First-order Logic

Alcino Cunha

Overview

Terminology

- First-order logic uses **quantified variables** to express properties over a **domain** (or universe) of discourse
- It also uses **predicates** to capture relationships between elements of the domain
- First-order logic is also known as predicate logic

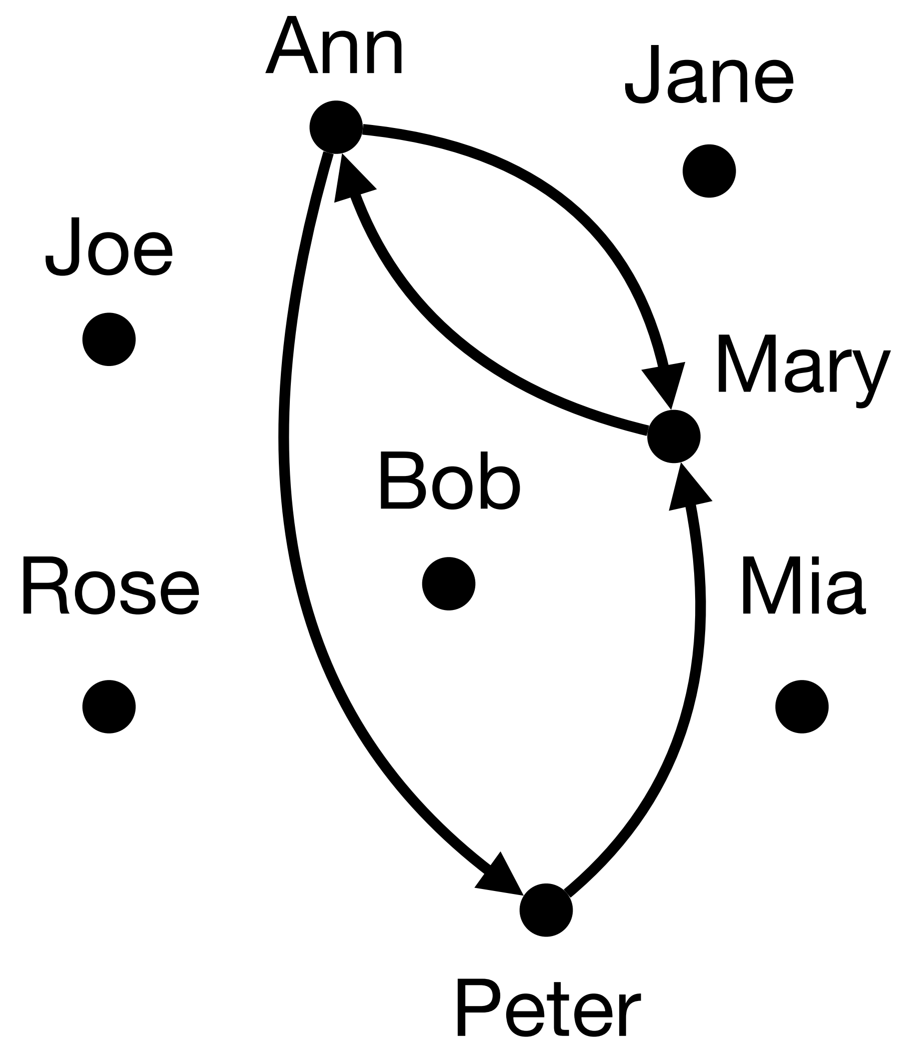
Predicates

- Predicates are **relations**, sets of tuples of elements of the domain
- All tuples have the same length, the **arity** of the predicate
- **Binary** predicates (of arity 2) represent relationships between elements
- **Unary** predicates (of arity 1) represent sets of elements

Binary predicates

$$\text{friend} = \{(\text{Ann}, \text{Peter}), (\text{Ann}, \text{Mary}), (\text{Mary}, \text{Ann}), (\text{Peter}, \text{Mary})\}$$

