Incrementality, Alignment and Shared Utterances – Revisited

Matthew Purver
with Ruth Kempson, Pat Healey, Eleni Gregoromichelaki, Christine Howes, Wilfried Meyer-Viol, Graham White

The Dynamics of Conversational Dialogue (DynDial)
ESRC-RES-062-23-0962
www.kcl.ac.uk/research/groups/ds

December 4, 2009
Outline

1. Dialogue and Incrementality

2. Dynamic Syntax
   - A Quick Introduction to DS
   - DS and Dialogue Modelling

3. Empirical Investigations
   - Split Utterances - Corpus Study
   - Split Utterances - Experiments
   - Priming - Corpus Study

4. Dynamic Syntax & Type Theory with Records
   - A Quick Introduction to TTR
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
Outline

1. **Dialogue and Incrementality**
2. **Dynamic Syntax**
   - A Quick Introduction to DS
   - DS and Dialogue Modelling
3. **Empirical Investigations**
   - Split Utterances - Corpus Study
   - Split Utterances - Experiments
   - Priming - Corpus Study
4. **Dynamic Syntax & Type Theory with Records**
   - A Quick Introduction to TTR
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
Plenty of interest in dialogue
- Formal models of dialogue moves, IS update, fragments

Plenty of interest in incrementality
- Incremental processing in psycholinguistics
- Incremental parsing and generation in computational linguistics

Increasing interest in incrementality in dialogue
- e.g. [Schlangen and Skantze, 2009, Schuler et al., 2009]
- Speeding up dialogue systems
- Processing human-human dialogue
- People do it this way . . .
The Dynamics of Conversational Dialogue

- An ESRC project, joint between QMUL and KCL
  - formal/computational linguists, logicians, experimental psychologists
- Linguistic modelling using Dynamic Syntax
  [Kempson et al., 2001]
- Empirical studies using corpora and experiments
The Dynamics of Conversational Dialogue

- An ESRC project, joint between QMUL and KCL
  - formal/computational linguists, logicians, experimental psychologists
- Linguistic modelling using Dynamic Syntax
  [Kempson et al., 2001]
- Empirical studies using corpora and experiments
- Non-sentential utterances
- Clarification requests
- Split utterances
- Priming/alignment
Non-Sentential Utterances

- “Fragments” – utterances without an explicit verbal predicate
- Common in dialogue (> 10% of turns)
- Established formal treatments
Non-Sentential Utterances

- “Fragments” – utterances without an explicit verbal predicate
- Common in dialogue (> 10% of turns)
- Established formal treatments
- Question-Under-Discussion analysis:
  - “Who left?” → \( \lambda x.\text{leave}(x) \) → \( QUD = \lambda x.\text{leave}(x) \)
  - “John” → \( QUD(john) \) → \( \text{leave}(john) \)
- SDRT analysis:
  - “Who left?” → \( \alpha = \lambda x.\text{leave}(x) \)
  - “John” → \( \beta = P(john) \) ~\( QAP(\alpha, \beta), \beta = \text{leave}(john) \)
Clarification Requests

- Requesting clarification or confirmation of a previous utterance
  - Most commonly in the form of NSUs
- Common in dialogue (3-5% of turns)
- Established formal treatments
Clarification Requests

- Requesting clarification or confirmation of a previous utterance
  - Most commonly in the form of NSUs
- Common in dialogue (3-5% of turns)
- Established formal treatments

**Question-Under-Discussion analysis:**

```
"John left" → leave(john)  ~ QUD = λx.?assert(leave(x))

"John?"  →  QUD(john) → ?assert(leave(john))
```

**SDRT analysis:**

```
"John left" → α = leave(john)
"John?" → β = P(john)

~ QElab(α, β), β = clarify(leave(john))
```
Split Utterances

- Utterances containing a change in speaker
  - ...and therefore a change in hearer
Split Utterances

- Utterances containing a change in speaker
- ...and therefore a change in hearer

A: The profit for the group is 190,000.
B: Which is superb. ("expansion")
Split Utterances

- Utterances containing a change in speaker
  - ...and therefore a change in hearer

A: The profit for the group is 190,000.
B: Which is superb. (“expansion”)

A: Before that then if they were ill
G: They get nothing. (“completion”)
**Split Utterances**

- Utterances containing a change in speaker
  - ...and therefore a change in hearer

A: The profit for the group is 190,000.
B: Which is superb. (“expansion”)

A: Before that then if they were ill
G: They get nothing. (“completion”)

- Fundamental requirement for incremental processing
  - A good test for syntactic and semantic dependencies
  - A good test of NSU & CR processing

- Treatment for one particular kind [Rieser and Poesio, prep]
  - LTAG grammar and conversational-event-based plan recognition
Split utterances

- Particularly interesting from an incrementality point of view
Split utterances

- Particularly interesting from an incrementality point of view
- Where can splits occur?

(1) Hugh: Ruth visited
   Alex: Trecastle,
Split utterances

- Particularly interesting from an incrementality point of view
- Where can splits occur?

(1) Hugh: Ruth visited
    Alex: Trecastle, to go to the farm shop
Split utterances

- Particularly interesting from an incrementality point of view
- Where can splits occur?

(1) Hugh: Ruth visited
    Alex: Trecastle, to go to the farm shop

- Splits can occur across syntactic/semantic dependencies:

(2) A: Have you read ...
    B: any of your chapters? Not yet.
Split utterances

- Particularly interesting from an incrementality point of view
- Where can splits occur?

(1) Hugh: Ruth visited
   Alex: Trecastle, to go to the farm shop

- Splits can occur across syntactic/semantic dependencies:

(2) A: Have you read ...
   B: any of your chapters? Not yet.

