An Introduction to Semantics using Type Theory with Records
Lecture 3

Jonathan Ginzburg
Université Paris-Diderot, Sorbonne Paris-Cité
Robin Cooper
University of Gothenburg

Today's Lecture

- Basic Illocutionary Interaction in KoS
- Adding abstract entities to the ontology
- Negation

Outline

Basic Illocutionary Interaction in KoS

Adding abstract entities to the semantic ontology

Asking, Asserting, Answering, and Accepting

Basic Illocutionary Interaction in KoS

Based on Chapter 4 from Jonathan Ginzburg. 2011. *The Interactive Stance: Meaning for conversation*. Oxford University Press.

Dialogue Analysis

- Dialogue analyst's task: describe conventionally acceptable patterns of interaction (*protocols*), in terms of sequences of information states.
- Methodological constraint: compositionality (but as with the sentential level not obsessively [cf. the need for constructionism].

Contexts in KoS

- in KoS, there is actually no single context, for reasons that will soon emerge.
- Instead of a single context, analysis is formulated at a level of information states, one per conversational participant.
- What the typed entities depicted in (1),(2) amount to is explained in the next few lectures.
- The type of information states is given in (1):

(1) $$\text{TotalInformationState (TIS)} = \begin{bmatrix} \text{dialoguegameboard} : \text{DGBType} \\ \text{private} : \text{Private} \end{bmatrix}$$

Contexts in KoS

- The dialogue gameboard (DGB) represents information that arises from publicized interactions and, for now, we can identify it as the public context.

(2) DGBType (initial definition) =

$$\begin{bmatrix} \text{spkr: Ind} \\ \text{addr: Ind} \\ \text{c-utt : addressing(spkr,addr)} \\ \text{Facts : Set(Prop)} \\ \text{Moves : list(IllocProp)} \\ \text{QUD : poset(Question)} \end{bmatrix}$$

Basics of Interaction

- The basic units of change are mappings between DGBs that specify how one DGB configuration can be modified into another. $\therefore$ *conversational rule*.
- The types specifying its domain and its range respectively the *preconditions* and the *effects*.

Basics of Interaction

- Notationwise a conversational rule will be specified as in (3a). We will often notate such a mapping as in (3b):

(3) a. r : $\begin{bmatrix} \ldots \\ \text{dgb1} : \text{DGBType} \\ \ldots \end{bmatrix} \mapsto \begin{bmatrix} \ldots \\ \text{dgb2} : \text{DGBType} \\ \ldots \end{bmatrix}$

b. $\begin{bmatrix} \text{pre(conds)} : \text{RType} \\ \text{effects} : \text{RType} \end{bmatrix}$

Greeting and Parting

- The conversational rule associated with greeting:

$$\begin{bmatrix} \text{pre} : & \begin{bmatrix} \text{spkr: Ind} \\ \text{addr: Ind} \\ \text{moves} = \text{elist} : \text{list(IllocProp)} \\ \text{qud} = \text{elist} : \text{list(Question)} \\ \text{facts} = \text{commonground1} : \text{Prop} \end{bmatrix} \\ \text{effects} : & \begin{bmatrix} \text{spkr} = \text{pre.spkr} : \text{Ind} \\ \text{addr} = \text{pre.addr} : \text{Ind} \\ \text{LatestMove} = \text{Greet(spkr,addr):IllocProp} \\ \text{qud} = \text{pre.qud} : \text{list(Question)} \\ \text{facts} = \text{pre.facts} : \text{Prop} \end{bmatrix} \end{bmatrix}$$

Greeting and Parting

- So we can *abbreviate* conversational rules as in (4a), which for the rule for greeting yields (4b).

