Back to the knights and knaves island

During your Thanksgiving trip, you found another island with knights and knaves (you have a feeling that you may encounter many such islands this semester). Knights always tell the truth and knaves always lie.

You talk to every person on the island and each one says "All of us on the island are of the same type".

Are they really the same type? Can you determine which type?

a. They are all knights.
b. They are all knaves.
c. Some are knights and some are knaves.

Predicate Logic: Informal Introduction and Translation

Alice Gao

October 12, 2017

These slides are based on materials from CS245 at UWaterloo and CPSC121 at UBC.

What can't we express using propositional logic?

Can we express the following ideas using propositional logic?

- Translate this sentence: Alice is married to Jay and Alice is not married to Leon.
- Translate this sentence: Every bear likes honey.
- Define what it means for a natural number to be prime.

What can't we express using propositional logic?

A few things that are difficult to express using propositional logic:

- Relationships among individuals: Alice is married to Jay and Alice is not married to Leon.
- Generalizing patterns: Every bear likes honey.
- Infinite domains: Define what it means for a natural number to be prime.

We can use predicate logic (first-order logic) to express all of these.

Would you really use predicate logic?

Examples of predicate logic in CS245 so far:

1. *Every* well-formed formula has an equal number of left and right brackets.
2. For *every* truth valuation t, if all the premises are true under t, then the conclusion is true under t.
3. If *there does not exist* a natural deduction proof from the premises to the conclusion, then the premises do not semantically entail the conclusion.

Would you really use predicate logic?

Examples of predicate logic in Computer Science:

1. Data Structure: *Every* key stored in the left subtree of a node N is smaller than the key stored at N.
2. Algorithms: in the worst case, *every* comparison sort requires at least $cn \log n$ comparisons to sort n values, *for some* constant $c > 0$.
3. Java example: *there is no* path via references from any variable in scope to any meomry location available for garbage collection...
4. Database query: select a person whose age is greater than or equal to the age of *every* other person in the table.

Elements of predicate logic

Predicate logic generalizes propositional logic.

New things in predicate logic:

- Domains
- Predicates
- Quantifiers

Domains

A *domain* is a non-empty set of objects. It is a world that our statement is situated within.

Examples of domains: natural numbers, people, animals, etc.

Why is it important to specify a domain? *The same statement can have different truth values in different domains.*

Consider this statement: There exists a number whose square is 2.

- If our domain is the set of natural numbers, is this statement true or false?
- If our domain is the set of real numbers, is this statement true or false?

Objects in a domain

Constants: concrete objects in the domain

- Natural numbers: 0, 6, 100, ...
- Alice, Bob, Eve, ...
- Animals: Winnie the Pooh, Micky Mouse, Simba, ...

Variables: placeholders for concrete objects, e.g. x, y, z.

A variable lets us refer to an object without specifying which particular object it is.

Predicates

A *predicate* represents

- a property of an individual, or
- a relationship among mulitple individuals.

Think of a predicate as a special type of function which takes constants and/or variables as inputs and *outputs* `T`/`F`. It can have any number of arguments.

Examples:

- Define $L(x)$ to mean "x is a lecturer". (unary predicate)
  - Alice is a lecturer: $L(Alice)$
  - Micky Mouse is not a lecturer: $(\neg L(MickyMouse))$
  - y is a lecturer: $L(y)$
- Define $Y(x, y)$ to mean "x is younger than y". (binary predicate/relation)
  - Alex is younger than Sam: $Y(Alex, Sam)$
  - a is younger than b: $Y(a, b)$

Quantifiers

For how many objects in the domain is the statement true?

- The universal quantifier $\forall$: the statement is true for every object in the domain.
- The existential quantifier $\exists$: the statement is true for one or more objects in the domain.

Translating English into Predicate Logic

Let the domain be the set of animals. $\mathit{Honey}(x)$ means that x likes honey. $\mathit{Bear}(x)$ means that x is a bear.

Consider the following translations of English sentences into predicate logic formulas. Are the translations correct?

1. At least one animal likes honey. $(\exists x\ \mathit{Honey}(x))$.
2. Not every animal likes honey. $(\neg(\forall x\ \mathit{Honey}(x)))$.

a. Both are correct.

b. 1 is correct and 2 is wrong.

c. 1 is wrong and 2 is correct.

d. Both are wrong.

Translating English into Predicate Logic

Translate the following sentences into predicate logic.

1. All animals like honey.
2. At least one animal likes honey.
3. Not every animal likes honey.
4. No animal likes honey.

Let the domain be the set of animals. $\mathit{Honey}(x)$ means that x likes honey. $\mathit{Bear}(x)$ means that x is a bear.

Translating English into Predicate Logic

Translate the following sentences into predicate logic.

5. No animal dislikes honey.
6. Not every animal dislikes honey.
7. Some animal dislikes honey.
8. Every animal dislikes honey.

Let the domain be the set of animals. $\mathit{Honey}(x)$ means that x likes honey. $\mathit{Bear}(x)$ means that x is a bear.

Translating English into Predicate Logic

Consider this sentence: *every* bear likes honey. Which one is the correct translation into predicate logic?

a. $(\forall x\,(Bear(x) \wedge Honey(x)))$

b. $(\forall x\,(Bear(x) \vee Honey(x)))$

c. $(\forall x\,(Bear(x) \rightarrow Honey(x)))$

d. $(\forall x\,(Honey(x) \rightarrow Bear(x)))$

Let the domain be the set of animals. *Honey*(x) means that x likes honey. *Bear*(x) means that x is a bear.

Translating English into Predicate Logic

Consider this sentence: *some* bear likes honey. Which one is the correct translation into predicate logic?

a. $(\exists x\,(Bear(x) \wedge Honey(x)))$

b. $(\exists x\,(Bear(x) \vee Honey(x)))$

c. $(\exists x\,(Bear(x) \rightarrow Honey(x)))$

d. $(\exists x\,(Honey(x) \rightarrow Bear(x)))$

Let the domain be the set of animals. *Honey*(x) means that x likes honey. *Bear*(x) means that x is a bear.

Multiple Quantifiers

Let the domain be the set of people. Let $L(x, y)$ mean that person x likes person y.

Translate the following formulas into English.

1. $(\forall x\,(\forall y\ L(x, y)))$
2. $(\exists x\,(\exists y\ L(x, y)))$
3. $(\forall x\,(\exists y\ L(x, y)))$
4. $(\exists y\,(\forall x\ L(x, y)))$