- Splits can occur before proposition-intention fixable:

(3) A: They X-rayed me, and took a urine sample, took a blood sample. Er, the doctor
   B: Chorlton?
   A: Chorlton, mhm, he examined me, erm, he, he said now they were on about a slide [unclear] on my heart.
Priming and/or Alignment

- Tendency to repeat previously used material
  - words
  - syntactic structures
  - multi-word expressions
  - ways of referring
- Both self- and other-effects
- Incremental through a dialogue but also through an utterance
- How should this affect our model of processing?
  - ... especially in the case of split utterances
Outline

1. Dialogue and Incrementality

2. Dynamic Syntax
   - A Quick Introduction to DS
   - DS and Dialogue Modelling

3. Empirical Investigations
   - Split Utterances - Corpus Study
   - Split Utterances - Experiments
   - Priming - Corpus Study

4. Dynamic Syntax & Type Theory with Records
   - A Quick Introduction to TTR
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
Dynamic Syntax

- An inherently incremental grammatical framework
- Word-by-word incremental construction of semantic interpretation:
  - no autonomous level of syntax
  - “syntax” defined via constraints on incremental semantic structure-building
  - “grammar” is a set of procedures for incremental parsing
  - “trees” are semantic representations defined using LoFT [Blackburn and Meyer-Viol, 1994]
- Monotonic growth with underspecification-plus-enrichment
- Procedural definitions: constraints on how interpretations are built
End product of parsing is a semantic tree

Nodes decorated with $Ty()$ type and $Fo()$ formula labels

"John likes Mary":

$$
Ty(t), \\
Fo(like(john, mary))
$$

$$
Ty(e), \\
Fo(john)
$$

$$
Ty(e \rightarrow t), \\
Fo(\lambda x. like(x, mary))
$$

$$
Ty(e), \\
Fo(mary)
$$

$$
Ty(e \rightarrow (e \rightarrow t)), \\
Fo(\lambda y \lambda x. like(x, y))
$$

Daughter order does not reflect sentence order!

Nodes interpretable as terms in the $\lambda$-calculus

NPs map onto terms of type $e$ using the $\epsilon$-calculus.
**Actions as tree-building procedures**

- Words induce sets of actions to be carried out: “want”

**IF** \{ ?Ty(e → t) \}
**THEN** make(⟨↓₁⟩); go(⟨↓₁⟩);
put(Fo(Want’), Ty(e → (e → t)));
go(⟨↑₁⟩); make(⟨↓₀⟩);
go(⟨↓₀⟩); put(?Ty(e))

**ELSE** ABORT

- General computational actions are also available e.g. requirement fulfillment, beta-reduction
Unfolding then building up the tree

Processing *Someone fainted*

\(? Ty(t), \diamondsuit\)
Unfolding then building up the tree

Processing *Someone fainted*

\[ ?Ty(t) \]

\[ \diamond, ?Ty(e) \quad ?Ty(e \rightarrow t) \]
Processing *Someone fainted*

```
\diamond, \text{Ty}(e) \quad \text{?Ty}(e \rightarrow t)
```

\(\epsilon, x, \text{person}(x)\)

Unfolding then building up the tree
Processing *Someone fainted*

\[
\begin{align*}
&Ty(t) \\
&Ty(e) \quad ?Ty(e \rightarrow t), \Diamond \\
\epsilon, x, \text{person}(x)
\end{align*}
\]
Unfolding then building up the tree

Processing *Someone fainted*

```
?Ty(t)
  |                           |
  |                           |
  Ty(e)                      ?Ty(e → t), ♦
  |                           |
  ε, x, person(x)            λy.faint(y)
```

Matthew Purver et al.  
Edinburgh ICCS 04/12/09  
15/65
Unfolding then building up the tree

Processing *Someone fainted*

\[
faint(\epsilon, x, \text{person}(x))
\]

\[
faint(\epsilon, x, \text{person}(x))
\]

\[
Ty(t), \diamond
\]

\[
Ty(e)
\]

\[
\epsilon, x, \text{person}(x)
\]

\[
Ty(e \rightarrow t)
\]

\[
\lambda y. faint(y)
\]
Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.

Each word licensed must update partial tree towards the goal tree via *subsumption* constraint

* Generating *Someone fainted*

**GOAL TREE**

\[
Ty(t), \diamond
\]

\[
faint(\epsilon, \text{person}(x))
\]

\[
Ty(e),\quad Ty(e \rightarrow t)
\]

\[
\epsilon, x, \text{person}(x)\quad \lambda y. faint(y)
\]

**TEST TREE**

\[
?Ty(t), \diamond
\]
Generation

- Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.
- Each word licensed must update partial tree towards the goal tree via subsumption constraint

* Generating *Someone fainted*

GOAL TREE

\[ Ty(t), \quad \Diamond \]

\[ \text{faint}(\epsilon, \text{person}(x)) \]

\[ Ty(e), \quad Ty(e \rightarrow t) \]

\[ \epsilon, x, \text{person}(x) \quad \lambda y.\text{faint}(y) \]

TEST TREE

\[ ?Ty(t), \quad \Diamond, ?Ty(e) \quad ?Ty(e \rightarrow t) \]
Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.

Each word licensed must update partial tree towards the goal tree via subsumption constraint

* Generating *Someone fainted*

**GOAL TREE**

\[ Ty(t), \diamond \]

\[ faint(\epsilon, person(x)) \]

\[ Ty(e), \quad Ty(e \rightarrow t) \]

\[ \epsilon, x, person(x) \quad \lambda y. faint(y) \]

**TEST TREE**

\[ ?Ty(t), \quad Ty(e) \]

\[ \epsilon, x, person(x) \]

Gen: “Someone
• Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.

• Each word licensed must update partial tree towards the goal tree via subsumption constraint

* Generating *Someone fainted*

GOAL TREE

\[ Ty(t), \diamondsuit \]

\[ faint(\epsilon, person(x)) \]

\[ Ty(e), \epsilon, x, person(x) \]

\[ Ty(e \rightarrow t), \lambda y. faint(y) \]

TEST TREE

\[ Ty(e) \]

\[ Ty(e \rightarrow t), \epsilon, x, person(x), \diamondsuit \]

\[ Ty(t), \epsilon, x, person(x) \]

Gen: “Someone
Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.