(4) a. $\begin{bmatrix} \text{pre: } \text{PreCondSpec} \\ \text{effects} : \text{ChangePrecondSpec} \end{bmatrix}$

b. $\begin{bmatrix} \text{pre} : \begin{bmatrix} \text{moves} = \text{elist} : \text{list(IllocProp)} \\ \text{qud} = \text{elist} : \text{list(Question)} \end{bmatrix} \\ \text{effects} : \begin{bmatrix} \text{LatestMove} = \text{Greet(spkr,addr):IllocProp} \end{bmatrix} \end{bmatrix}$

Greeting and Parting

$$\left[\begin{array}{lll}\text{preconds} & : & \left[\begin{array}{l}\text{qud} = \text{eset} : \text{poset(Question)}\\ \text{facts} : \text{Set(Prop)}\\ \text{f} = \text{MinInteraction}(\left\{\text{spkr,addr}\right\}) : \text{Prop}\\ \text{c1} : \text{member(f,facts)}\end{array}\right]\\ \text{effects} & : & \left[\text{LatestMove} = \text{Part(spkr,addr)}: \text{IllocProp}\right]\end{array}\right]$$

Greeting and Parting

- Counterparting:
- $$\begin{bmatrix} \text{preconds} & : \begin{bmatrix} \text{LatestMove} = \text{Part(spkr,addr)} : \text{IllocProp} \\ \text{qud} = \text{eset} : \text{poset(Question)} \end{bmatrix} \\ \text{effects} & : \begin{bmatrix} \text{spkr} = \text{preconds.addr}: \text{Ind} \\ \text{addr} = \text{preconds.spkr}: \text{Ind} \\ \text{LatestMove} = \text{CounterPart(spkr,addr)} : \text{IllocProp} \end{bmatrix} \end{bmatrix}$$

Greeting and Parting

$$\left[\begin{array}{ll} \text{preconds} & : \left[\begin{array}{l} \text{LatestMove} = \text{CounterPart(spkr,addr)}:\text{IllocProp} \\ \text{qud} = \text{eset} : \text{poset(Question)} \end{array}\right] \\ \text{effects} & : \left[\text{LatestMove} = \text{Disengaged}(\left\{\text{spkr,addr}\right\}) : \text{IllocProp}\right] \end{array}\right]$$

Outline

Adding abstract entities to the semantic ontology

Based on Chapter 3 from Jonathan Ginzburg. 2011. *The Interactive Stance: Meaning for conversation*. Oxford University Press.

Propositions

- Propositional-like entities, more intensional than events/situations, are a necessary ingredient for accounts of illocutionary acts, as well as of attitude reports.
- *Sets* of situations, although somewhat more fine grained than sets of worlds, also succumb to sophisticated variants of logical omniscience (see e.g. Soames' puzzle (Soames (1985)).).
- Building on a conception articulated 30 years earlier by Austin (Austin (1961)), Barwise and Etchemendy (1987) developed a theory of propositions in which a proposition is a structured object $prop(s, \sigma)$, individuated in terms of a situation s and a situation type σ.

(5) a. prop(s, σ) is true iff $s : \sigma$ (*s is of type* σ).
    b. prop(s, σ) is false iff $s \not: \sigma$ (*s is not of type* σ).

Propositions

- There are two ways to maintain the insights of an Austinian approach in TTR, implicitly Austinian or explicitly so.
- Cooper (2005) develops the former in which a proposition p is taken to be a record type.
- A witness for this type is a situation.
- On this strategy, a witness is not directly included in the semantic representation.

Propositions

- record types *are* competitive in such a role: they are sufficiently fine-grained to distinguish identity statements that involve distinct constituents.
- In this set up substitutivity of co-referentials (6e) and cross-linguistic equivalents ((6e), the Hebrew equivalent of (6a)) can be enforced:

(6) a. Enescu is identical with himself.
b. Poulenc is identical with himself.
c. $\left[\text{c} : \text{Identical(enescu, enescu)}\right]$
d. $\left[\text{c} : \text{Identical(poulenc, poulenc)}\right]$
e. He is identical with himself.
f. Enesku zehe leacmo.

Propositions

- A situational witness for the record type could also be deduced to explicate cases of event anaphora, as in(7); indeed, a similar strategy will be invoked when we discuss nominal anaphora.

(7) a. A: Jo and Mo got married yesterday. It was a wonderful occasion.
b. A: Jo's arriving next week. B: No, that's happening in about a month.

