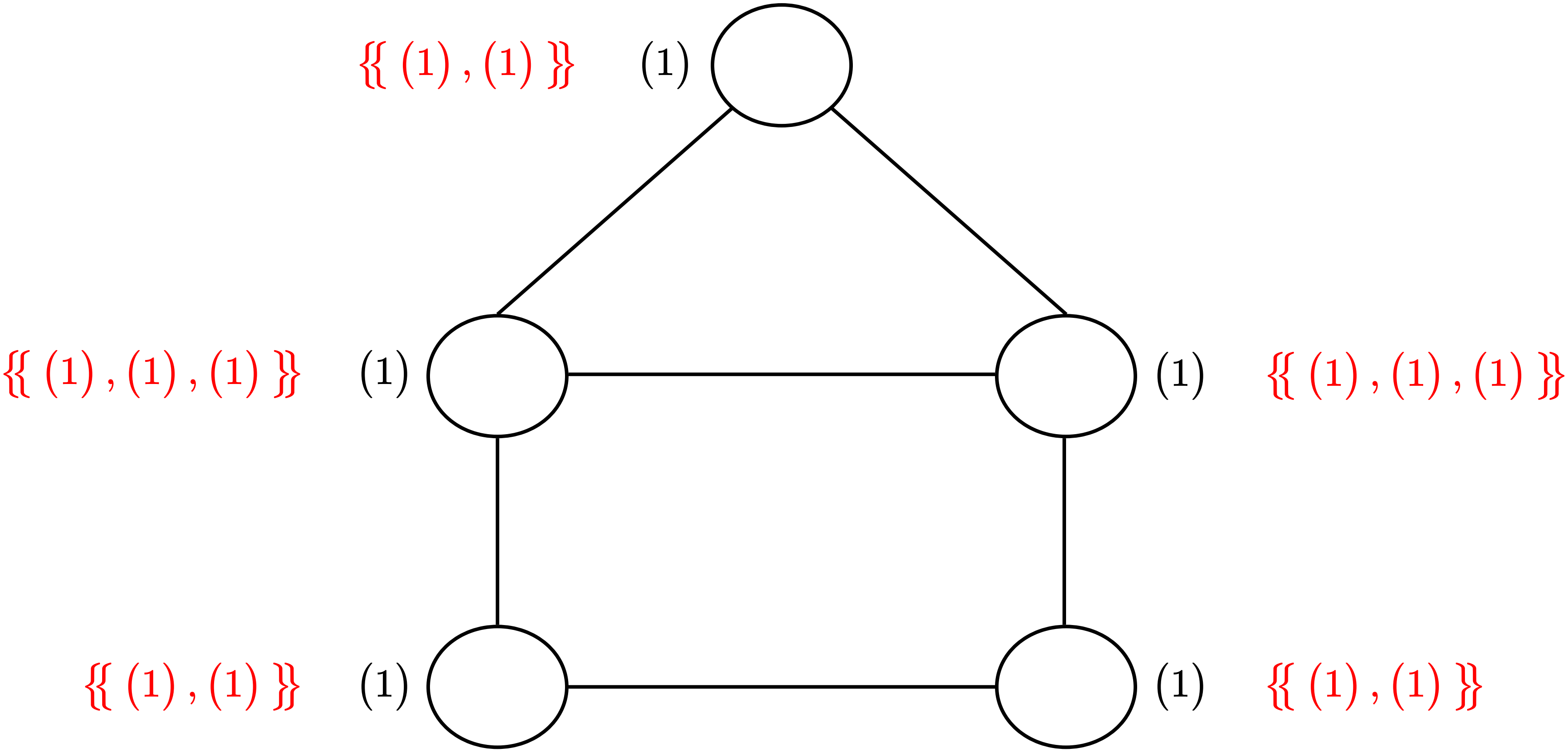
WL test for graph isomorphism



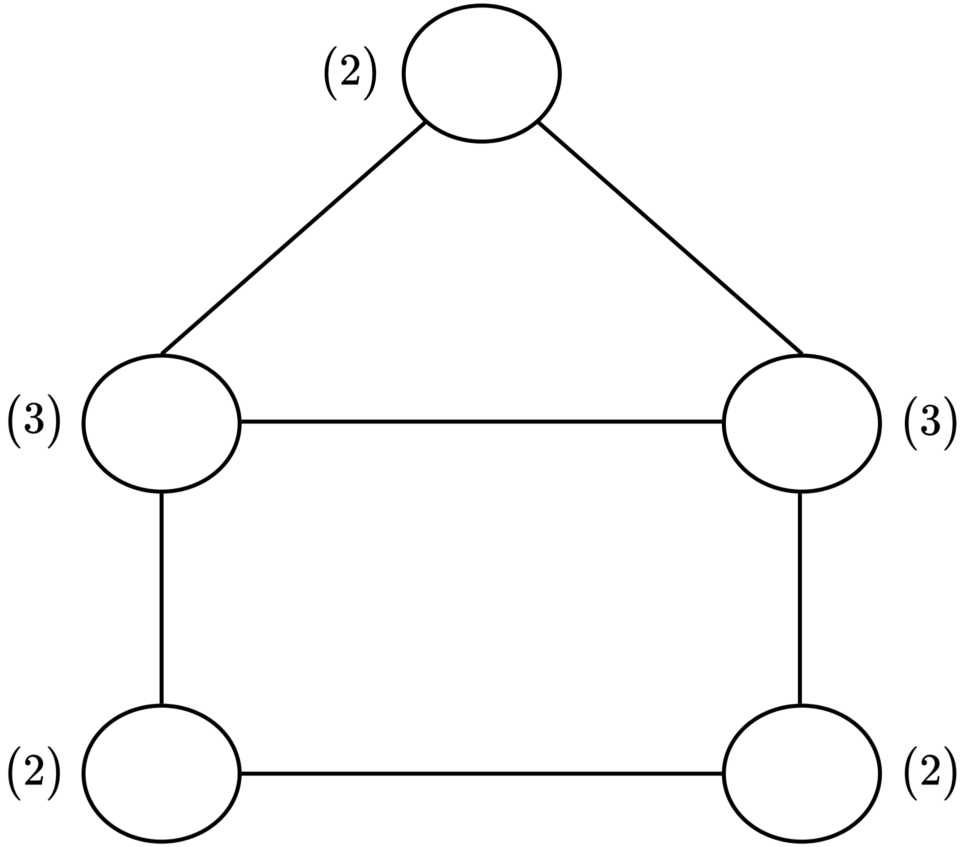