friend	
Ann	Peter
Ann	Mary
Mary	Ann
Peter	Mary

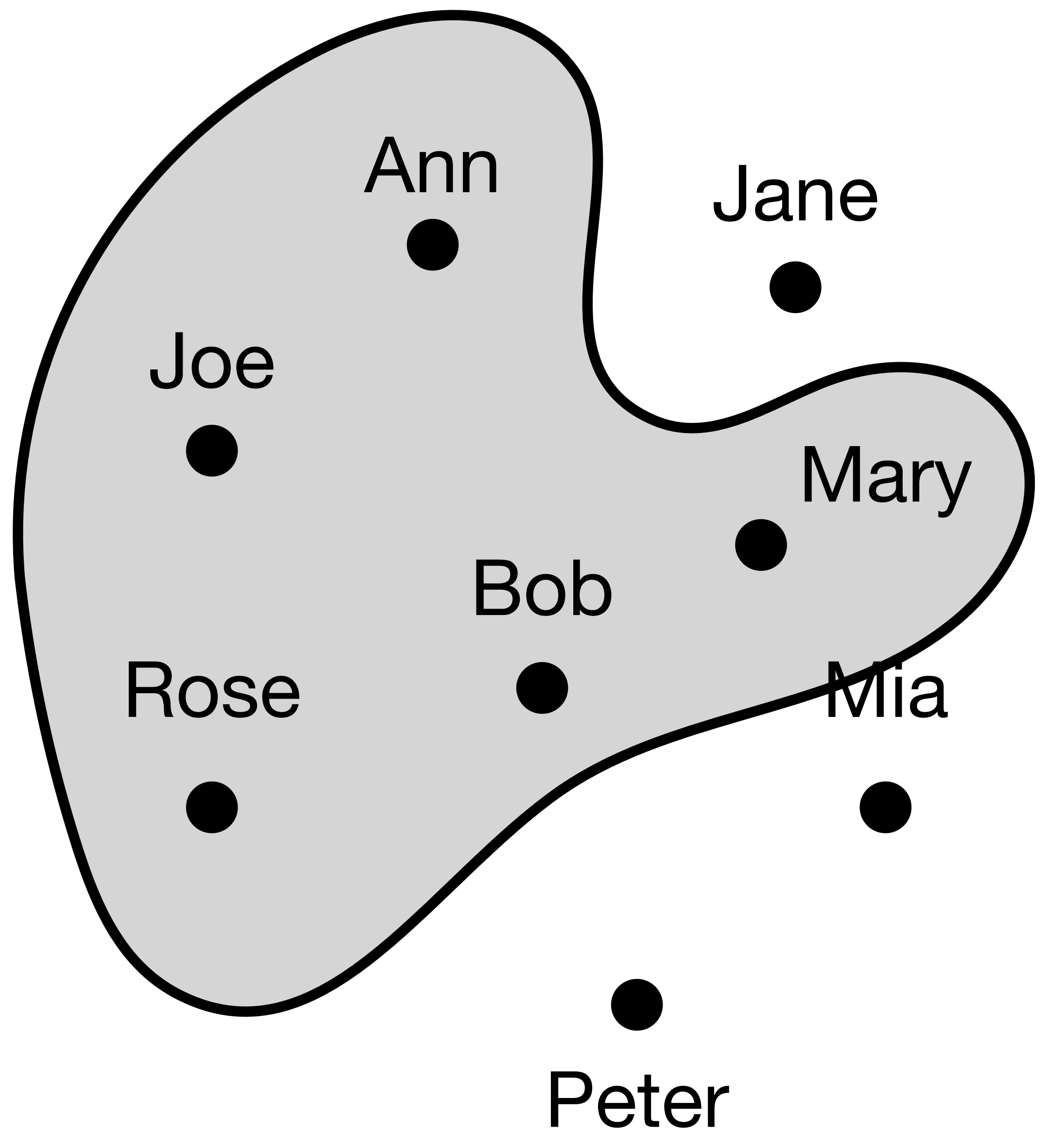


friend(Ann, Peter)

Unary predicates

$$\text{Student} = \{(\text{Ann}), (\text{Mary}), (\text{Joe}), (\text{Bob}), (\text{Rose})\}$$

Student
Ann
Mary
Joe
Bob
Rose



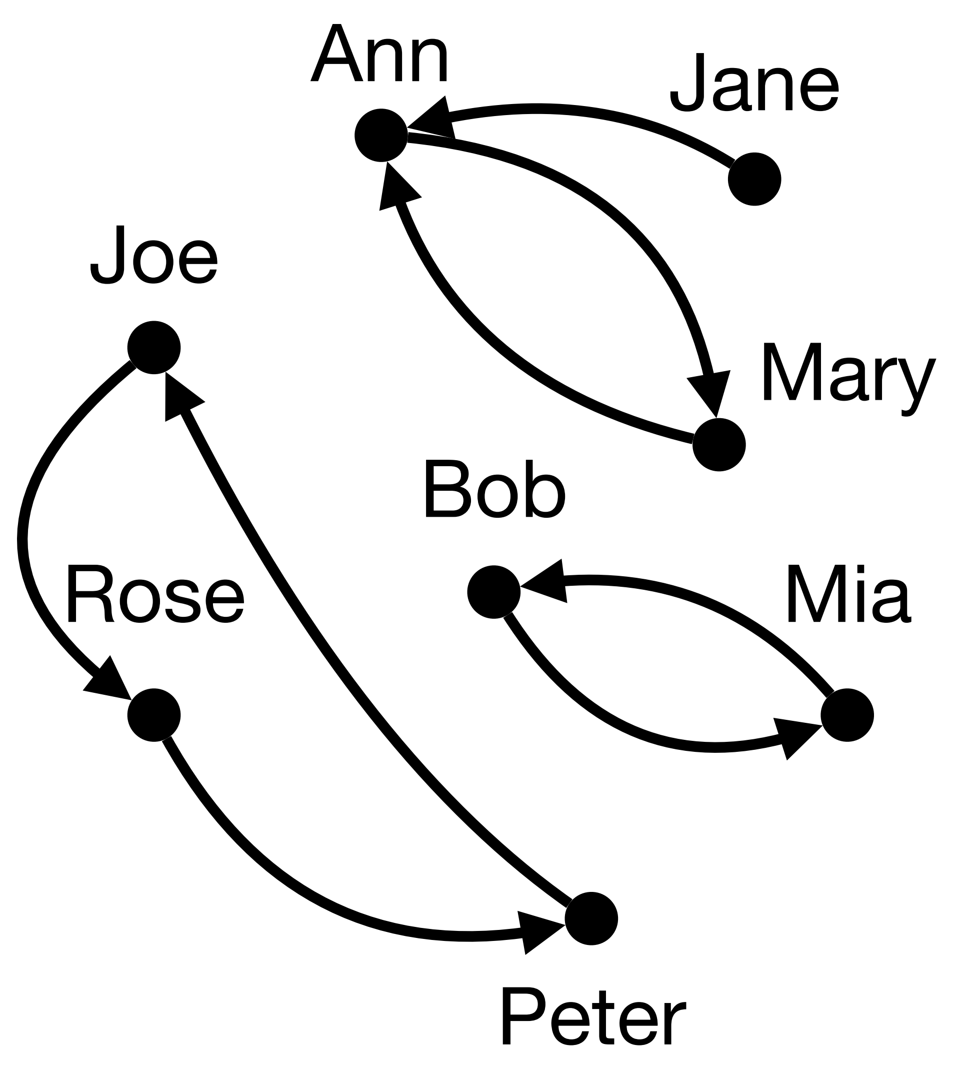
Student(Ann)

Functions, constants, and terms

- **Functions** are special relationships between tuples of elements and exactly one element
- The number of elements in the input is the arity of the function
- **Constants** denote specific elements of the domain
- With functions, constants, and variables we can build **terms** that represent specific elements of the domain

Unary functions

$$\text{bff} = \{\text{Ann} \mapsto \text{Mary}, \text{Mary} \mapsto \text{Ann}, \text{Peter} \mapsto \text{Joe}, \text{Joe} \mapsto \text{Rose}, \text{Rose} \mapsto \text{Peter}, \text{Bob} \mapsto \text{Mia}, \text{Jane} \mapsto \text{Ann}, \text{Mia} \mapsto \text{Bob}\}$$



$$\text{Rose} = \text{bff}(\text{bff}(\text{Peter}))$$

Syntax

- Variables: $x, y, z, \ldots$
- Constants: $a, b, c, \ldots$
- Functions: $f, g, h, \ldots$
- Predicates: $P, Q, R, \ldots$
- Logic connectives: $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\forall$ (for all), $\exists$ (exists)
- Equality: $=$
- Auxiliary symbols: parenthesis, dot