Propositions I

- Here we develop an explicitly Austinian approach, where the situational witness is directly included in the semantic representation.
- The original Austinian conception was that s is a situation deictically indicated by a speaker making an assertion—teasing the semantic difference between implicit and explicit witnesses is a difficult semantic task.
- Some other motivation from negation later on today.

Propositions II

- propositions can also play a role in characterizing the communicative process: in subsequent lectures we will show that (*locutionary propositions*) individuated in terms of an utterance event u_0 as well as to its grammatical type T_{u_0} allows one to simultaneously define update and clarification potential for utterances.
- In this case, there are potentially many instances of distinct locutionary propositions, which need to be differentiated on the basis of the utterance token—minimally any two utterances classified as being of the same type by the grammar.

Propositions

- TTR offers a straightforward way for us to model propositions using records. A proposition is a record of the form in (8a). The type of propositions is the record type (8b):

(8) a. $\begin{bmatrix} \text{sit} = r_0 \\ \text{sit-type} = p_0 \end{bmatrix}$

b. $\text{Prop} = \begin{bmatrix} \text{sit} : \text{Record} \\ \text{sit-type} : \text{RecType} \end{bmatrix}$

Propositions

- TTR offers a straightforward way for us to model propositions using records. A proposition is a record of the form in (10a). The type of propositions is the record type (10b):

(10) a. $\begin{bmatrix} \text{sit} = r_0 \\ \text{sit-type} = p_0 \end{bmatrix}$

b. $\text{Prop} = \begin{bmatrix} \text{sit} : \text{Record} \\ \text{sit-type} : \text{RecType} \end{bmatrix}$

- Truth:

(11) A proposition $\begin{bmatrix} \text{sit} = r_0 \\ \text{sit-type} = p_0 \end{bmatrix}$ is true iff $r_0 : p_0$

Con/Disjunctive Propositions

- Given the existence of meet/join operations on types, we can define conjunction and disjunction operations on propositions, see (Ginzburg, 2012) for details

Questions in TTR

- Given the existence of Austinian-like propositions and a theory of λ-abstraction given to us by existence of functional types, it is relatively straightforward to develop a theory of questions as propositional abstracts in TTR, building on earlier work in situation theory in Ginzburg (1995); Ginzburg and Sag (2000).

Questions: some simple examples

- A question will be a function from records into propositions:

(12) a. who ran

b. TTR representation—$\lambda r : \begin{bmatrix} \text{x} : \text{Ind} \\ \text{c1} : \text{person(x)} \end{bmatrix}$)

$$\begin{bmatrix} \text{sit} = r_1 \\ \text{sit-type} = \begin{bmatrix} \text{c} : \text{run(r.x)} \end{bmatrix} \end{bmatrix}$$

That is, a function that maps records r : T_{who} = $\begin{bmatrix} \text{x} : \text{Ind} \\ \text{c1} : \text{person(x)} \end{bmatrix}$ into propositions of the form

$$\begin{bmatrix} \text{sit} = r_1 \\ \text{sit-type} = \begin{bmatrix} \text{c} : \text{run(r.x)} \end{bmatrix} \end{bmatrix}$$

Questions: some simple examples

(13) a. who saw what

b. TTR representation—$(\mathrm{r} : \begin{bmatrix} \mathrm{x} : \mathrm{Ind} \\ \mathrm{c1} : \mathrm{person(x)} \\ \mathrm{y} : \mathrm{Ind} \\ \mathrm{c2} : \mathrm{thing(y)} \end{bmatrix})$

$$\begin{bmatrix} \mathrm{sit} = r_1 \\ \mathrm{sit\text{-}type} = \begin{bmatrix} \mathrm{c} : \mathrm{saw(r.x,r.y)} \end{bmatrix} \end{bmatrix}$$

Questions: some simple examples

- And, by extension, a question will be a 0-ary propositional abstract.

(14) a. Did Bo run

b. TTR representation—$\left(r : \begin{bmatrix} \end{bmatrix} \right) \begin{bmatrix} \text{sit} = r_1 \\ \text{sit-type} = \begin{bmatrix} \text{c} : \text{run(b)} \end{bmatrix} \end{bmatrix}$.

