Predicate Logic

CS 330 : Discrete Structures

Predicate logic adds variables, predicates, and quantifiers to propositional logic.

e.g. propositional logic:

p: "John likes cake." q: "Jane likes cake"

e.g. predicate logic:

$P(x)$: "x likes cake"

$\exists x\, P(x)$: "there exists x such that x likes cake"

A predicate is a statement that is True or False, depending on the value of one or more variables

taken from the domain or "universe of discourse"

A predicate $P(x, y, \dots)$ can be thought of as a propositional function that evaluates to T/F, based on its inputs $x, y, \dots$

e.g. $P(x)$: $x < 10$ and x is prime

$P(3)$?

$P(9)$?

$P(11)$?

$P(20)$?

$P(5) \vee P(20)$?

$P(7) \wedge P(10)$?

$P(1) \rightarrow P(12)$?

e.g. $Q(x, y, z) : x + y = z$

$Q(1, 2, 3)$?

$Q(3, -1, 4)$?

$Q(2, 4, 7) \rightarrow Q(3, 4, 7)$?

A quantifier specifies how many values from the domain, when assigned to a particular variable, satisfy a predicate

Most important quantifiers:

- universal quantifier : $\forall$ — "for all"
- existential quantifier : $\exists$ — "there exists"

Eg. 1

the quantifiers bind the variable x

$\forall x\, P(x)$: "For all x in the domain, P(x) is True."

$\exists x\, P(x)$: "There exists some x in the domain such that P(x) is True."

note: important to specify the domain!
(we'll discuss different ways of doing this)

Quantifiers can be thought of as looping over values in the domain.

- $\forall x\, P(x)$ loops over all x in domain
  - only true if $P(x)$ is true for all iterations
  - if any x causes $P(x)$ to be false, terminate w/ false
- $\exists x\, P(x)$ loops over all x in domain
  - if we find $P(x)$ true for some x, terminate w/ true
  - if loop terminates w/o find $P(x)$ true for any x, evaluates to false

Quantifiers can also be considered in terms of logical conjunctions / disjunctions (for finite domains)

- $\forall x\, P(x)$ is the logical conjunction of $P(x)$ for all x

  i.e., $\forall x\, P(x) \approx \bigwedge_{x} P(x)$

- $\exists x\, P(x)$ is the logical disjunction of $P(x)$ for all x

  i.e., $\exists x\, P(x) \approx \bigvee_{x} P(x)$

On Precedence

$\forall$ and $\exists$ have higher precedence than all logical operators!

E.g. $\forall x P(x) \vee Q(x) \equiv (\forall x P(x)) \vee Q(x)$

$\forall x P(x) \vee Q(x) \not\equiv \forall x (P(x) \vee Q(x))$

We can come up w/ other quantifiers . . .

e.g., "there are exactly N values . . ."

"for the majority of values . . ." (assuming finite domain)

"there is a unique value . . ."

but we can express most other quantifiers using propositional operators.

Uniqueness Quantifier "$\exists!$"

$\exists! x\, P(x)$: "There is a unique x such that $P(x)$."

e.g. $P(x)$: $x + 10 = 0$, domain is $\mathbb{Z}$

$\exists! x\, P(x)$? T

e.g. $P(x)$: $x < 0$, domain is $\mathbb{Z}$

$\exists! x\, P(x)$? F

Express $\exists!$ in terms of $\exists$ and $\forall$:

$$\exists! x\, P(x) \equiv \exists x \left(P(x) \wedge \forall y \left(P(y) \rightarrow y = x \right) \right)$$

"nested" quantifier

Translating from English to logic

E.g. "Every student in CS 330 can program in Java"

① Assume domain of x is students in CS 330

$P(x)$: "x can program in Java"

$\forall x P(x)$

② Assume domain of x is all students

$Q(x)$: "x is a student in CS 330"

$\forall x (Q(x) \to P(x))$ $(\forall x (Q(x) \wedge P(x)))$

is wrong!

Translating from English to logic

E.g. "Some student in CS 330 can program in Java"

① Assume domain of x is students in CS 330

$P(x)$: "x can program in Java"

$\exists x\, P(x)$

② Assume domain of x is all students

$Q(x)$: "x is a student in CS 330"

$\exists x\, (P(x) \wedge Q(x))$ $\quad (\forall x\, (P(x) \rightarrow Q(x)))$

is wrong!

Negating Quantifiers.