Each word licensed must update partial tree towards the goal tree via subsumption constraint

* Generating *Someone fainted*

**GOAL TREE**

\[
\begin{align*}
&Ty(t), \diamond \\
&\text{faint}(\epsilon, \text{person}(x)) \\
&Ty(e), \quad Ty(e \to t) \\
&\epsilon, x, \text{person}(x) \quad \lambda y.\text{faint}(y)
\end{align*}
\]

**TEST TREE**

\[
\begin{align*}
&Ty(e) \\
?Ty(e \to t), \diamond \\
&\epsilon, x, \text{person}(x) \quad \lambda y.\text{faint}(y)
\end{align*}
\]

Gen: “Someone fainted”
Speakers go through the same tree-growth actions, except they also have a somewhat richer goal tree.

Each word licensed must update partial tree towards the goal tree via *subsumption* constraint

* Generating *Someone fainted*

**GOAL TREE**

\[
Ty(t), \Diamond
\]
\[
faint(\varepsilon, person(x))
\]
\[
Ty(e), \quad Ty(e \to t)
\]
\[
\varepsilon, x, person(x) \quad \lambda y. faint(y)
\]

**TEST TREE**

\[
Ty(t), \Diamond
\]
\[
faint(\varepsilon, x, person(x))
\]
\[
Ty(e), \quad Ty(e \to t)
\]
\[
\varepsilon, x, person(x) \quad \lambda y. faint(y)
\]

Gen: “Someone fainted”
Underspecification: structural

- “Unfixed” nodes - building underspecified tree relations

\[
\begin{align*}
?Ty(t), Tn(0) \\
Ty(e), Mary' \quad ?Ty(e) \quad ?Ty(e \rightarrow t)
\end{align*}
\]

- Left-dislocation “Mary, John likes”
Underspecification: content

- Pronouns project **META-VARIABLES (U)**
- Substituted by item from context during construction
Underspecification: content

- Pronouns project META-VARIABLES (U)
- Substituted by item from context during construction

(1) Someone smoked  He fainted.
Underspecification: content

- Pronouns project **META-VARIABLES (U)**
- Substituted by item from context during construction

1. Someone smoked He fainted.

**Tree as Context:** **Tree under Construction:**

\[
\text{smoke}(\epsilon, x, \text{person}(x))
\]

\[
\epsilon, x, \text{person}(x) \xrightarrow{\text{SUBSTITUTION}} \lambda y.\text{smoke}(y)
\]

\[
?\exists x \text{Fo}(x) \xrightarrow{\text{U}} \lambda y.\text{faint}(y)
\]

Matthew Purver et al. Edinburgh ICCS 04/12/09 18/65
Underspecification: ellipsis

- Auxiliaries also project META-VARIABLES (V)
  Substituted by item from context in the same way
Underspecification: ellipsis

- Auxiliaries also project META-VARIABLES (V)
  Substituted by item from context in the same way

(1) John smoked  Bill did too.
Auxiliaries also project **META-VARIABLES** \((V)\)

Substituted by item from context in the same way

\[(1)\] John smoked Bill did too.

**Tree as Context:**  **Tree under Construction:**

- `smoke(john)`
- `\(\lambda y. smoke(y)\)`
- `U,` `?\exists x. Fo(x)`
- `diamond`
Underspecification: ellipsis

- Auxiliaries also project META-VARIABLES ($V$)
  Substituted by item from context in the same way

(1) John smoked  Bill did too.

Tree as Context:  Tree under Construction:

```
smoke(john)
```

- Alternatively can use actions from context (sloppy readings)
- Simple model of context containing previous (partial) trees and action sequences
Context-dependence: **LINKed tree-pairs**

- **Relative clauses**: pairs of LINKed trees evaluated as conjunction
  - e.g. Bill, who fainted, smokes.
  
  \[
  \text{Smoke}'(\text{Bill}') \land \text{Faint}'(\text{Bill}')
  \]

```
                  L
                 /\     /
                Smoke'   Faint'(Bill')
                  /  \
              Bill'   Faint'
```

Matthew Purver et al.  Edinburgh ICCS 04/12/09
Appositions as L\textsc{ink}ed trees

e.g. A friend, a musician, smokes.
Appositions as **LINKed** trees

e.g. A friend, a musician, smokes.

- Partial tree as context with term enriched by **LINKed** tree of same type
- Parsing *A friend, a musician*

```
\epsilon . x . \text{Friend}'(x), \Diamond \quad ? Ty(e \rightarrow t) \quad Ty(e) \quad \epsilon, x, \text{Musician}'(x)
```

Matthew Purver et al. Edinburgh ICCS 04/12/09 21/65
Appositions as LINKed trees

e.g. A friend, a musician, smokes.

- Partial tree as context with term enriched by LINKed tree of same type
- Parsing *A friend, a musician*

\[
\begin{align*}
\epsilon, x, \text{Friend}'(x), \& & ? Ty(t) \\
\epsilon, x, \text{Musician}'(x) & \rightarrow Ty(e) \\
\epsilon, x, \text{Friend}'(x) \wedge \text{Musician}'(x)
\end{align*}
\]

Evaluation of LINKed nodes both of type \(e\) yields composite term:
\[
\epsilon, x, \text{Friend}'(x) \wedge \text{Musician}'(x)
\]

Final formula: \(Smoke'(\epsilon, x, \text{Friend}'(x) \wedge \text{Musician}'(x))\)
DS and Split Utterances

- DS *seems* well suited for split utterances
- Inherent word-by-word incrementality in parsing and generation
- Use of semantic constraints rather than “syntax”
- Use of same actions and partial structures in parsing and generation
DS and Split Utterances

- DS *seems* well suited for split utterances
- Inherent word-by-word incrementality in parsing and generation
- Use of semantic constraints rather than “syntax”
- Use of same actions and partial structures in parsing and generation
- Is it too general (what are the real constraints)?
- Is it too simplistic (what do split utterances *mean*)?
**DS and Non-Sentential Utterances**

- DS *seems* well suited for non-sentential utterances
- LINK mechanism for apposition allows general fragment processing
- Semantics very underspecified
- Advantages & disadvantages
DS and Non-Sentential Utterances

- DS *seems* well suited for non-sentential utterances
- LINK mechanism for apposition allows general fragment processing
- Semantics very underspecified
- Advantages & disadvantages
- Is it too simplistic (what do NSUs *mean*)?
DS and Priming/Alignment