Syntax

$$
\begin{aligned}
x, y, z, \ldots &\in \mathscr{X} \\
a, b, c, \ldots &\in \mathscr{C} \\
f, g, h, \ldots &\in \mathscr{F} \\
P, Q, R, \ldots &\in \mathscr{P} \\
\mathscr{V} &= \mathscr{C} \cup \mathscr{F} \cup \mathscr{P} \\
t, u, \ldots &\in \mathbf{Term}_{\mathscr{V}} \\
\phi, \psi, \ldots &\in \mathbf{Form}_{\mathscr{V}}
\end{aligned}
$$

$$
\begin{aligned}
\phi, \psi \doteq\ & P(t_1, \ldots, t_{|P|}) \\
&\mid t = u \\
&\mid \top \\
&\mid \bot \\
&\mid (\neg \phi) \\
&\mid (\phi \wedge \psi) \\
&\mid (\phi \vee \psi) \\
&\mid (\phi \rightarrow \psi) \\
&\mid (\phi \leftrightarrow \psi) \\
&\mid (\forall x \,.\, \phi) \\
&\mid (\exists x \,.\, \phi)
\end{aligned}
$$

$$
\begin{aligned}
t, u \doteq\ & x, y, z, \ldots \\
&\mid a, b, c, \ldots \\
&\mid f(t_1, \ldots, t_{|f|})
\end{aligned}
$$

Examples

- Ann is the bff of Mary — $\text{Ann} = \text{bff}(\text{Mary})$
- Ann is friend of everyone — $\forall x\,.\,\text{friend}(\text{Ann}, x)$
- Friendship is symmetric — $\forall x\,.\,\forall y\,.\,\text{friend}(x, y) \to \text{friend}(y, x)$
- Bffs are friends — $\forall x\,.\,\text{friend}(x, \text{bff}(x))$
- Everyone has a student friend — $\forall x\,.\,\exists y\,.\,\text{Student}(y) \wedge \text{friend}(x, y)$

Functions vs predicates

- Functions and constants simplify the writing of formulas but are not strictly necessary

- A function f of arity n can be represented by a predicate of arity $n + 1$ with additional constraints

$$\forall x \,.\, \exists y \,.\, f(x, y)$$

$$\forall x \,.\, \forall y \,.\, \forall z \,.\, f(x, y) \wedge f(x, z) \rightarrow y = z$$

- A constant a is just a function of arity 0 and can also be represented by a predicate of arity 1 with additional constraints

$$\exists x \,.\, a(x)$$

$$\forall x \,.\, \forall y \,.\, a(x) \wedge a(y) \rightarrow x = y$$

Functions vs predicates

$\forall x \,.\, \text{friend}(x, \text{bff}(x))$

$\forall x \,.\, \forall y \,.\, \text{bff}(x, y) \rightarrow \text{friend}(x, y)$

$\forall x \,.\, \text{friend}(\text{Ann}, x)$

$\forall x \,.\, \text{Ann}(x) \rightarrow \forall y \,.\, \text{friend}(x, y)$

$\exists x \,.\, \text{Ann}(x) \wedge \forall y \,.\, \text{friend}(x, y)$

$\text{Ann} = \text{bff}(\text{Mary})$

$\forall x \,.\, \forall y \,.\, \text{Ann}(x) \wedge \text{Mary}(y) \rightarrow \text{bff}(x, y)$

$\exists x \,.\, \exists y \,.\, \text{Ann}(x) \wedge \text{Mary}(y) \wedge \text{bff}(x, y)$

Simplified syntax

$$
\begin{aligned}
x, y, z, \ldots &\in \mathscr{X} \\
P, Q, R, \ldots &\in \mathscr{P} \\
\mathscr{V} &= \mathscr{P} \\
\phi, \psi, \ldots &\in \mathbf{Form}_{\mathscr{V}}
\end{aligned}
$$

$$
\begin{aligned}
\phi, \psi \doteq\ & P(x_1, \ldots, x_{|P|}) \\
& \mid x = y \\
& \mid \top \\
& \mid \bot \\
& \mid (\neg \phi) \\
& \mid (\phi \wedge \psi) \\
& \mid (\phi \vee \psi) \\
& \mid (\phi \rightarrow \psi) \\
& \mid (\phi \leftrightarrow \psi) \\
& \mid (\forall x \,.\, \phi) \\
& \mid (\exists x \,.\, \phi)
\end{aligned}
$$