That is, a function that maps records $r : T_0 = \begin{bmatrix} \end{bmatrix}$

into propositions of the form $\begin{bmatrix} \text{sit} = r_1 \\ \text{sit-type} = \begin{bmatrix} \text{c} : \text{run(b)} \end{bmatrix} \end{bmatrix}$

Answerhood

- There are a number of notions of answerhood that are of importance to dialogue.
- We concern ourself here with one that relates to coherence: any speaker of a given language can recognize, independently of domain knowledge and of the goals underlying an interaction, that certain propositions are *about* or *directly concern* a given question.

Simple Answerhood in TTR

(15) Given a question $q : (A)B$:

a. $\text{AtomAns(q)} =_{def} \begin{bmatrix} \text{shortans} : \text{A} \\ \text{propans} = \text{q(shortans)} : \text{Prop} \end{bmatrix}$

b. $\text{NegAtomAns(q)} =_{def} \begin{bmatrix} \text{shortans} : \text{A} \\ \text{propans} = \neg\ \text{q(shortans)} : \text{Prop} \end{bmatrix}$

c. $\text{SimpleAns(q)} =_{def} \text{AtomAns(q)} \vee \text{NegAtomAns(q)}$

Simple Answerhood

- Simple answerhood covers a fair amount of ground. But it clearly underdetermines aboutness.
- On the polar front, it leaves out the whole gamut of answers to polar questions that are weaker than p or $\neg p$ such as conditional answers 'If r, then p' (e.g. 16a) or weakly modalized answers 'probably/possibly/maybe/possibly not p'(e.g. 16b).
- As far as wh-questions go, it leaves out quantificational answers (16c-g), as well as disjunctive answers.

Simple Answerhood v. Aboutness

(16) a. Christopher: Can I have some ice-cream then?
Dorothy: you can do if there is any. (BNC, KBW)

b. Anon: Are you voting for Tory?
Denise: I might. (BNC, KB?, slightly modified)

c. Dorothy: What did grandma have to catch?
Christopher: A bus. (BNC, KBW, slightly modified)

d. Rhiannon: How much tape have you used up?
Chris: About half of one side. (BNC, KB?)

e. Dorothy: What do you want on this?
Andrew: I would like some yogurt please. (BNC, KBW, slightly modified)

f. Elinor: Where are you going to hide it?
Tim: Somewhere you can't have it.(BNC, KBW)

g. Christopher: Where is the box?
Dorothy: Near the window. (BNC, KBW)

Aboutness via disjunction

- One straightforward way to enrich simple answerhood is to consider the relation that emerges by closing simple answerhood under disjunction. See Ginzburg and Sag (2000) for more discussion.

  (17) p is about q iff p entails a disjunction of simple answers.

Aboutness via disjunction

- If r : SimpleAns(q), then since r : AtomAns(q) or r : NegAtomAns(q), r is a record of the form in (i), p1 then is the value r gets on the propans field:

  (i) $\mathrm{r} = \begin{bmatrix} \mathrm{shortans} = \mathrm{a} \\ \mathrm{propans} = \mathrm{p} \end{bmatrix}$

(18) Given a question $q : (A)B$:

a. $\mathrm{Aboutness(q)} =_{def}$

$$\begin{bmatrix} \mathrm{r1} : \mathrm{SimpleAns(q)} \\ \mathrm{p1} = \mathrm{r1.propans} : \mathrm{Prop} \\ \mathrm{r2} : \mathrm{SimpleAns(q)} \\ \mathrm{p2} = \mathrm{r2.propans} : \mathrm{Prop} \\ \mathrm{propans} : \mathrm{Prop} \\ \mathrm{c} : \mathit{Entails}(\mathit{propans}, p1 \vee_{prop} p2) \end{bmatrix}$$

Answering a question with a question

(19) a. A: Who murdered Smith? B: Who was in town?

b. A: Who is going to win the race? B: Who is going to participate?

c. Carol: Right, what do you want for your dinner?
Chris: What do you (pause) suggest? (BNC, KbJ)

d. Chris: Where's mummy?
Emma: Mm?
Chris: Mummy?
Emma: What do you want her for? (BNC, KbJ)

- Detailed discussion of these on Fri in the class on theories of questions.