WL test for graph isomorphism



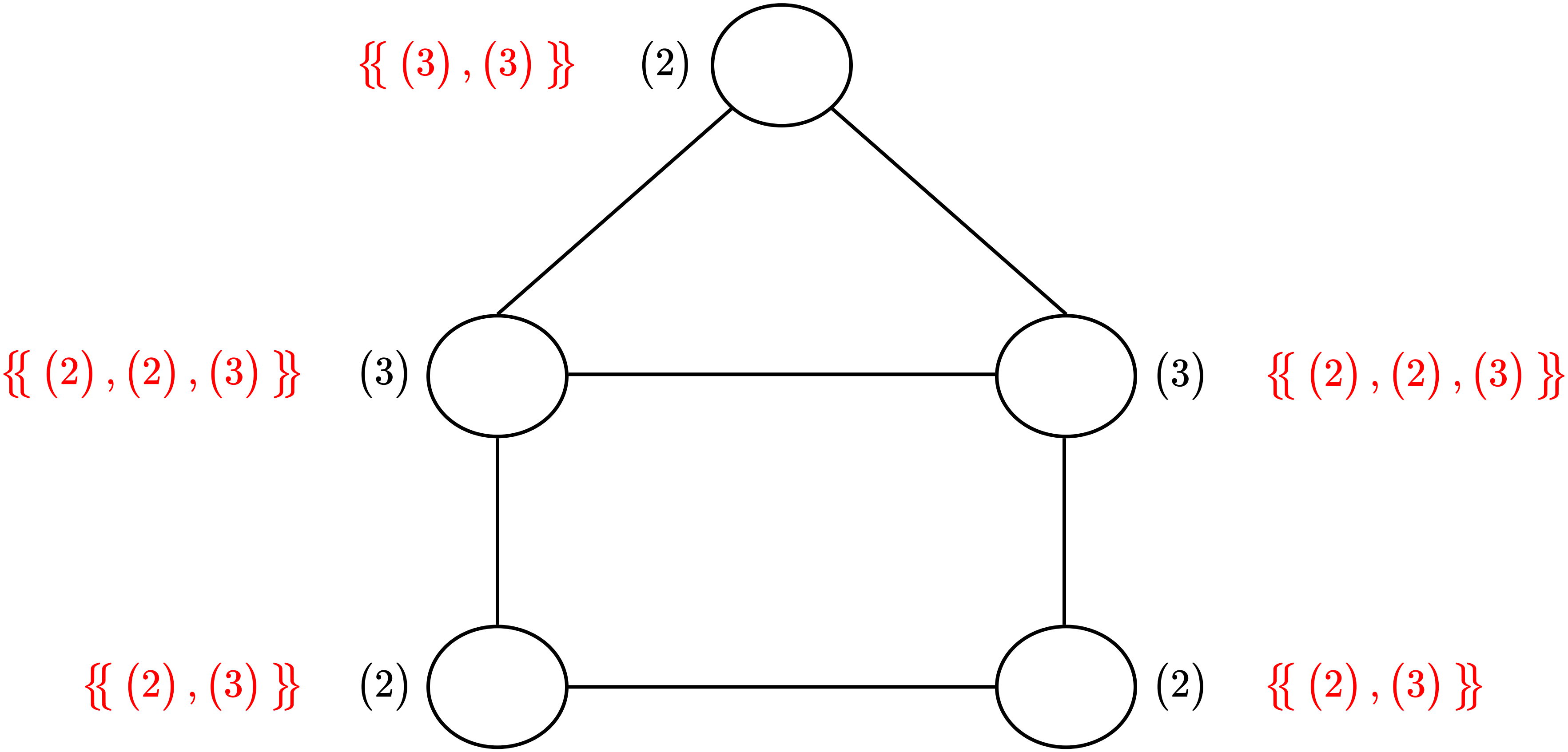
WL test for graph isomorphism