- DS *seems* well suited to explain priming/alignment phenomena
- Use of actions at all levels of processing
- Availability of recent action (sequences) for re-use
  - Lexical choice and disambiguation
  - Syntactic phenomena (e.g. DO/PO alternation [Branigan et al., 2000])
  - Semantic/pragmatic phenomena (e.g. routines [Garrod and Anderson, 1987], ellipsis construal [Hardt, 2008])
DS seems well suited to explain priming/alignment phenomena

- Use of actions at all levels of processing
- Availability of recent action (sequences) for re-use
  - Lexical choice and disambiguation
  - Syntactic phenomena (e.g. DO/PO alternation [Branigan et al., 2000])
  - Semantic/pragmatic phenomena (e.g. routines [Garrod and Anderson, 1987], ellipsis construal [Hardt, 2008])

- Does this really explain general (non-lexical) effects?
- Re-use of specific lexical action sequences should lead to priming
- What about re-use of computational action sequences?
Outline

1. Dialogue and Incrementality

2. Dynamic Syntax
   - A Quick Introduction to DS
   - DS and Dialogue Modelling

3. Empirical Investigations
   - Split Utterances - Corpus Study
   - Split Utterances - Experiments
   - Priming - Corpus Study

4. Dynamic Syntax & Type Theory with Records
   - A Quick Introduction to TTR
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
Empirical Investigations

- What do these phenomena really look like?
- Do split utterances really behave the way we think?
  - How common are they?
  - Where does the split happen?
  - What do they mean?
- What’s the deal with lexical and syntactic priming?
  - Do we see them in ordinary dialogue?
  - Can we tell which effect is greater?
BNC Corpus Study

- Take a portion of the BNC (as annotated by Fernández, 2006)
- Find all the split utterances
  - not just other-person cases [Skuplik, 1999, Szczepak, 2000]
  - or particular CA phenomena [Lerner, 2004, Rühlemann, 2007]
- See how often they occur, for same- and other-person cases
- See how variable the split point is
  - Completeness/constituency of the two halves
    - completion/expansion
  - Dependencies across the split
- See what happens in between ...
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
Corpus Study: Annotation Schema

- A1: I’ll definitely use that ← END-COMPLETE=Y →
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- A1: in getting to know
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- A1: in getting to know

CONTINUES
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- A1: in getting to know

START-COMPLETE=N

END-COMPLETE=N
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- A1: in getting to know
- A1: new year seven
Corpus Study: Annotation Schema

- A1: I’ll definitely use that

- A1: in getting to know

- A1: new year seven

CONTINUES
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- A1: in getting to know
- A1: new year seven

END-COMPLETE=Y
START-COMPLETE=N
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- UX: [reading] Get a headache?
- A1: in getting to know

- A1: new year seven
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- UX: [reading] Get a headache?
- A1: [in getting to know]
- A2: [Year seven]
- A1: new year seven
Corpus Study: Annotation Schema

- A1: I’ll definitely use that
- UX: [reading] Get a headache?
- A1: [in getting to know]
- A2: [Year seven]
- A1: new [year seven]
- A2: [Oh yeah] for year seven
Corpus Study: Observations

- They’re common: 19% of all contributions continue something
- 85% of these are same-person cases
- 15% are other-person cases
  - this is about 3% of all dialogue contributions (i.e. about as common as clarification)
Corpus Study: Observations

- They’re common: 19% of all contributions continue something
- 85% of these are same-person cases
- 15% are other-person cases
  - this is about 3% of all dialogue contributions (i.e. about as common as clarification)
- Many are within-turn (although these are still interesting!)
- Some may be artefacts of the BNC transcription protocol
  - overlapping speech forces a split into two contributions
- But even without all these, 10% of contributions are SUs
They’re not always adjacent:

- Same-person: 35% separated by a backchannel, 20% by 1 or more other turns
- Other-person: 5% separated by a backchannel, 5% by 1 or more other turns

Intervening material is often a clarification:

1. J: If you press N
   S: N?
   J: N for name, it’ll let you type in the document name.

The antecedent for clarification is often incomplete . . .
Corpus Study: Observations

- The first part is often (but not always) incomplete: 26-28% of cases
- Some neat “syntactic” categories exist, as expected
- But these only cover 50-60% of cases
- Splits can apparently happen at any syntactic point, including inside NPs/PPs:
  
  (2) F: We are going to call you the  
  U: Wallering
  
  (3) A: And they went over just to be fitted with the  
  G: just fitted with the brass
- Note the presence of repair: only 5% of cases
Corpus Study: Observations

- Continuations often don’t perform the same *function* as the antecedent:
  
  (4) G: Had their own men  
  A: unload the boats?  
  G: unload the boats, yes.

  (5) J: How does it generate?  
  M: It’s generated with a handle and  
  J: Wound round?  
  M: Yes, wind them round

- Very often a clarification request, but others possible e.g. confirmation, reformulation
- Not quite as simple as just completing a semantic structure
Some conclusions play right into DS’s hands . . .

- Splits across syntactic & semantic constraints
- Not always collaborative as per [Rieser and Poesio, prep]
- Intervening turns use incomplete antecedents (partial trees)
Corpus Study: Conclusions

- Some conclusions play right into DS’s hands …
  - Splits across syntactic & semantic constraints
  - Not always collaborative as per [Rieser and Poesio, prep]
  - Intervening turns use incomplete antecedents (partial trees)
- …but some don’t:
  - Repair
  - Clarifications
Experimental Study: the DiET chattool

- Corpora tell us nothing about processing questions
- DiET: a toolbox for experimenting with dialogue [Healey et al., 2003]
- Basic setup: a multi-way chat tool, a bit like MSN Messenger
- Communication is mediated by a server, allowing controlled manipulations
  - transform real turns
  - introduce “fake” turns
- Use this to introduce split utterances, and observe the effects
sam, E and cyn are having a three-way conversation
Dialogue and Incrementality
Dynamic Syntax
Empirical Investigations
Dynamic Syntax & Type Theory with Records

Split Utterances - Corpus Study
Split Utterances - Experiments
Priming - Corpus Study

sam types a turn
Dialogue and Incrementality
Dynamic Syntax
Empirical Investigations
Dynamic Syntax & Type Theory with Records

Split Utterances - Corpus Study
Split Utterances - Experiments
Priming - Corpus Study

Matthew Purver et al.
Edinburgh ICCS 04/12/09

turn typed by sam intercepted by server
First part of SU relayed to E ...