Outline

Asking, Asserting, Answering, and Accepting

- Broadly speaking queries and assertions are either *issue initiating*—they introduce an issue unrelated to those currently under discussion— or they are *reactive*—they involve a reaction to a previously raised issue.
- Accounting for the the reactive ones using DGB–based conversational rules is simple. These can also be used to explicate the effects *issue initiating* moves have. More on that a bit later.

Simple assertion and querying: ingredients

querying	assertion
LatestMove = Ask(A,q)	LatestMove = Assert(A,p)
A: push q onto QUD; release turn;	A: push p? onto QUD; release turn
B: push q onto QUD; take turn; make q—specific utterance take turn.	B: push p? onto QUD; take turn; Option 1: Discuss p? Option 2: Accept p
	LatestMove = Accept(B,p)
	B: increment FACTS with p; pop p? from QUD;
	A: increment FACTS with p; pop p? from QUD;

Simple assertion and querying: ingredients

- q-specific utterance: an utterance whose content is either a proposition p About max-qud (*partial answer*) or a question q_1 on which max-qud Depends (*sub-question*).

Asking, Asserting, Answering, and Accepting

- Two aspects of this protocol are not query specific:
  1. The protocol is like the ones we have seen for greeting and parting a 2-person turn exchange protocol (2-PTEP).
  2. The specification `make q-specific utterance` is an instance of a general constraint that characterizes the contextual background of reactive queries and assertions.

Asking, Asserting, Answering, and Accepting

- QSPEC: if *q* is QUD–maximal, then subsequent to this either conversational participant may make a move constrained to be *q*–specific (i.e. either About or Influencing *q*.).

(20) QSPEC

$$\begin{bmatrix} \text{pre} : \begin{bmatrix} \text{qud} = \left\langle \text{q, Q} \right\rangle : \text{poset(Question)} \end{bmatrix} \\ \text{effects} : \text{TurnUnderspec} \wedge_{merge} \\ \begin{bmatrix} \text{r} : \text{AbSemObj} \\ \text{R: IllocRel} \\ \text{LatestMove} = \text{R(spkr,addr,r)} : \text{IllocProp} \\ \text{c1} : \text{Qspecific(r,q)} \end{bmatrix} \end{bmatrix}$$

Turn Change and abbreviation

- Turnholder-Underspecified =

$$\left[\begin{array}{l}\text{pre} : \left[\begin{array}{l}\text{spkr} : \text{Ind}\\ \text{addr} : \text{Ind}\end{array}\right]\\ \text{effects} : \left[\begin{array}{l}\text{PrevAud} = \left\{\text{pre.spkr,pre.addr}\right\}) : \text{Set(Ind)}\\ \text{spkr} : \text{Ind}\\ \text{c1} : \text{member(spkr, PrevAud)}\\ \text{addr} : \text{Ind}\\ \text{c2}: \text{member(addr, PrevAud)} \wedge \text{addr} \neq \text{spkr}\end{array}\right]\end{array}\right]$$

Asking, Asserting, Answering, and Accepting

- The only query specific aspect of the querying protocol is:

(21) Ask QUD–incrementation:

$$\begin{bmatrix} \text{pre} : \begin{bmatrix} \text{q} : \text{Question} \\ \text{LatestMove} = \text{Ask(spkr,addr,q)}\text{:IllocProp} \end{bmatrix} \\ \text{effects} : \begin{bmatrix} \text{qud} = \left\langle \text{q,pre.qud} \right\rangle \text{: poset(Question)} \end{bmatrix} \end{bmatrix}$$

Asking, Asserting, Answering, and Accepting

- What are the components of the assertion protocol? Not specific to assertion is the fact that it is a 2-PTEP; similarly, the discussion option is simply an instance of QSPEC.
- This leaves two novel components: QUD incrementation with p? and acceptance.

