lecture 21, the formal language GPLI

phil1012 introductory logic

overview

this lecture

- an introduction to the formal language GPLI
- some of the limitations of GPL which motivate the shift to GPLI
- the identity relation
- translations into GPLI
- the semantics of identity

learning outcomes

- after doing the relevant reading for this lecture, listening to the lecture, and attending the relevant tutorial, you will be able to:
  - explain the limitations of GPL and the motivations for GPLI
  - explain what the identity relation is
  - translate propositions from English into GPLI
  - explain the motivation behind the semantics of identity
  - determine whether a GPLI proposition is true or false on a model

required reading

- sections 13.1, 13.2, and 13.3 of chapter 13

expressive limitations of GPL

expressive limitations of GPL

- consider the following argument

P1.	John loves something
P2.	John does not love himself
C1.	John loves something other than himself

- we have the following glossary:
  - jj: John
  - LxyLxy: xx loves yy
- and we translate the premises as follows:
  - ∃xLjx\exists x Ljx
  - ¬Ljj\lnot Ljj
- but we have no way of translating the conclusion
- we have no way of saying that John loves someone other than himself

introducing GPLI

introducing GPLI

- we introduce a new two-place predicate into our logical language: the **identity** predicate
  - I2I^2

- in GPL I2I^2 is not used for a two-place predicate. Although I1I^1, I3I^3, I4I^4, and so on are
- we are extending GPL to GPLI which includes the symbol I2I^2 in addition to all the symbols of GPL
- syntactically, I2I^2 functions just like a two-place predicate in GPL
- semantically, I2I^2 is treated differently
  - its meaning does not vary from model to model
  - it is not given a glossary
  - in every model it expresses the **identity relation**

- since the two-place predicate I2I^2 is so special in GPLI we usually abbreviate it as follows:
  - ==
- and we abbreviate negations involving I2I^2 using:
  - ≠\neq
- it is important to remember, however, that == is just an abbreviation of a two-place preciate

the identity relation

the identity relation

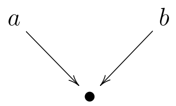
- == expresses the identity relation
- but what is the relation of identity?

- B1 and B2 are identical
- B1 is identical to B2
- B1 and B2 are *exactly the same in all respects*

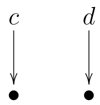
- B1 and B2 are identical
- B1 is identical to B2
- B1 and B2 are *one and the same thing*

- we have two relations:
  - **identity** (being one and the same thing)
  - **exact similarity** (being the same in all respects)
- how are they related?
  - Leibniz's Law/Indiscernability of Identicals: if xx is identical to yy then xx is exactly similar to yy. (a logical truth)
  - Identity of Indiscernibles: if xx is exactly similar to yy then xx is identical to yy. (controversial)

- aa and bb are identical (the names 'aa' and 'bb' pick out the same thing):



* cc and dd are non-identical (the names ‘cc’ and ‘dd’ pick out different things):



the predicate ==

the predicate ==

* consider the following sentences:
* B1 is a banana
* B1 is B2
* the word ‘is’ has a different meaning in each. in the first case, it is the ‘is’ of predication. in the second, it is the ‘is’ of identity.

---

* we have the following glossary:
    * aa: B1
    * bb: B2
    * ByBy: xx is a banana

---

* we have the following translations:
    * BaBa
    * a=ba=b

---

* recall that I2I^2 (or ==) is part of the logical vocabulary
* we do not put an entry for it in the glossary

translations into GPLI

translations into GPLI

* GPLI allows us to express more than just that a and b are identical
* let’s take a look at the expressive power we have gained

---

* return to our example:

P1.	John loves something
P2.	John does not love himself
C1.	John loves something other than himself

---

* here’s a glossary:
    * jj: John
    * LxyLxy: xx loves yy

- how should we translate the conclusion of the argument?
    - ∃x(Ljx∧¬I2jx)\exists x (Ljx \land \lnot I^2jx)
- there is something that John loves and that thing is not John
- John loves something other than himself!

- let’s look at some other examples . . .

- here’s a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John isn’t Mary

  ¬I2jm\lnot I^2 jm
  j≠mj \neq m

- here’s a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John loves someone other than himself

  ∃x(Px∧Ljx∧¬I2xj)\exists x (Px \land Ljx \land \lnot I^2xj)
  ∃x(Px∧Ljx∧x≠j)\exists x (Px \land Ljx \land x \neq j)

- here’s a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John loves everyone except Sam

  ∀x((Px∧¬I2xs)→Ljx)\forall x ((Px \land \lnot I^2xs) \rightarrow Ljx)
  ∀x((Px∧x≠s)→Ljx)\forall x ((Px \land x \neq s) \rightarrow Ljx)

- here’s a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  Some baker other than John loves Mary

∃x(Bx∧¬I2xj∧Lxm)\exists x (Bx \land \lnot I^2xj \land Lxm)
∃x(Bx∧x≠j∧Lxm)\exists x (Bx \land x \neq j \land Lxm)