Semantics

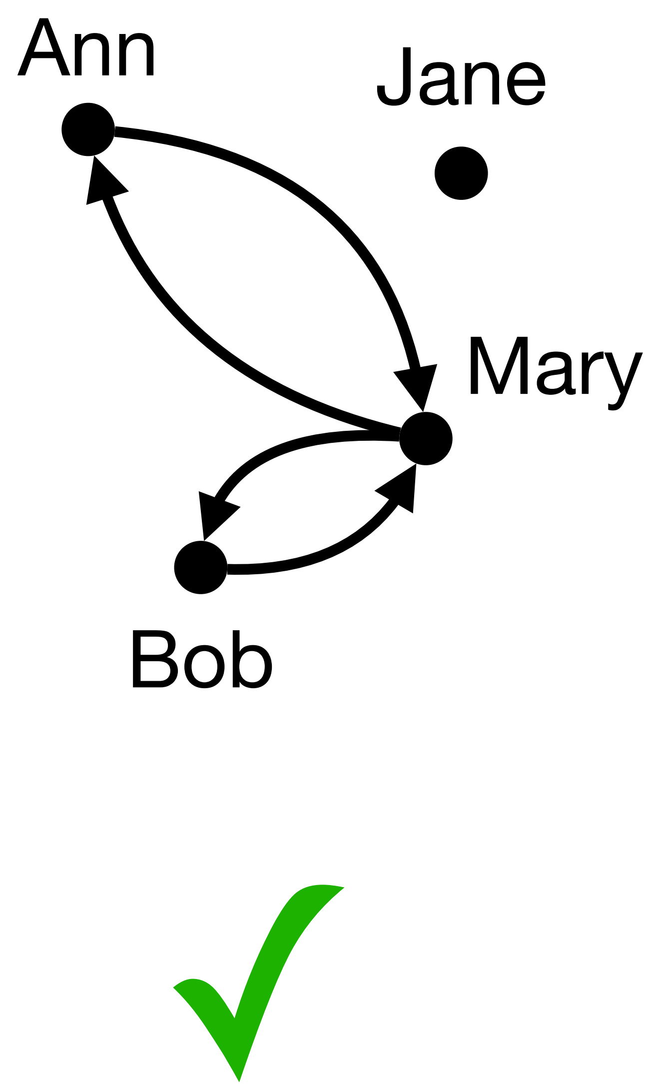
- To determine the truth value of a formula we need a **structure** $\mathscr{M} = (D, I)$
  - D is a set with the **domain** of discourse
  - I is an **interpretation** for predicates, for each $P \in \mathscr{P}$ we have $I(P) \subseteq D^{|P|}$
- We also need an assignment $\mathscr{A} : \mathscr{X} \mapsto D$ with the value of the free variables
  - A variable is **free** if it is not associated with a quantifier, otherwise it is **bound**
  - A formula without free variables is **closed**
- The fact that ϕ holds under $\mathscr{M}$ with $\mathscr{A}$ is denoted by $\mathscr{M}, \mathscr{A} \vDash \phi$

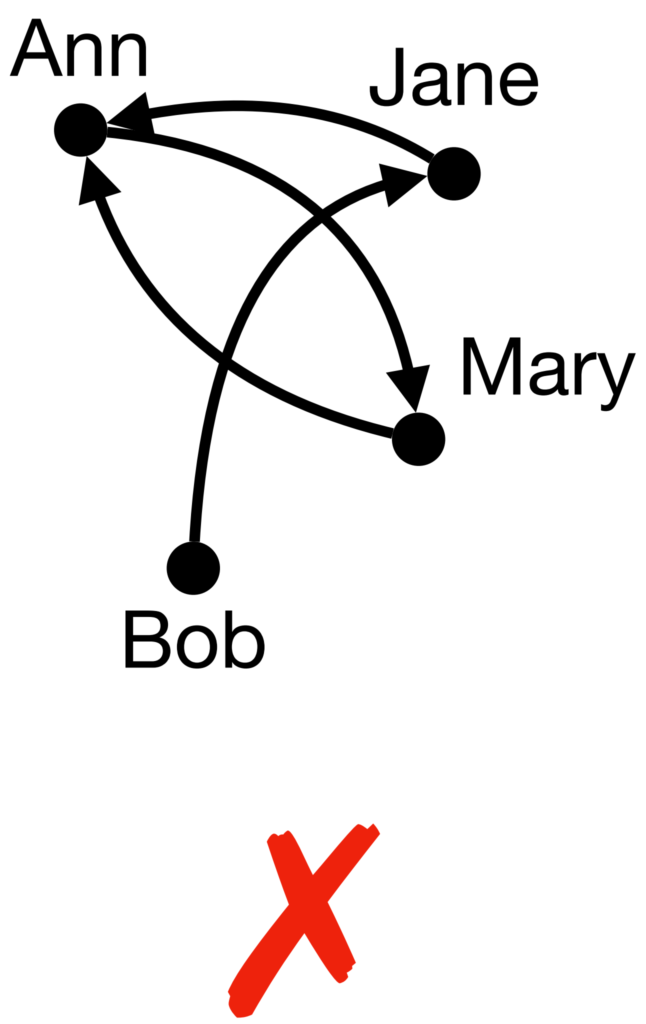
Inductive semantics

$\mathscr{M}, \mathscr{A} \vDash \top$		
$\mathscr{M}, \mathscr{A} \nvDash \bot$		
$\mathscr{M}, \mathscr{A} \vDash P(x_1, \ldots, x_n)$	iff	$(\mathscr{A}(x_1), \ldots, \mathscr{A}(x_n)) \in I(P)$
$\mathscr{M}, \mathscr{A} \vDash x = y$	iff	$\mathscr{A}(x)$ is equal to $\mathscr{A}(y)$
$\mathscr{M}, \mathscr{A} \vDash \neg\phi$	iff	$\mathscr{M}, \mathscr{A} \nvDash \phi$
$\mathscr{M}, \mathscr{A} \vDash \phi \wedge \psi$	iff	$\mathscr{M}, \mathscr{A} \vDash \phi$ and $\mathscr{M}, \mathscr{A} \vDash \psi$
$\mathscr{M}, \mathscr{A} \vDash \phi \vee \psi$	iff	$\mathscr{M}, \mathscr{A} \vDash \phi$ or $\mathscr{M}, \mathscr{A} \vDash \psi$
$\mathscr{M}, \mathscr{A} \vDash \phi \to \psi$	iff	$\mathscr{M}, \mathscr{A} \nvDash \phi$ or $\mathscr{M}, \mathscr{A} \vDash \psi$
$\mathscr{M}, \mathscr{A} \vDash \phi \leftrightarrow \psi$	iff	$\mathscr{M}, \mathscr{A} \vDash \phi$ iff $\mathscr{M}, \mathscr{A} \vDash \psi$
$\mathscr{M}, \mathscr{A} \vDash \forall x\,.\,\phi$	iff	$\mathscr{M}, \mathscr{A}[x \mapsto a] \vDash \phi$ for all $a \in D$
$\mathscr{M}, \mathscr{A} \vDash \exists x\,.\,\phi$	iff	$\mathscr{M}, \mathscr{A}[x \mapsto a] \vDash \phi$ for some $a \in D$