Asking, Asserting, Answering, and Accepting

(22) Assert QUD–incrementation:

$$\begin{bmatrix} \text{preconds} & : \begin{bmatrix} \text{p} : \text{Prop} \\ \text{LatestMove} = \text{Assert(spkr,addr,p)}:\text{IllocProp} \end{bmatrix} \\ \text{effects} & : \begin{bmatrix} \text{qud} = \left\langle \text{p?,pre.qud} \right\rangle : \text{poset(Question)} \end{bmatrix} \end{bmatrix}$$

Asking, Asserting, Answering, and Accepting

- The second component of acceptance is the incrementation of FACTS by *p*.
- This is not quite as straightforward as it might seem: when FACTS gets incremented, we also need to ensure that p? gets downdated from QUD—only Nonresolved questions can be in QUD (see notes for discussion, leads to account of rhetorical questions.).
- In order to ensure that this is the case, we need to check for all existing elements of QUD that they are not resolved by the new value of FACTS.
- Hence, accepting p involves both an update of FACTS and a downdate of QUD—minimally just removing p?, potentially removing other questions as well.

Fact Update/ QUD Downdate

- NonResolve is a function that maps a partially ordered set of questions *poset*(q) and a set of propositions P to a partially ordered set of questions *poset*$'$(q) which is identical to *poset*(q) modulo those questions in *poset*(q) resolved by members of P.

(23) Fact Update/ QUD Downdate

$$\begin{bmatrix} \text{pre} : \begin{bmatrix} \text{p} : \text{Prop} \\ \text{LatestMove} = \text{Accept(spkr,addr,p)} \vee \\ \text{Confirm(spkr,addr,p)} \\ \text{qud} = \left\langle \text{p?,Q} \right\rangle : \text{poset(Question)} \end{bmatrix} \\ \text{effects} : \begin{bmatrix} \text{facts} = \text{pre.facts} \cup \left\{ \text{p} \right\} : \text{Set(Prop)} \\ \text{qud} = \text{NonResolve(Q,facts)} : \text{poset(Question)} \end{bmatrix} \end{bmatrix}$$

What drives the dialogue?

- Baseline rule: Free Speech; when QUD is empty one can say whatever one likes.
- This is true when one stands on the proverbial soap box or (various caveats) when sitting in a restaurant with friends . . .
- $$\left[\begin{array}{l} \text{pre} : \left[\text{qud} = \langle\rangle : \text{eset(question)}\right] \\ \text{effects} : \text{TurnUnderspec} \wedge_{merge} \\ \left[\begin{array}{l} \text{r} : \text{AbSemObj} \\ \text{R}: \text{IllocRel} \\ \text{LatestMove} = \text{R(spkr,addr,r)} \text{ IllocProp} \end{array}\right] \end{array}\right]$$

A simple example

(24) a. A: Hi
B: Hi
A: Who's coming tomorrow?
B: Several colleagues of mine (are coming).
A: I see.
B: Mike (is coming) too.

A simple example

Utt.	DGB Update (Conditions)	Rule
initial	MOVES $= \langle\rangle$	
	QUD $= \langle\rangle$	
	FACTS $=$ cg1	
1	LatestMove := Greet(A,B)	greeting
2	LatestMove := CounterGreet(B,A)	countergreeting
3	LatestMove := Ask(A,B,q0)	Free Speech
	QUD : $= \langle q0 \rangle$	Ask QUD–incrementation
4	LatestMove := Assert(B,A,p1)	QSPEC
	(About(p1,q0))	
	QUD : $= \langle p1?, q0 \rangle$	Assert QUD–incrementation
5	LatestMove := Accept(A,B,p1)	Accept
	QUD := $\langle q0 \rangle$	Fact update/QUD downdate
	FACTS := cg1 $\wedge$ p1	
6	LatestMove := Assert(B,A,p2)	QSPEC
	(About(p2,q0))	
	QUD : $= \langle p2?, q0 \rangle$	Assert QUD–incrementation

Assertoric Benchmark 2

- Assertion benchmark 2: Accommodate disagreement

(25) A(1): Who will agree to come?
B(2): Helen and Jelle.
A(3): I doubt Helen will want to come after last time.
B(4): I think she's forgiven and forgotten.
A(5): OK.