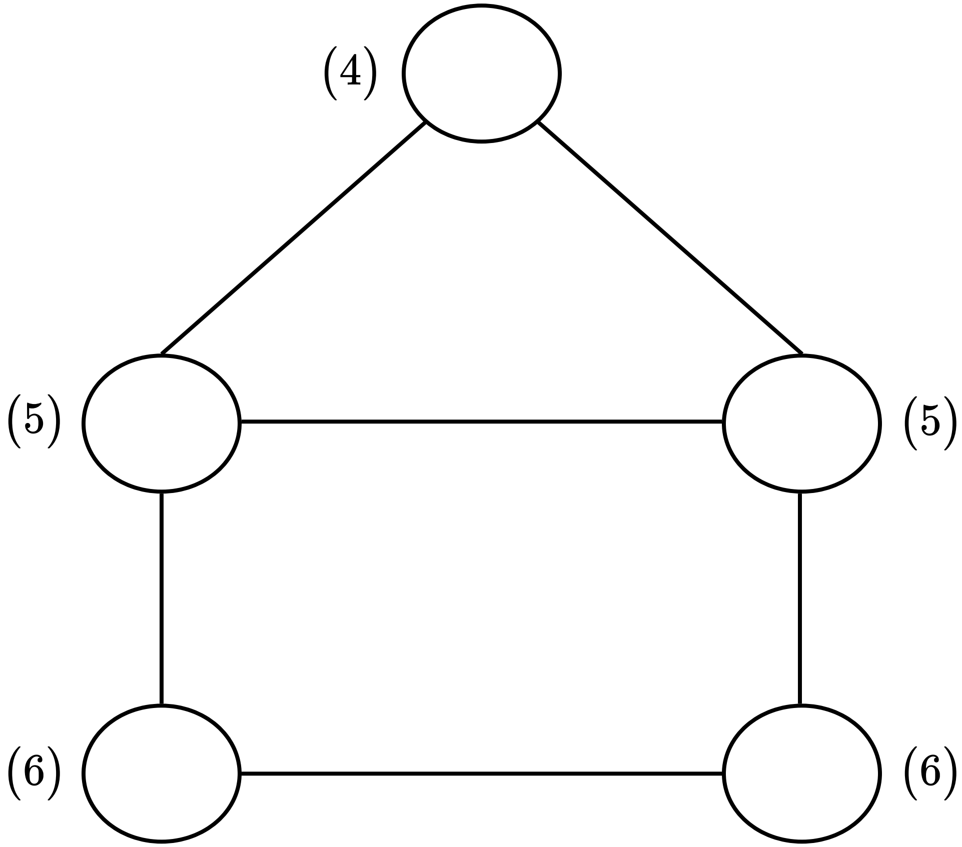
WL test for graph isomorphism



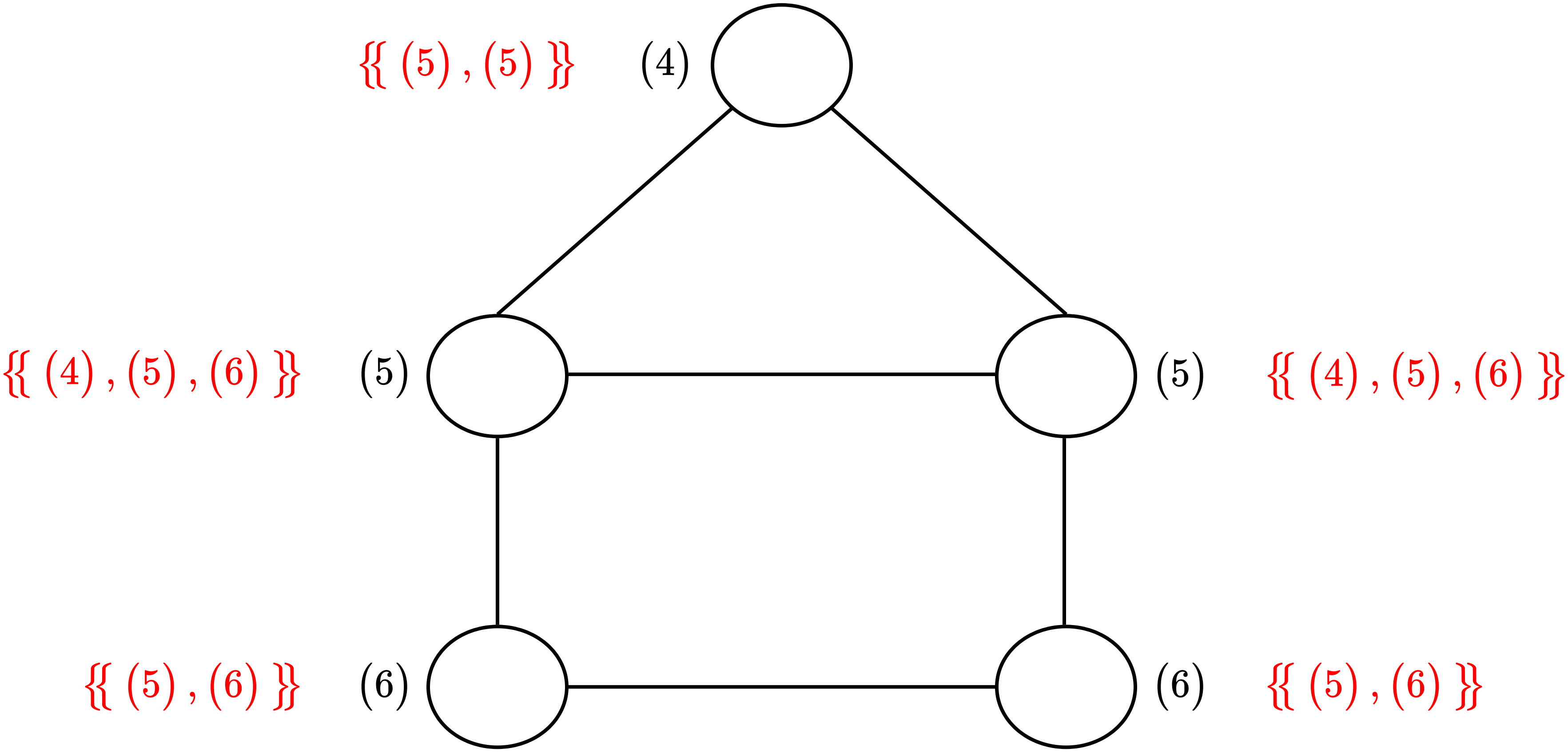