- here's a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  Everyone loves themselves

  ∀x(Px→Lxx)\forall x(Px \rightarrow Lxx)

- here's a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John loves everyone other than himself

  ∀x((Px∧¬I2xj)→Ljx)\forall x((Px \land \lnot I^2xj) \rightarrow Ljx)
  ∀x((Px∧x≠j)→Ljx)\forall x((Px \land x \neq j) \rightarrow Ljx)

- here's a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John loves everyone other than himself

  ∀x((Px∧¬I2xj)→Ljx)∧¬Ljj\forall x((Px \land \lnot I^2xj) \rightarrow Ljx) \land \lnot Ljj
  ∀x((Px∧x≠j)→Ljx)∧¬Ljj\forall x((Px \land x \neq j) \rightarrow Ljx) \land \lnot Ljj

---

- here's a glossary:

  jj: John, mm: Mary, ss: Sam
  LxyLxy: xx loves yy, SxSx: xx is a baker, PxPx: xx is a person

- translate the following into GPLI:

  John loves everything but does not love himself

  ∀xLjx∧¬Ljj\forall x Ljx \land \lnot Ljj

- (this is a contradiction. it is false in every model.)

the semantics of ==

the semantics of ==

- since I2I^2 is a two-place predicate, its extension is a set of ordered pairs of members of the domain
- once we have an extension for I2I^2, the semantics for GPLI is just like the semantics of GPL
- so all we need is an extension for I2I^2

- in a model in which the domain is {1, 2, 3}, the extension of I2I^2 will be the following set of ordered pairs:
    - {⟨1,1⟩,⟨2,2⟩,⟨3,3⟩}\{ \langle 1, 1 \rangle, \langle 2, 2 \rangle, \langle 3, 3 \rangle \}

- in a model in which the domain is {Bill,Ben}\{Bill, Ben\}, the extension of I2I^2 will be the following set of ordered pairs:
    - {⟨Bill,Bill⟩,⟨Ben,Ben⟩}\{ \langle Bill, Bill \rangle, \langle Ben, Ben \rangle\}

- whatever the domain is, the extension of I2I^2 will contain exactly one ordered pair for each object in the domain: the pair containing that object in both the first and second place
- the identity relation is a relation which holds between each object and itself and between no other objects

---

- note! The extension of the identity predicate is not exactly the same on all models
- given a domain, however, the extension of the identity predicate is fixed
- this is not true of other predicates
- this is why it is part of the logical vocabulary

propositions of the form I2abI^2ab

- I2abI^2ab is true in a model if and only if the pair consisting of the referent of aa and the referent of bb is in the extension of I2I^2
    - Domain: {Bill,Ben}\{Bill, Ben\}
    - Referents: aa: BillBill, bb: BillBill
    - Extension of I2I^2
        - {⟨Bill,Bill⟩,⟨Ben,Ben⟩}\{ \langle Bill, Bill \rangle, \langle Ben, Ben \rangle\}
- I2abI^2ab is true in a model if and only if aa and bb have the same referent on that model

---

- let's run through a couple of examples

- here's a model:
    - Domain: {Sydney,Canberra,Melbourne}\{Sydney, Canberra, Melbourne\}
    - Referents: aa: MelbourneMelbourne, bb: CanberraCanberra, cc: SydneySydney
    - Extensions: NN: {⟨Sydney,Canberra⟩,\{ \langle Sydney,

Canberra \rangle, ⟨Canberra,Sydney⟩,⟨Melbourne,\langle Canberra, Sydney \rangle, \langle Melbourne, Canberra⟩,⟨Canberra,Melbourne⟩}Canberra \rangle, \langle Canberra, Melbourne \rangle \}

- is the following true or false in the model?
  - ∀x∀y(Nxy→(I2xb∨I2yb))\forall x \forall y (Nxy \rightarrow (I^2xb \lor I^2yb))

- here’s a model:
  - Domain: {Sydney,Canberra,Melbourne}\{Sydney, Canberra, Melbourne\}
  - Referents: aa: MelbourneMelbourne, bb: CanberraCanberra, cc: SydneySydney
  - Extensions: NN: {⟨Sydney,Canberra⟩,\{ \langle Sydney, Canberra \rangle, ⟨Canberra,Sydney⟩,⟨Melbourne,\langle Canberra, Sydney \rangle, \langle Melbourne, Canberra⟩,⟨Canberra,Melbourne⟩}Canberra \rangle, \langle Canberra, Melbourne \rangle \}
- is the following true or false in the model?
  - (I2ab→∀x∀yNxy)(I^2ab \rightarrow \forall x \forall y Nxy)

wrapping up

this lecture

- some of the limitations of GPL which motivate the shift to GPLI
- introducing GPLI
- the identity relation
- translations into GPLI
- the semantics of identity

next lecture

- lecture 22, trees for GPLI