Semantic Theory

Lexical Semantics III

Manfred Pinkal/ Stefan Thater
Saarland University
Summer 2012

Mass Nouns and Plurals

- *water, gold, wood, money, soup, ...*

Mass nouns behave like plurals in different respects:

- Mass nouns and plurals are closed under summation:
  students plus *students* is *students*
  water plus *water* is *water*
- Mass nouns and plurals combine with cardinalities:
  5 students – 5 liters of water
- Mass nouns and plurals share grammatical patterns:
  e.g., indefinite plural NPs and indefinite mass term NPs don't take an article in English and German

Mass Nouns vs. Plurals

- Mass nouns are divisive, unlike plurals: An amount of *water* can always be subdivided into proper parts, which are *water* again.
- Mass nouns are a challenge for model theoretic semantics: Their denotations cannot be reduced to atomic individuals.

Model structure for mass nouns

- We add another sort of entities, the "portions of matter" M, to the model structure, and distinguish an individual part and a material part relation, writing $\leq_i$ for the former, and $\leq_m$ for the latter:
  $M = \langle\langle U, \leq_i\rangle, \langle M, \leq_m\rangle, V\rangle$
  - $U \cap M = \varnothing$
  - $\langle U, \leq_i\rangle$ is an atomic join semi-lattice
  - $\langle M, \leq_m\rangle$ is a non-atomic and dense join semi-lattice
  - V is a value assignment function
- In the logical representation language, we add a material fusion operation and a material part relation, and distinguish $\oplus_i$, $\oplus_m$, $\triangleleft_i$, and $\triangleleft_m$.
- We use ***x, y, z***, ... as variables referring to matters.

Model structure for mass nouns

- There is close relationship between the domain of (atomic and sum) individuals and material entities: Each individual consists of a specific portion of matter.
- To model the object-matter relation, we introduce a "materialization" function *h* into the model structure: a homomorphism that maps (atomic and pluralic) individuals to the matter they consist of.
- $M = \langle\langle U, \leq_i\rangle, \langle M, \leq_m\rangle, h, V\rangle$
- Because h is a homomorphism, the following hold:

  $a \leq_i b$ iff $h(a) \leq_m h(b)$

  $h(a \sqcup_i b) = h(a) \sqcup_m h(b)$
- We express the materialization function with the new logical operator *m* (type <e,e>): $[\![m(\alpha)]\!]^{M,g} = h([\![\alpha]\!]^{M,g})$, where α:e is an expression denoting an individual entity.

Examples

The/A ring is made of gold

→ $\exists y(ring(y) \wedge gold(m(y)))$

The/A ring contains gold

→ $\exists y \exists \boldsymbol{x} (ring(y) \wedge \boldsymbol{x} \triangleleft_m m(y) \wedge gold(\boldsymbol{x}))$

Back to Event Semantics

- A model structure with events and temporal precedence is defined as

  $M = (U, E, <, e_u, V)$,

  with $U \cap E = \emptyset$,

  $< \subseteq E \times E$ an asymmetric relation (temporal precedence)

  $e_u \in E$ the utterance event

  V an interpretation function like in standard FOL, with

  $D_e = U \cup E$

Model Structure with Sub-Events

- In analogy to plural semantics, we can represent sub-event relations via a join semi-lattice.

  $M = (U, \langle E, \leq_e\rangle, <, e_u, V)$,

  with $U \cap E = \emptyset$,

  $< \subseteq E \times E$ an asymmetric relation (temporal precedence)

  $e_u \in E$ the utterance event

  $\langle E, \leq_e\rangle$ a join semi-lattice

  V an interpretation function

- The model structure must observe some additional constraints on < and $\leq_e$, e.g.:

  If $e_1 < e_2$, $e_1' \leq_e e_1$, $e_2' \leq_e e_2$, then $e_1' < e_2'$.

  If $e_1' \circ e_2'$, $e_1' \leq_e e_1$, $e_2' \leq_e e_2$, then $e_1 \circ e_2$.

Model Structure with Sub-Events

Application:

- Modeling complex events as sequences of temporally ordered sub-events (e.g. "scripts" like: *visit a restaurant, shopping in the supermarket*)

Processes vs. proper events

- *John walked from 8 a.m. to 11 a.m.* ⊨ *John walked from 9 to 10 a.m.*
- *John walked from 8 to 9 and from 9 to 10* ⊨ *John walked from 8 to 10 a.m.*
- *John painted a picture from 8 a.m. to 11 a.m.* ⊭ *John painted a picture from 9 to 10 a.m.*

Processes and mass terms

- Processes are cumulative and divisive:
- $\text{rain}(e_1), \text{rain}(e_2) \vDash \text{rain}(e_1 \oplus_e e_2)$
- $e_1 \lhd_e e_2, \text{rain}(e_2) \vDash \text{rain}(e_1)$
- Assume individual events and "event matter", in analogy to the semantics of common nouns, and represent them through different join semi-lattices:
  $M = (\langle U, \leq_i \rangle, \langle M, \leq_m \rangle, h, \langle E_i, \leq_{ei} \rangle, \langle E_m, \leq_{em} \rangle, <, e_u, V)$
- ... plus a materialisation function that maps individual events to processes:
  $M = (\langle U, \leq_i \rangle, \langle M, \leq_m \rangle, h, \langle E_i, \leq_{ei} \rangle, \langle E_m, \leq_{em} \rangle, h_e, <, e_u, V)$
- Add relations $\lhd_{ei}$, $\lhd_{em}$, and operators $\oplus_{ei}$, $\oplus_{em}$, m_e to the representation language, and give them the straightforward semantic interpretation in terms of $\leq_{ei}$, $\leq_{em}$, $\sqcup_{ei}$, $\sqcup_{em}$, h_e.

The Progressive

The progressive tense has the materialization function h_e as its semantics, which maps individual events (the telic action of John's eating an apple) to the process or activity carried out to bring the result about.

- *John is eating an apple*

- Progressive operator: PROG := $\lambda E \lambda \boldsymbol{e}\, \exists e(E(e) \wedge \boldsymbol{e} = m_e(e))$
- $\lambda E \lambda \boldsymbol{e}\, \exists e(E(e) \wedge \boldsymbol{e} = m_e(e))((\lambda e \exists x[\text{apple}(x) \wedge \text{eat}(e, j^*, x)])))$
  $\Leftrightarrow_\beta \lambda \boldsymbol{e}\, \exists e(\exists x[\text{apple}(x) \wedge \text{eat}(e, j^*, x)] \wedge \boldsymbol{e} = m_e(e))$

- PRES : $\lambda E \exists e(E(e) \wedge e\, o\, e_u)$
- $\lambda E \exists e(E(e) \wedge e\, o\, e_u)\, (\lambda \boldsymbol{e}\, \exists e(\exists x[\text{apple}(x) \wedge \text{eat}(e, j^*, x)] \wedge \boldsymbol{e} = m_e(e)))$
  $\Leftrightarrow_\beta \exists \boldsymbol{e}(\exists e \exists x[\text{apple}(x) \wedge \text{eat}(e, j^*, x)] \wedge \boldsymbol{e} = m_e(e) \wedge \boldsymbol{e}\, o\, e_u)$