Logical and Theoretical Foundations of Computer Science

LaTFoCS

Pamela Fleischmann

fpa@informatik.uni-kiel.de

Winter Semester 2019

Kiel University
Dependable Systems Group

Predicate Logic -
The need for a richer language

Back to the syllogisms

- Some bags are pockets.
  No pocket is a pouch.
  Conclusion: all bags are not pouches.
- Some pigs are predators.
  No predator is a pet.
  Conclusion: some pigs are not pets
- Some maggots are flies.
  No fly is welcome.
  Conclusion: no maggots are welcome.
- Some doctors are fools.
  All fools are rich.
  Conclusion: some doctors are rich.

Closer look into the syllogisms

Some bags are pockets.

No pocket is a pouch.

Conclusion: all bags are not pouches.

Closer look into the syllogisms

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- Can we express these sentences with propositional logic to determine the truth value of the conclusion mathematically?

Closer look into the syllogisms

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- Can we express these sentences with propositional logic to determine the truth value of the conclusion mathematically?
- How to express *Some*, *being something*, *All*?

Informal Introduction of Predicates

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- it is about objects being bags, pockets, or pouches

Informal Introduction of Predicates

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- it is about objects being bags, pockets, or pouches
- let's take an arbitrary object x

Informal Introduction of Predicates

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- it is about objects being bags, pockets, or pouches
- let's take an arbitrary object x
- we define predicates Bag, Pocket, Pouch with

Informal Introduction of Predicates

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- it is about objects being bags, pockets, or pouches
- let's take an arbitrary object x
- we define predicates Bag, Pocket, Pouch with
  - Bag(x) is `true` iff x is a bag
  - Pocket(x) is `true` iff x is a pocket
  - Pouch(x) is `true` iff x is a pouch

Informal Introduction of Quantifiers

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- how to describe *some*, *no*, *all*?

Informal Introduction of Quantifiers

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- how to describe *some*, *no*, *all*?
- *no* and *all* are opposed: no pocket is a pouch $\rightsquigarrow$ all pockets are not pouches

Informal Introduction of Quantifiers

Some bags are pockets.
No pocket is a pouch.
Conclusion: all bags are not pouches.

- how to describe *some*, *no*, *all*?
- *no* and *all* are opposed: no pocket is a pouch $\rightsquigarrow$ all pockets are not pouches
- we need to model *some* and *all*
  - $\exists x$: for some x, there exist x (notice the plural! - at least one)
  - $\forall x$: for all x, every x

Back to the syllogism

Some bags are pockets.	$\exists x(\text{Bag}(x) \rightarrow \text{Pocket}(x))$
No pocket is a pouch.	$\forall x(\text{Pocket}(x) \rightarrow \neg\,\text{Pouch}(x))$
Conclusion:	
All bags are not pouches.	$\forall x(\text{Bag}(x) \rightarrow \neg\,\text{Pouch}(x))$

Back to the syllogism

Some bags are pockets.	$\exists x(\mathrm{Bag}(x) \rightarrow \mathrm{Pocket}(x))$
No pocket is a pouch.	$\forall x(\mathrm{Pocket}(x) \rightarrow \neg\,\mathrm{Pouch}(x))$
Conclusion:	
All bags are not pouches.	$\forall x(\mathrm{Bag}(x) \rightarrow \neg\,\mathrm{Pouch}(x))$

And now? How can we decide wether the conclusion is `true`?

Roadmap

As in propositional logic we define

1. the syntax

Roadmap

As in propositional logic we define

1. the syntax
2. the semantics

Roadmap

As in propositional logic we define

1. the syntax
2. the semantics

And then we have a look what we can deduce.

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?
- in Germany each car x has exactly one owner y: owner(x)

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?
- in Germany each car x has exactly one owner y: owner(x)
  - owner is a function

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?
- in Germany each car x has exactly one owner y: owner(x)
  - owner is a function
  - input: car x

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?
- in Germany each car x has exactly one owner y: owner(x)
  - owner is a function
  - input: car x
  - output: x's owner y

Functions for more Elegance (Informal)

- predicates can have a higher arity Brother(x, y), Mother(x, y), Owner(x, y)
- each x may have several brothers $y \rightsquigarrow$ relation is good
- can each x have several mothers y? or is the mother y of x uniquely determined?
- in Germany each car x has exactly one owner y: owner(x)
  - owner is a function
  - input: car x
  - output: x's owner y
- functions without arguments are called constants