Example

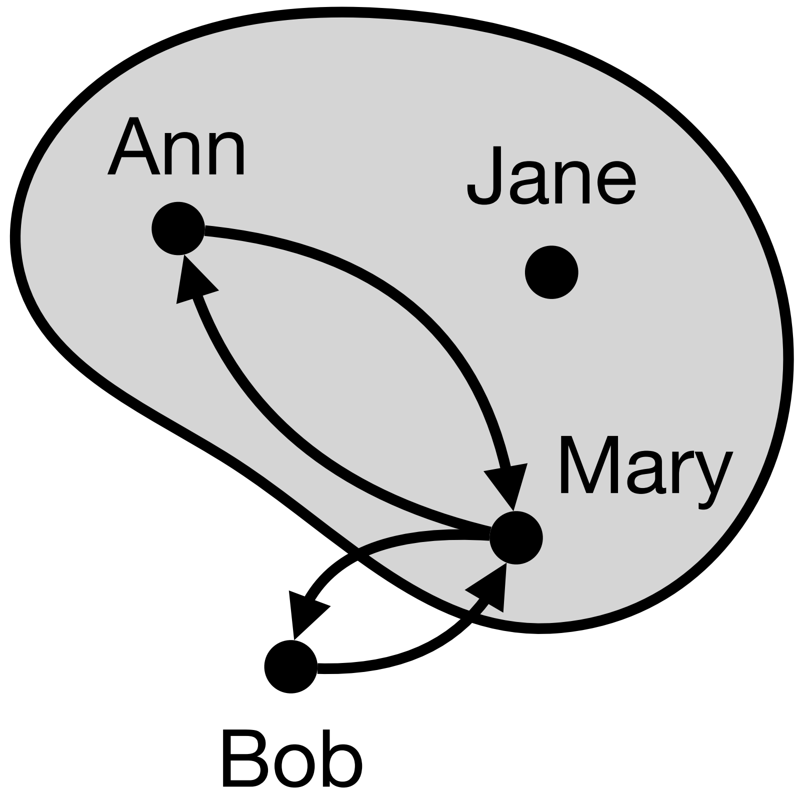
$$\forall x \,.\, \forall y \,.\, \mathrm{friend}(x, y) \rightarrow \mathrm{friend}(y, x)$$



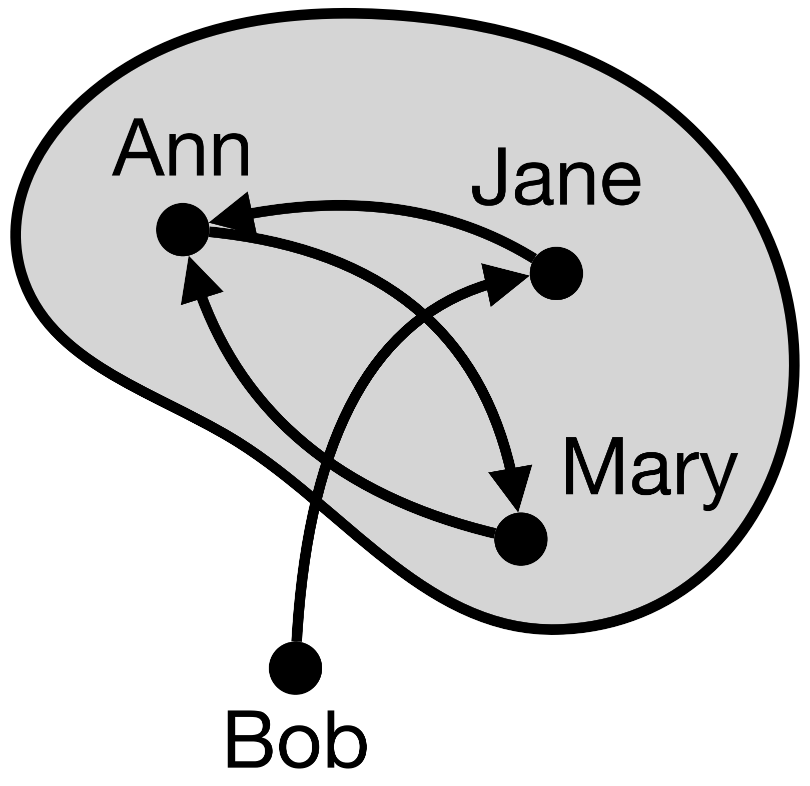


Example

$$\forall x \,.\, \exists y \,.\, \mathrm{Student}(y) \wedge \mathrm{friend}(x, y)$$



✗



✓

More terminology

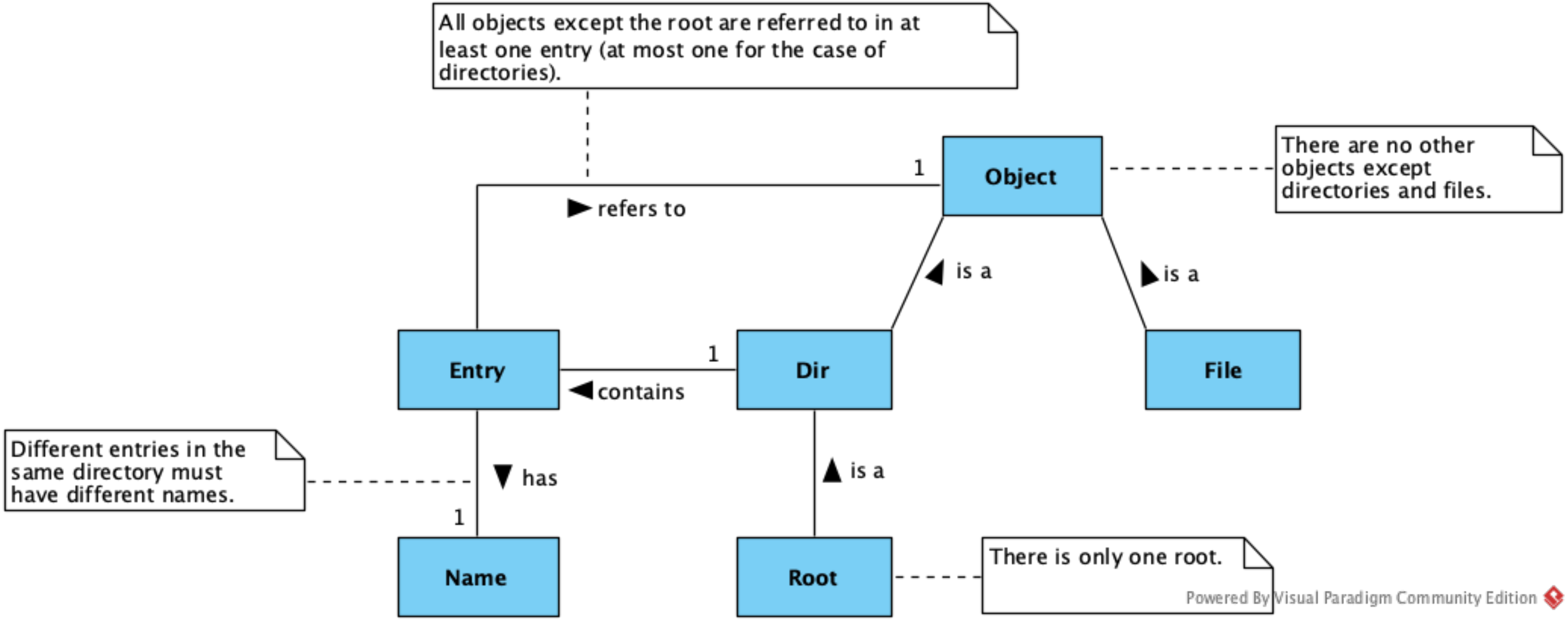
- A formula ϕ is
  - **valid** or a **tautology** iff it holds under all interpretations with all assignments
  - **satisfiable** iff it holds under some interpretation with some assignment
  - **unsatisfiable** or a **contradiction** iff it does not hold under all interpretations with all assignments
  - **refutable** iff it does not hold under some interpretation with some assignment