WL test for graph isomorphism



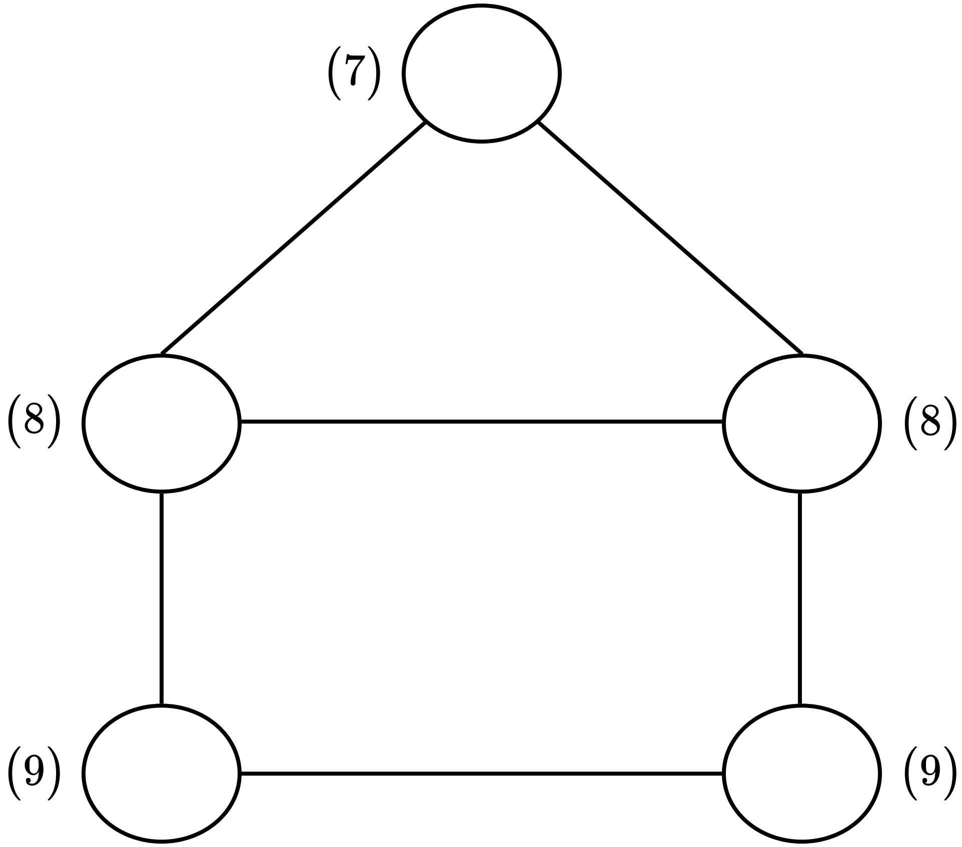
WL test for graph isomorphism



WL test for graph isomorphism



WL test for graph isomorphism



WL test for graph isomorphism