Assertoric Benchmark 2

Utt.	DGB Update (Conditions)	Rule
initial	MOVES = $\langle\rangle$	
	QUD = $\langle\rangle$	
	FACTS = cg1	
1	LatestMove := Ask(A,B,q0)	Free Speech
	QUD : = $\langle q0 \rangle$	Ask QUD–incrementation
2	LatestMove := Assert(B,A,p1)	QSPEC
	(About(p1,q0))	
	QUD : = $\langle p1?, q0 \rangle$	Assert QUD–incrementation
3	LatestMove := Assert(A,B,p2)	QSPEC
	(About(p2,p1?))	
	QUD := $\langle p2?, p1?, q0 \rangle$	Assert QUD–incrementation
4	LatestMove := Assert(B,A,p3)	QSPEC
	(About(p3,p2?))	
	QUD := $\langle p3?, p2?, p1?, q0? \rangle$	Assert QUD–incrementation
5	LatestMove := Accept(A,B,p3)	Accept
	QUD := $\langle p1?, q0 \rangle$	Fact update/QUD downdate

Query benchmark 4: query responses

- Query benchmark4: accommodate sub-questions.

(26) A(1): Who should we invite for tomorrow?
B(2): Who will agree to come?
A(3): Helen and Jelle and Fran and maybe Sunil.
B(4) : (a) I see. (b) So, Jelle I think.
A(5): OK.

Query benchmark 4: query responses

Utt.	DGB Update (Conditions)	Rule
initial	MOVES = $\langle\rangle$	
	QUD = $\langle\rangle$	
	FACTS = cg1	
1	LatestMove := Ask(A,B,q0)	Free Speech
	QUD : = $\langle q0 \rangle$	Ask QUD–incrementation
2	LatestMove := Ask(B,A,q1)	QSPEC
	Influence(q1,q0)	
	QUD : = $\langle q1, q0 \rangle$	Assert QUD–incrementation
3	LatestMove := Assert(A,B,p1)	QSPEC
	(About(p1,q1))	
	QUD := $\langle p1?, q1, q0 \rangle$	Assert QUD–incrementation
4a	LatestMove := Accept(B,A,p1)	Accept
	FACTS := cg1 ∪ {p1}	
	QUD := $\langle q0 \rangle$	Fact update/QUD downdate
4b	LatestMove := Assert(B,A,p2)	QSPEC
	(About(p2,q0))	
	QUD := $\langle p2?, q0 \rangle$	Assert QUD–incrementation
5	LatestMove := Accept(A,B,p2)	Accept
	FACTS := cg1 ∪ {p1,p2}&	
	QUD := $\langle q0 \rangle$	Fact update/QUD downdate

Query Benchmark 6 (partially)

- Same basic mechanism seems to regulate queries/assertions, across varying sizes of participant sets:

  (27) a. Monologue: self answering (*A: Who should we invite? Perhaps Noam.*)
  b. Dialogue: querier/responder (*A: Who should we invite? B: Perhaps Noam.*)
  c. Multilogue: discussed later, maybe.

- Query benchmark 6: ensure approach scales down to monologue

Austin, J. L. (1961). Truth. In J. Urmson and G. J. Warnock, editors, *Philosophical Papers*. Oxford University Press. Paper originally published in 1950.

Barwise, J. and Etchemendy, J. (1987). *The Liar*. Oxford University Press, New York.

Cooper, R. (2005). Austinian truth, attitudes and type theory. *Research on Language and Computation*, **3**(4), 333–362.

Ginzburg, J. (1995). Resolving questions, i. *Linguistics and Philosophy*, **18**, 459–527.

Ginzburg, J. (2012). *The Interactive Stance: Meaning for Conversation*. Oxford University Press, Oxford.

Ginzburg, J. and Sag, I. A. (2000). *Interrogative Investigations: the form, meaning and use of English Interrogatives*. Number 123 in CSLI Lecture Notes. CSLI Publications, Stanford: California.

Soames, S. (1985). Lost innocence. *Linguistics and Philosophy*, **8**, 59–71.