Decidability

- The decision problem “Is ϕ satisfiable?” when ϕ is a first-order formula is **undecidable**
  - Solvers for FOL may not terminate or answer with an unknown
- There are two strategies to “recover” decidability
  - **Bound** the domain of analysis up to a finite scope (effectively reducing FOL to PL)
  - Work on subsets of FOL (**theories**) that are decidable, e.g. linear arithmetic

Domain modelling

Domain modeling *a la* UML



Domain model analysis

- Are the requirements consistent?
- Any forgotten or redundant requirements?
- Do the requirements entail all the expected properties?

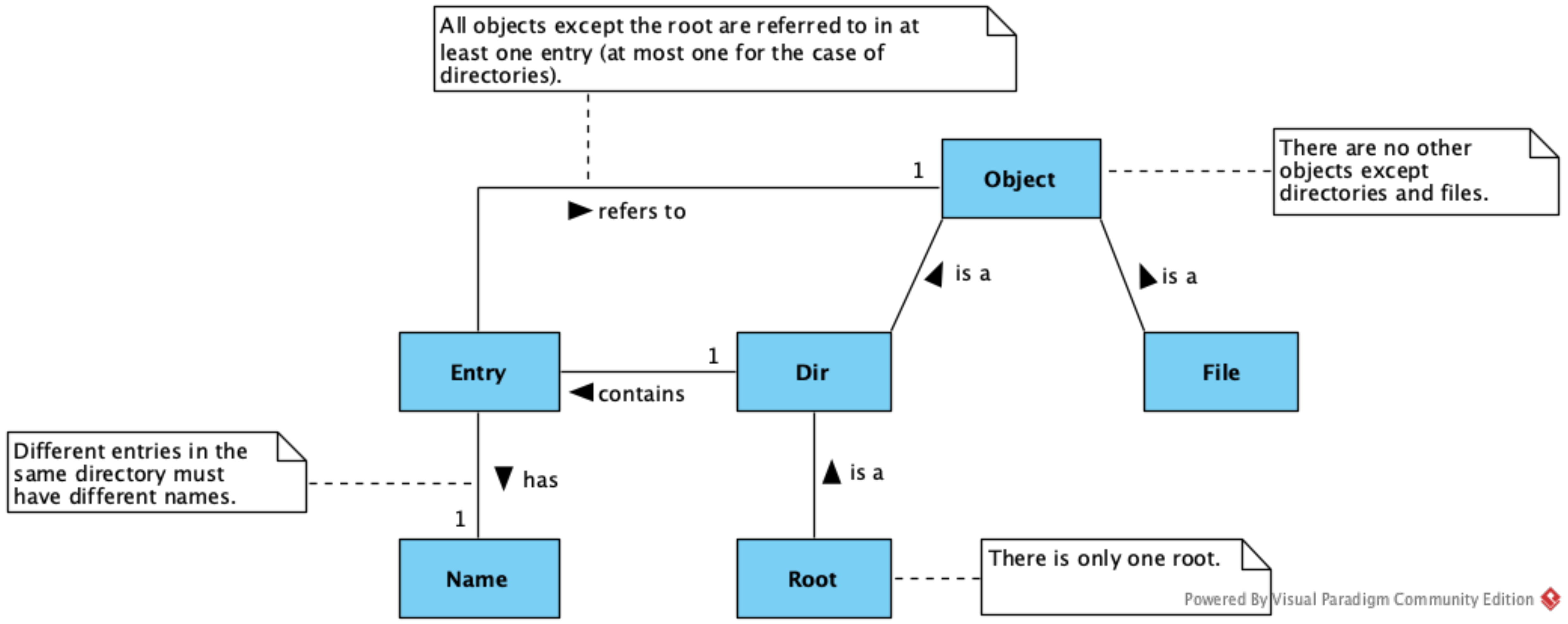
Entities

- The “is a” relationship denotes a specialization or extension
- Formally it can be seen as a subset relationship
- All entities in a diagram not related by “is a” are disjoint
  - Top-level entities are disjoint
  - Multiple extensions of the same entity are disjoint

Formalizing entities

- Each entity can be formalized by a unary predicate
- Constraints should be added to specify
    - The subset relationship of extension signatures
    - The disjointness of all other signatures