Dialogue and Incrementality
Dynamic Syntax
Empirical Investigations
Dynamic Syntax & Type Theory with Records

Matthew Purver et al. Edinburgh ICCS 04/12/09
... and cyn
Second part of SU relayed to E...
... and cyn – with apparent origin E
Experimental Study: An example

- ‘Bancil’ types:
  the only loss here is a pilot and a father which is kinda bad but someones gotta go

- ‘Aryan’ sees (AA):
  Bancil: the only loss here is a pilot and a father
  Aryan: which is kinda bad but someones gotta go

- ‘efparxng’ sees (AB):
  Bancil: the only loss here is a pilot and a father
  Aryan: which is kinda bad but someones gotta go
Experimental Study: Results

- We can observe: typing time of turn, number of ‘deletes’ used
  - next turn effects: the next participant to type
  - global effects: all participants turns until next intervention
- We can compare: speaker switch (AA/BB vs. AB/BA)
- We can compare: floor change (AA/BA vs. BB/AB)
- We can compare: first/second part coherence (Y/N)
Main effect of *floor change* on typing time of turn

If the second part of the SU is misattributed (AB & BB cases), people take **longer** constructing responses.

Next turns:
(F(3,249) = 7.13, p < 0.05)
Globally:
(F(3,486) = 3.78, p < 0.05)
Experimental Study: Results

- Main effect of *speaker switch* on number of ‘deletes’

If the SU appears to be a cross-person one (AB & BA cases), people use **fewer** deletes in their responses.

Next turns:
(F(3,249) = 6.26, p < 0.05)
Globally:
(F(3,486) = 9.23, p < 0.05)
Experimental Study: Results

- Interaction effect of 1st- x 2nd-part coherence on ‘deletes’

If BOTH parts of the split could standalone (YY), or if NEITHER part could (NN), then participants use fewer deletes in their first response.

\[ F(249) = 4.05, \ p < 0.05 \]
Experimental Study: Conclusions

- Lack of speaker-switch effect on typing time suggests ease of processing
- Effect on deletes may be due to apparent party formation?
- Effect of floor change may be due to interference in turn-taking organisation
- Effect of 1st/2nd-part coherence suggests “garden-path”-style revision
- We’re worried about the robustness of the setup . . .
  - . . . a character-by-character version is almost complete
Primed: Designing a corpus experiment

- DS seems to predict lexical(-syntactic) effects more than general syntactic effects.

- Previous dialogue experiments (e.g. [Reitter et al., 2006]) suggest that:
  - General syntactic effects are stronger in task-specific dialogue than in general conversation.
  - General syntactic effects are stronger within-person than cross-person.

- But no direct control condition:
  - What about dialogue structure effects?
  - How similar would recent turns be by chance?
  - Switchboard corpus is strange.
Corpus experiment: Method

- DCPSE corpus, all 2-person dialogues from 3 largest genre samples:
  - face-to-face formal (60 dialogues, 90,000 words)
  - face-to-face informal (91 dialogues, 403,000 words)
  - telephone conversations (89 dialogues, 77,000 words)
- For each dialogue $D$, create a “fake” control dialogue:
  - keep all turns from first speaker $S_1D$
  - choose a different dialogue $D'$, matching by length and within genre
  - interleave the turns from $S_1D$ with those from $S_2D'$
- Compare average turn similarity between real and control dialogues
Corpus experiment: Method

A: Hello
B: Hi
A: How are you?
B: Fine - you?
A: Yeah fine thanks
B: Uh-huh

A’: Hi
B’: Hello
A’: What’s up?
B’: Not much
A’: Me neither
B’: Uh-huh
Corpus experiment: Method

A: Hello

A: How are you?

B’: Not much

A: Yeah fine thanks

B’: Uh-huh
Corpus experiment: Method

A: Hello
B’: Hello
A: How are you?
B’: Not much
A: Yeah fine thanks
B’: Uh-huh
Corpus experiment: Lexical results

- Lexical similarity expressed via word pair kernel:
  - number of matching word pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{lex} = \frac{N_{AB}}{\sqrt{N_{AA} \cdot N_{BB}}}$
Corpus experiment: Lexical results

- Lexical similarity expressed via word pair kernel:
  - number of matching word pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{\text{lex}} = \frac{N_{AB}}{\sqrt{N_{AA} \cdot N_{BB}}}$

- ANOVA for real vs. control shows a reliable difference:
  $$F_{(1,253)} = 106.55, \ p = 0.00$$

- Real dialogues mean other-person similarity
  $$S_{\text{lex}} = 0.094 \ (SD = 0.04)$$

- Control dialogues mean other-person similarity
  $$S_{\text{lex}} = 0.059 \ (SD = 0.03)$$
Corpus experiment: Syntactic results

- Syntactic similarity via tree kernel (variant of [Moschitti, 2006]):
  - number of matching non-terminal syntactic rule pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{syn} = \frac{N_{AB}}{\sqrt{N_{AA}.N_{BB}}}$
Corpus experiment: Syntactic results

- Syntactic similarity via tree kernel (variant of [Moschitti, 2006]):
  - number of matching non-terminal syntactic rule pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{syn} = \frac{N_{AB}}{\sqrt{N_{AA}N_{BB}}}$

- ANOVA for real vs. control shows no reliable difference
  - $F_{(1,253)} = 1.32$, $p = 0.25$

- Real dialogues mean other-person similarity
  - $S_{syn} = 0.19$ ($SD = 0.06$)

- Control dialogues mean other-person similarity
  - $S_{syn} = 0.18$ ($SD = 0.06$)
Corpus experiment: Syntactic results