E.g. $P(x)$: "x loves honey" ; domain of x is all bees

$\forall x P(x)$: "all bees love honey"

$\neg \forall x P(x)$: "it is not true that all bees love honey"

$\not\equiv$ "all bees do not love honey"

$\equiv$ "there is a bee that doesn't love honey"

i.e., $\neg \forall x P(x) \equiv \exists x (\neg P(x))$

De Morgan's Laws for Quantifiers

$$\neg \forall x P(x) \equiv \exists x \neg P(x)$$

$$\neg \exists x P(x) \equiv \forall x \neg P(x)$$

Example — domain = { fleegles, snurds, thingamabobs }

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "Everything is a fleegle"

$\forall x\, F(x)$

Example — domain = { fleegles, snurds, thingamabobs }

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "Nothing is a snurd"

$$\forall x \neg S(x)$$

$$\equiv \neg \exists x \, S(x)$$

Example — domain = $\{$ fleegles, snurds, thingamabobs $\}$

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "All fleegles are snurds"

$$\forall x\,(F(x) \rightarrow S(x))$$

Example — domain = {fleegles, snurds, thingamabobs}

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "Some fleegles are thingamabobs"

$$\exists x\,(F(x) \wedge T(x))$$

Example — domain = {fleegles, snurds, thingamabobs}

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "No snurd is a thingamabob"

$$\neg \exists x \left(S(x) \wedge T(x) \right) \qquad \forall x \left(S(x) \Rightarrow \neg T(x) \right)$$

$$\equiv$$

$$\forall x \left(\neg S(x) \vee \neg T(x) \right)$$

Example — domain = $\{$ fleegles, snurds, thingamabobs $\}$

$F(x)$: x is a fleegle

$S(x)$: x is a snurd

$T(x)$: x is a thingamabob

Translate "If any fleegle is a snurd then it is also a thingamabob"

$$\forall x \left(\left(F(x) \land S(x) \right) \to T(x) \right)$$

Validity & Satisfiability

- an assertion involving predicates + quantifiers is valid if it is true for all domains, and all possible predicates

  e.g., $\forall x \neg P(x) \leftrightarrow \neg \exists x P(x)$

- an assertion is satisfiable if it is only true for some domains and predicates

  e.g., $\forall x (P(x) \leftrightarrow Q(x))$

- otherwise, it is unsatisfiable

  e.g., $\forall x (P(x) \wedge \neg P(x))$

Nested Quantifiers

- quantifiers that appear w/in the scope of other quantifiers

e.g. $\forall x \, \exists y \, P(x,y)$

"for every x, there exists a y for which $P(x,y)$ is true"

Nested Quantifiers

- can be thought of as nested loops — for each value of the variable bound by the outside quantifier, step through all values of inner quantifiers

- order matters!

e.g. $P(x,y): x+y=0$; x, y from $\mathbb{R}$

$$\forall x \exists y\, P(x,y) = T$$

$$\exists y \forall x\, P(x,y) = F$$

Given: $L(x,y)$: "x loves y"

Express the following using quantifiers:

- "Everybody loves somebody":

  $\forall x \exists y\, L(x,y)$

- "There is someone whom everybody loves.":

  $\exists x \forall y\, L(y,x)$

- "Nobody loves everybody"

  $\neg \exists x \forall y\, L(x,y)$ or $\forall x \exists y \neg L(x,y)$

Negating nested quantifiers:

- rewrite the following so that negations only appear directly in front of predicates

$$\neg \forall x \forall y\, P(x,y)$$

$$\exists x \exists y\, \neg P(x,y)$$

$$\neg(\exists x \exists y\, \neg P(x,y) \wedge \forall x \forall y\,(Q(x,y) \vee \neg R(x,y)))$$

$$\forall x \forall y\, P(x,y) \vee \exists x \exists y\,(\neg Q(x,y) \wedge R(x,y))$$

Given : $T(s,c)$: student s has taken class c

domain of s = all IIT students

domain of c = all CS classes

Translate to English :

$$\exists x\,(T(\text{Michael}, x) \wedge T(\text{Shannon}, x))$$

"There is a class that both Michael and Shannon have taken."

Given: $T(s,c)$: student s has taken class c

domain of s = all IIT students

domain of c = all CS classes

Translate to English:

$$\exists x \forall y \left(x \neq \text{Michael} \wedge \left(T(\text{michael}, y) \to T(x,y)\right)\right)$$

"There is a student who has taken all the classes Michael has taken."

Given: $T(s,c)$: student s has taken class c

domain of s = all IIT students

domain of c = all CS classes

Translate to English:

$$\exists x \exists y \forall z \left((x \neq y) \wedge \left(T(x,z) \leftrightarrow T(y,z) \right) \right)$$

"There are two separate students that have taken precisely the same classes."