Formalizing entities



$$\text{Object} \subseteq D$$

$$\text{File} \subseteq D$$

$$\text{Dir} \subseteq D$$

$$\text{Root} \subseteq D$$

$$\text{Entry} \subseteq D$$

$$\text{Name} \subseteq D$$

Formalizing entities

$$\forall x \,.\, \neg(\mathrm{Object}(x) \land \mathrm{Entry}(x))$$

$$\forall x \,.\, \neg(\mathrm{Object}(x) \land \mathrm{Name}(x))$$

$$\forall x \,.\, \neg(\mathrm{Name}(x) \land \mathrm{Entry}(x))$$

$$\forall x \,.\, \mathrm{File}(x) \rightarrow \mathrm{Object}(x)$$

$$\forall x \,.\, \mathrm{Dir}(x) \rightarrow \mathrm{Object}(x)$$

$$\forall x \,.\, \neg(\mathrm{File}(x) \land \mathrm{Dir}(x))$$

$$\forall x \,.\, \mathrm{Root}(x) \rightarrow \mathrm{Dir}(x)$$

Formalizing associations

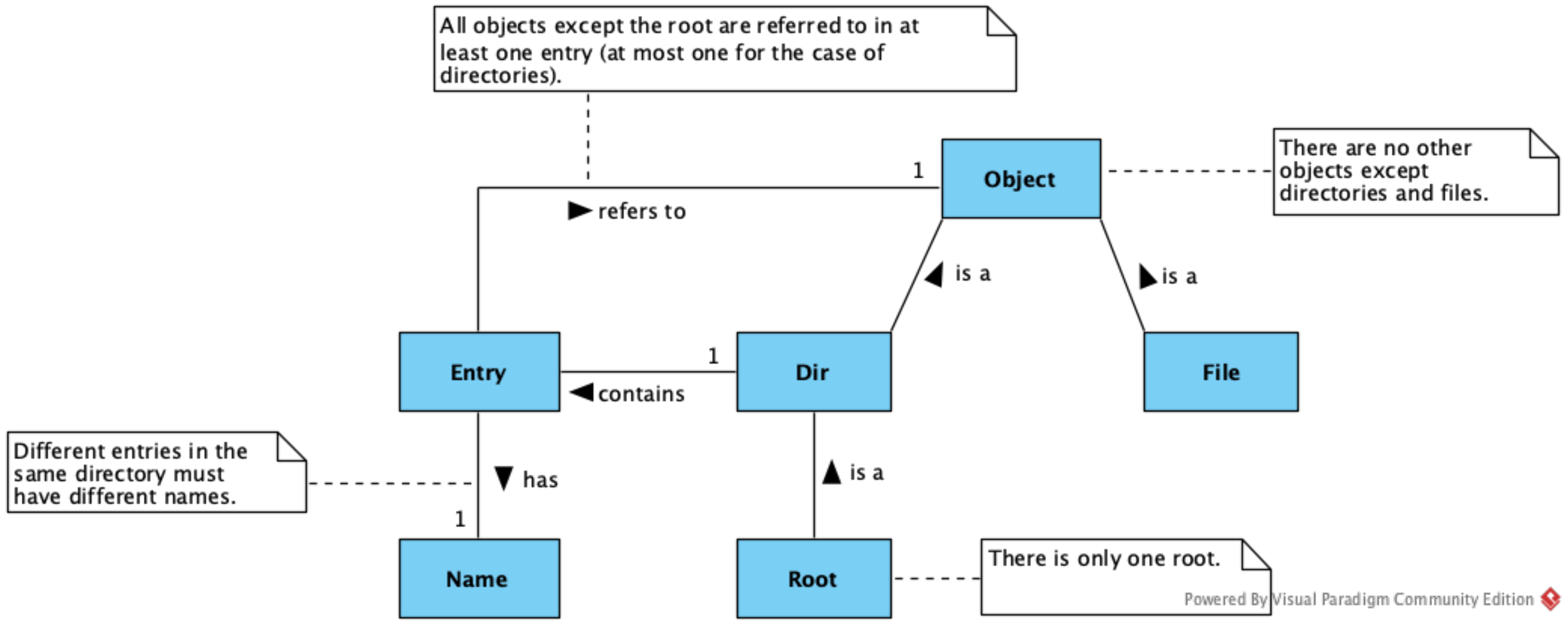
- Each association relationship can be formalized by a predicate
- Constraints should be added to specify
  - The type of related entities
  - The multiplicity restrictions at each end

Bounded quantifiers

$$\forall x : A \,.\, \phi \quad \equiv \quad \forall x \,.\, A(x) \to \phi$$
$$\exists x : A \,.\, \phi \quad \equiv \quad \exists x \,.\, A(x) \wedge \phi$$

$$\forall x : A, y : B \,.\, \phi \quad \equiv \quad \forall x : A \,.\, \forall y : B \,.\, \phi$$
$$\forall x, y : A \,.\, \phi \quad \equiv \quad \forall x : A, y : A \,.\, \phi$$
$$\exists x : A, y : B \,.\, \phi \quad \equiv \quad \exists x : A \,.\, \exists y : B \,.\, \phi$$
$$\exists x, y : A \,.\, \phi \quad \equiv \quad \exists x : A, y : A \,.\, \phi$$

Formalizing associations



$$\text{contains} \subseteq D \times D$$

$$\text{refersTo} \subseteq D \times D$$

$$\text{has} \subseteq D \times D$$

Syntactic sugar

$$\forall x : A \,.\, \phi \quad \equiv \quad \forall x \,.\, A(x) \to \phi$$

$$\exists x : A \,.\, \phi \quad \equiv \quad \exists x \,.\, A(x) \wedge \phi$$

$$\forall x : A, y : B \,.\, \phi \quad \equiv \quad \forall x : A \,.\, \forall y : B \,.\, \phi$$

$$\forall x, y : A \,.\, \phi \quad \equiv \quad \forall x : A, y : A \,.\, \phi$$

$$\exists x : A, y : B \,.\, \phi \quad \equiv \quad \exists x : A \,.\, \exists y : B \,.\, \phi$$

$$\exists x, y : A \,.\, \phi \quad \equiv \quad \exists x : A, y : A \,.\, \phi$$