- Syntactic similarity via tree kernel (variant of [Moschitti, 2006]):
  - number of matching non-terminal syntactic rule pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{syn} = \frac{N_{AB}}{\sqrt{N_{AA}N_{BB}}}$
- ANOVA for real vs. control shows no reliable difference
  \[ F_{(1,253)} = 1.32, \quad p = 0.25 \]
- Real dialogues mean other-person similarity
  \[ S_{syn} = 0.19 \quad (SD = 0.06) \]
- Control dialogues mean other-person similarity
  \[ S_{syn} = 0.18 \quad (SD = 0.06) \]
- But: a reliable effect of genre ($F_{(2,237)} = 20.13, \quad p = 0.00$):

<table>
<thead>
<tr>
<th></th>
<th>formal</th>
<th>informal</th>
<th>telephone</th>
</tr>
</thead>
<tbody>
<tr>
<td>real</td>
<td>0.21</td>
<td>0.19</td>
<td>0.17</td>
</tr>
<tr>
<td>control</td>
<td>0.21</td>
<td>0.18</td>
<td>0.16</td>
</tr>
</tbody>
</table>
Following [Reitter et al., 2006], we can examine average similarity to recent turns. Syntactic self-similarity shows a significant linear trend ($p = 0.00$). Syntactic other-similarity not reliable ($p = 0.15$). Plotting real and control dialogues is interesting though. Are we just seeing the effect of dialogue structure?
Corpus experiment: Conclusions

- We can measure the effect of lexical priming
- We can’t measure the effect of syntactic priming
  - We don’t have enough statistical power here to say there’s no effect
  - But it must be quite small (relative to the lexical effect)
- We can measure the effect of genre on syntactic similarity
  - This seems to agree with (some of) [Reitter et al., 2006]’s results
- We’d like more (parsed) data – working on the BNC now . . .
Outline

1. Dialogue and Incrementality
2. Dynamic Syntax
   - A Quick Introduction to DS
   - DS and Dialogue Modelling
3. Empirical Investigations
   - Split Utterances - Corpus Study
   - Split Utterances - Experiments
   - Priming - Corpus Study
4. Dynamic Syntax & Type Theory with Records
   - A Quick Introduction to TTR
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
**DS and TTR: Motivation**

- So far, we’re happy that we’re going in roughly the right direction:
  - Split utterances seem to fit the DS approach
  - Priming results fit with prediction (so far as we can tell)
So far, we’re happy that we’re going in roughly the right direction:

- Split utterances seem to fit the DS approach
- Priming results fit with prediction (so far as we can tell)

For a proper treatment of NSUs and SUs, DS needs more structured representations

- Responsibility for a (sub-)utterance (speaker, hearer?)
- Utterance function (speech acts?)

Want to avoid forcing this into all representations . . .

- What should really be in the grammar?
See [Betarte and Tasistro, 1998], following Martin-Löf.

Records are sequences of label/value pairs:

\[
\begin{bmatrix}
  l_1 &= v_1 \\
  l_2 &= v_2 \\
  l_3 &= v_3
\end{bmatrix}
\]

Record types are sequences of label/type pairs:

\[
\begin{bmatrix}
  l_1 : T_1 \\
  l_2 : T_2 \\
  l_3 : T_3
\end{bmatrix}
\]

Record types are true iff they are inhabited/witnessed:

- there exists at least one record of that type
- successful type judgements for each label/value pair:
  \( v_1 : T_1, \ v_2 : T_2, \ v_3 : T_3 \)
Type Theory With Records

- Types can be *dependent* on earlier (higher-up) types:

\[
\begin{align*}
{l_1} &: T_1 \\
{l_2} &: T_2(l_1) \\
{l_3} &: T_3(l_1, l_2)
\end{align*}
\]

- We can have *nested* records and record types:

\[
\begin{align*}
{l_1} &: T_1 \\
{l_2} &: \left[ \begin{align*}
{l_1}' &: T_1' \\
{l_2}' &: T_2'
\end{align*} \right] \\
{l_3} &: T_3(l_1, l_2, l_1', l_2')
\end{align*}
\]
We can have functional record types:

\[
\lambda r : \left[\begin{array}{l}
l_1 : T_1 \\
l_2 : T_2
\end{array}\right] \left(\begin{array}{l}
l_3 : T_3 \\
l_4 : T_4(r.l_1, r.l_2)
\end{array}\right)
\]

Given a record \( r = \left[\begin{array}{l}
l_1 = v_1 \\
l_2 = v_2
\end{array}\right] \) of type \( \left[\begin{array}{l}
l_1 : T_1 \\
l_2 : T_2
\end{array}\right] \),
Type Theory With Records

- Used for sentential semantics, e.g. [Cooper, 2005, Ginzburg, 2005]

  “A man left”:

  \[
  \begin{array}{c}
  x : \text{man} \\
  p : \text{leave}(x)
  \end{array}
  \]

  for truth: \(x\) must be a man, \(p\) a proof that \(x\) left

- Similarities to DRT representation:

  \[
  \begin{array}{c | c | c}
  \hline
  \text{ } & \text{man}(x) & \text{leave}(x) \\
  \hline
  x & \text{man}(x) & \text{leave}(x) \\
  \hline
  \end{array}
  \]

- “Every man left”:

  \[
  \lambda r : \begin{array}{c}
  x : \text{man}
  \end{array} \left( \begin{array}{c}
  p : \text{leave}(r.x)
  \end{array} \right)
  \]
Type Theory With Records

- Used for dialogue modelling in the information-state-based tradition
The best of both worlds?

- TTR gives us a type-theoretic framework, applicable to dialogue phenomena
- DS gives us an incremental framework using type theory as an underlying mechanism
- Can we combine the two?
The best of both worlds?

- TTR gives us a type-theoretic framework, applicable to dialogue phenomena
- DS gives us an incremental framework using type theory as an underlying mechanism
- Can we combine the two?

\[ \diamond, \text{leave}(john), \text{Ty}(t) \]
\[ \begin{array}{c}
  \text{john,} \\
  \text{Ty(e)}
\end{array} \quad \begin{array}{c}
  \lambda x.\text{leave}(x), \\
  \text{Ty(e \rightarrow t)}
\end{array} \]
The best of both worlds?

- TTR gives us a type-theoretic framework, applicable to dialogue phenomena.
- DS gives us an incremental framework using type theory as an underlying mechanism.
- Can we combine the two?

\[ \diamond, \text{leave}(john), \text{Ty}(t) \]

\[ \text{john}, \text{Ty}(e), \lambda x.\text{leave}(x), \text{Ty}(e \to t) \]

\[ \diamond, \left[ x : \text{john} \right], \lambda [x]. \left[ e : \text{leave}(x) \right] \]

\[ \left[ x : \text{john} \right], \lambda [x]. \left[ p : \text{leave}(x) \right] \]
A simple version

- Replace $Fo()$ epsilon-calculus labels with TTR record types
Replace $Fo()$ epsilon-calculus labels with TTR record types

- $Ty(e), Ty(e \rightarrow t), \lambda x.\text{leave}(x)$
- $\diamond, ?Ty(t)$
- IF $?Ty(e)$
  THEN put($Ty(e)$)
  put($Fo(john)$)
- ELSE abort
A simple version

- Replace $Fo()$ epsilon-calculus labels with TTR record types

Diamond, $\text{Ty}(t)$

$\text{Ty}(e)$, $[x : john]$ $\text{Ty}(e \rightarrow t)$, $\lambda x.\text{leave}(x)$

IF $\text{Ty}(e)$
THEN $\text{put} (\text{Ty}(e))$
ELSE abort
A simple version

- Replace $Fo()$ epsilon-calculus labels with TTR record types
- Interpret $Ty()$ simple type labels as referring to *final* TTR field type

\[
\begin{align*}
&\diamond, ?Ty(t) \\
&Ty(e), \\
&\left[ x : john \right] \\
&Ty(e \to t), \\
&\lambda x. leave(x)
\end{align*}
\]

IF ?$Ty(e)$
THEN put($Ty(e)$)
ELSE put($\left[ x : john \right]$) \\
ELSE abort
A simple version

- Replace $Fo()$ epsilon-calculus labels with TTR record types
- Interpret $Ty()$ simple type labels as referring to final TTR field type

\[
\begin{align*}
\diamond, \ ?Ty(t) \\
Ty(e), \\
[ x : john ] & \quad \quad Ty(e \rightarrow t), \\
\lambda [ x : e ] & . \quad \quad \lambda \ [ x : e ] \ . \ \\
& \quad \quad \quad \quad \begin{array}{l}
 x : e \\
 p : leave(x)
\end{array}
\end{align*}
\]
A simple version

- Replace \( Fo() \) epsilon-calculus labels with TTR record types
- Interpret \( Ty() \) simple type labels as referring to final TTR field type
- Function application as before for DS elimination process

\[
\begin{align*}
\diamond, Ty(t), & \quad \left[ x : john \atop p : leave(x) \right] \\
Ty(e), & \quad Ty(e \rightarrow t), \\
\left[ x : john \right] & \quad \lambda \left[ x : e \right] \cdot \left[ x : e \atop p : leave(x) \right]
\end{align*}
\]
For LINKed trees, we need conjunction

“Bill, who fainted, smokes.”

\[\text{smoke}(\text{bill}) \land \text{faint}(\text{bill})\]

\[
\text{faint}(\text{bill})
\]

\[
\text{bill} \lambda x.\text{smoke}(x)
\]

\[
\text{bill} \lambda x.\text{faint}(x)
\]
Adding in LINK relations

- For LINKed trees, we need conjunction
- Use extension: \( \oplus r_1 \oplus r_2 \) adds \( r_2 \) to the end of \( r_1 \)
  - only for distinct labels

“Bill, who fainted, smokes.”

\[
\text{smoke}(\text{bill}) \land \text{faint}(\text{bill})
\]

\[
\begin{array}{c}
\text{bill} \\
\lambda x. \text{smoke}(x)
\end{array} \\
\begin{array}{c}
\text{faint}(\text{bill}) \\
\lambda x. \text{faint}(x)
\end{array}
\]
For LINKed trees, we need conjunction

Use extension: \( \oplus r_1 \oplus r_2 \) adds \( r_2 \) to the end of \( r_1 \)
  - only for distinct labels

“Bill, who fainted, smokes.”

\[
\begin{align*}
\text{smoke}(bill) \land \text{faint}(bill) \\
\text{faint}(bill) \\
\text{smoke}(bill)
\end{align*}
\]

\[
\begin{align*}
\text{faint}(bill) \\
\text{smoke}(bill)
\end{align*}
\]

\[
\begin{align*}
\left[ p : \text{smoke}(bill) \right] \\
\left[ q : \text{faint}(bill) \right]
\end{align*}
\]

\[
\begin{align*}
\left[ x : \text{bill} \right] \lambda [x]. \left[ p : \text{smoke}(x) \right]
\end{align*}
\]

\[
\begin{align*}
\left[ q : \text{faint}(bill) \right]
\end{align*}
\]

\[
\begin{align*}
\left[ x : \text{bill} \right] \lambda [x]. \left[ q : \text{faint}(x) \right]
\end{align*}
\]
Can we do better?

- From an implementational point of view, this is OK
- But we’re in danger of losing something
  - DS trees as they stand have a direct correspondence with semantics
  - Nodes are terms in the lambda-calculus
    - (Unreduced terms at daughter nodes)
  - What exactly are they now?
- Would prefer tree definitions via TTR(-compatible) logic
  - Type dependencies rather than abstraction (via [Kopylov, 2003] dependent intersection)
  - Initial versions for basic framework; LINK more complicated
    - (Meyer-Viol/White, forthcoming)
LINK as optional enrichment process

- Add utterance-event information
- Add speaker (or rather “responsible party”) information

“John left”

\[ \diamond, Ty(t), \begin{bmatrix} x : john \\ p : leave(x) \end{bmatrix} \]

\[ Ty(e), \begin{bmatrix} x : john \end{bmatrix} \quad \lambda [x]. \begin{bmatrix} p : leave(x) \end{bmatrix} \]

\[ Ty(e \rightarrow t), \begin{bmatrix} x : john \end{bmatrix} \]
**LINK as optional enrichment process**