Formalizing associations

$$\forall x, y \,.\, \text{contains}(x, y) \rightarrow \text{Dir}(x) \land \text{Entry}(y)$$

$$\forall x, y \,.\, \text{refersTo}(x, y) \rightarrow \text{Entry}(x) \land \text{Object}(y)$$

$$\forall x, y \,.\, \text{has}(x, y) \rightarrow \text{Entry}(x) \land \text{Name}(y)$$

$$\forall x : \text{Entry} \,.\, \exists y \,.\, \text{has}(x, y)$$

$$\forall x, y, z \,.\, \text{has}(x, y) \land \text{has}(x, z) \rightarrow y = z$$

$$\forall x : \text{Entry} \,.\, \exists y \,.\, \text{refersTo}(x, y)$$

$$\forall x, y, z \,.\, \text{refersTo}(x, y) \land \text{refersTo}(x, z) \rightarrow y = z$$

$$\forall x : \text{Entry} \,.\, \exists y \,.\, \text{contains}(y, x)$$

$$\forall x, y, z \,.\, \text{contains}(y, x) \land \text{contains}(z, x) \rightarrow y = z$$

Specifying requirements

- There is only one root

$$\exists x \,.\, \mathrm{Root}(x)$$

$$\forall x, y : \mathrm{Root} \,.\, x = y$$

- There are no other objects except directories and files

$$\forall x : \mathrm{Object} \,.\, \mathrm{Dir}(x) \vee \mathrm{File}(x)$$

- Different entries in the same directory must have different names

$$\forall x, y, z, w \,.\, \mathrm{contains}(x, y) \wedge \mathrm{contains}(x, z) \wedge \mathrm{has}(y, w) \wedge \mathrm{has}(z, w) \rightarrow y = z$$

Specifying requirements

- All objects except the root are referred to in at least one entry (at most one for the case of directories)
  - All objects except the root are referred to in at least one entry

$$\forall x : \text{Object} . \neg \text{Root}(x) \rightarrow \exists y . \text{refersTo}(y, x)$$

  - The root is not referred in any entry

$$\forall x : \text{Entry}, y : \text{Root} . \neg \text{refersTo}(x, y)$$

  - All directories are referred to in at most one entry

$$\forall x : \text{Dir} . \forall y, z . \text{refersTo}(y, x) \wedge \text{refersTo}(z, x) \rightarrow y = z$$

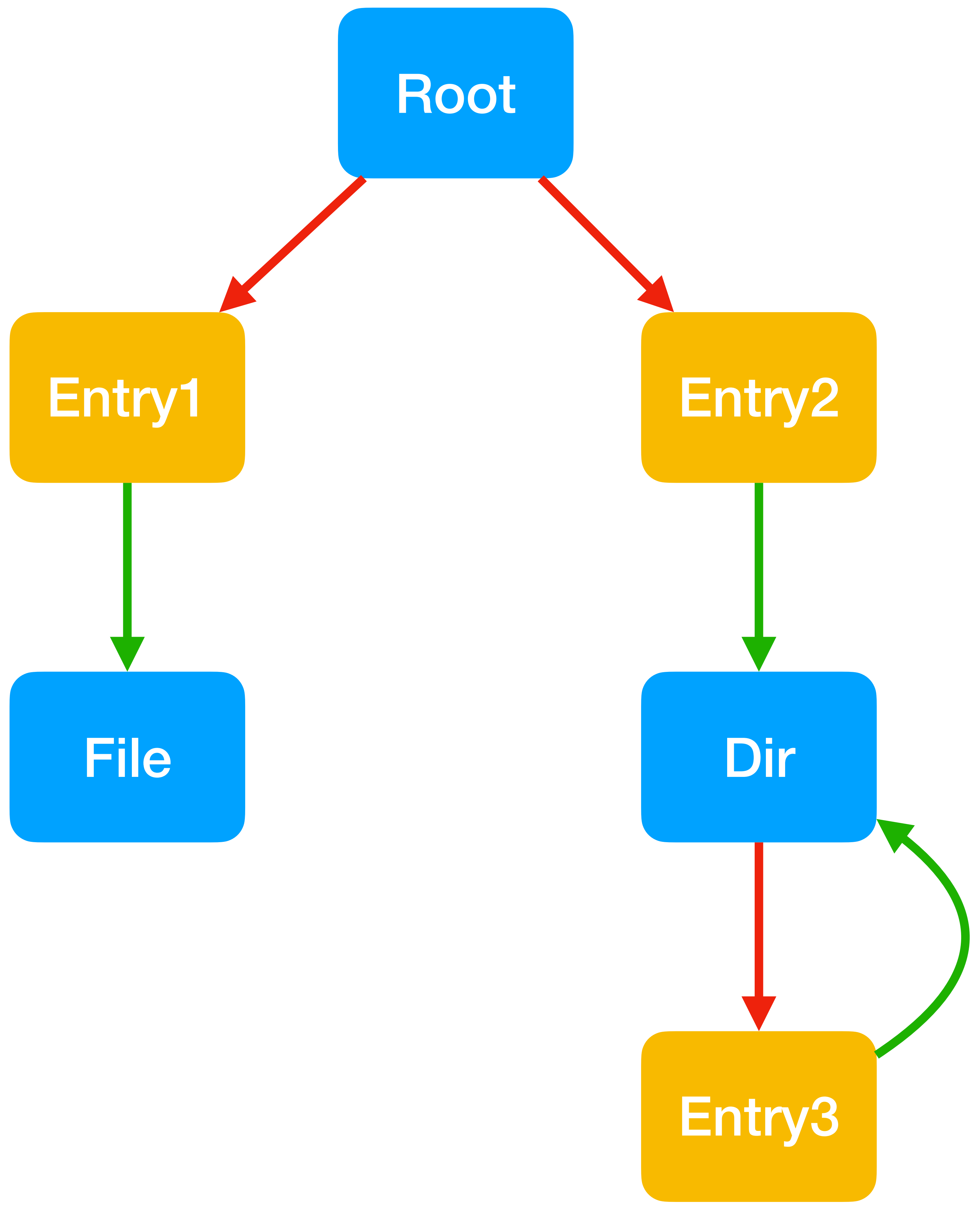
Domain model analysis

- Are the requirements consistent?
  - Check if the specified requirements are SAT
- Any forgotten or redundant requirements?
  - Inspect specific scenarios
- Do the requirements entail all the expected properties?
  - Check the validity of expected properties

Let's do it with Z3!

DEMO

Scenario depiction



Limitations

- Very tedious to formalize entities and associations
- Scenarios are very difficult to understand
- Analysis can diverge

Alloy