- Add utterance-event information
- Add speaker (or rather “responsible party”) information

"John left"

\[ u_0 : \text{utt} \rightarrow \text{event} \]

\[ a : \text{spkr}(u_0) \]

\[ x : \text{john} \]

\[ p : \text{leave}(x) \]

\[ \text{Ty}(t), \quad \text{Ty}(e \rightarrow t), \quad \lambda [x]. [ p : \text{leave}(x) ] \]
LINK as optional enrichment process

- Add utterance-event information
- Add speaker (or rather “responsible party”) information

“John left”

\[
\begin{align*}
\diamond, \, Ty(t), & \\
\[ & u_0 : \text{utt} - \text{event} \\
\[ & a : \text{spkr}(u_0) \\
\[ & x : \text{john} \\
\[ & p : \text{leave}(x) \\
\end{align*}
\]

\[ Ty(e), \quad \lambda [x] . [ \, p : \text{leave}(x) \, ] \]

- Allow *optional* inferences about speech acts
**LINK as optional enrichment process**

- Add utterance-event information
- Add speaker (or rather “responsible party”) information

```
“John left”

◊, Ty(t),

\[
\begin{align*}
    u_0 & : \text{utt} \rightarrow \text{event} \\
    a & : \text{spkr}(u_0) \\
    x & : \text{john} \\
    p & : \text{leave}(x)
\end{align*}
\]

\[
\begin{align*}
    u_0 & : \text{utt} \rightarrow \text{event} \\
    a & : \text{spkr}(u_0) \\
    x & : \text{john} \\
    p & : \text{leave}(x) \\
    q & : \text{assert}(u_0, a, p)
\end{align*}
\]

\[
\begin{align*}
    \text{Ty}(e), & \quad \text{Ty}(e \rightarrow t), \\
    [ x : \text{john } ] & \quad \lambda [x]. [ p : \text{leave}(x) ]
\end{align*}
\]

- Allow *optional* inferences about speech acts
**LINK as optional enrichment process**

- Speech act inferences conditional on syntax/semantics

```
"Did John leave?"

◊, +Q, Ty(t),

\[
\begin{aligned}
  u_0 & : \text{utt} \rightarrow \text{event} \\
  a & : \text{spkr}(u_0) \\
  x & : \text{john} \\
  p & : \text{leave}(x)
\end{aligned}
\]

\[ Ty(e),
\]

\[ Ty(e \rightarrow t),
\]

\[ \lambda [x]. [ p : \text{leave}(x) ] \]
LINK as optional enrichment process

- Speech act inferences conditional on syntax/semantics

```
"Did John leave?"

\[ \diamond, +Q, Ty(t), \begin{cases} u_0 : utt \rightarrow event \\
    a : spkr(u_0) \\
    x : john \\
    p : leave(x) \end{cases} \begin{cases} u_0 : utt \rightarrow event \\
    a : spkr(u_0) \\
    x : john \\
    p : leave(x) \\
    q : ask(u_0, a, ?p) \end{cases} \]

\[ Ty(e), [x : john] \]

\[ Ty(e \rightarrow t), \lambda [x]. [p : leave(x)] \]
```
LINK as optional enrichment process

- Speech act inferences conditional on syntax/semantics

```
“Did John leave?”

◊, +Q, Ty(t),

\[
\begin{align*}
  u_0 & : \text{utt} - \text{event} \\
  a & : \text{spkr}(u_0) \\
  x & : \text{john} \\
  p & : \text{leave}(x)
\end{align*}
\]

\[
\begin{align*}
  u_0 & : \text{utt} - \text{event} \\
  a & : \text{spkr}(u_0) \\
  x & : \text{john} \\
  p & : \text{leave}(x) \\
  q & : \text{ask}(u_0, a, ?p)
\end{align*}
\]

\[
\begin{align*}
  \text{Ty}(e), \\
  & [x : \text{john}] \\
  \text{Ty}(e \rightarrow t), \\
  & \lambda [x]. [p : \text{leave}(x)]
\end{align*}
\]

- Similarities with [Ginzburg et al., 2003]
An example: a “clausal” clarification request

A: “Did John leave?”

\[ +Q, \begin{array}{l}
  u_0 : utt \rightarrow event \\
  a : spkr(u_0) \\
  x : john \\
  p : leave(x)
\end{array} \]

\[ \lambda [x]. [ p : leave(x) ] \]
An example: a “clausal” clarification request

A: “Did John leave?”

\[
\begin{align*}
\text{\quad A: } & \text{“Did John leave?”} \\
\text{\quad } & + Q, \\
\text{\quad } & \begin{bmatrix}
\text{\quad } & u_0 : utt - event \\
\text{\quad } & a : spkr(u_0) \\
\text{\quad } & x : john \\
\text{\quad } & p : leave(x)
\end{bmatrix} \\
\text{\quad } & \begin{bmatrix}
\text{\quad } & u_0 : utt - event \\
\text{\quad } & a : spkr(u_0) \\
\text{\quad } & x : john \\
\text{\quad } & p : leave(x) \\
\text{\quad } & q_0 : ask(u_0, a, \text{?}p)
\end{bmatrix}
\end{align*}
\]

\[
\begin{bmatrix}
\text{\quad } & x : john \\
\text{\quad } & \lambda [x]. [ p : leave(x) ]
\end{bmatrix}
\]
An example: a “clausal” clarification request

A: “Did John leave?”

\[ u_0 : \text{utt} - \text{event} \]
\[ a : \text{spkr}(u_0) \]
\[ x : \text{john} \]
\[ p : \text{leave}(x) \]

\[ +Q, \]
\[ x : \text{john} \quad \lambda [x]. [ p : \text{leave}(x) ] \]

B: “John?”

\[ u_1 : \text{utt} - \text{event} \]
\[ b : \text{spkr}(u_1) \]
\[ x : \text{john} \]
An example: a “clausal” clarification request

A: “Did John leave?”

\[ u_0 : \text{utt} - \text{event} \]
\[ a : spkr(u_0) \]
\[ x : john \]
\[ p : \text{leave}(x) \]

\[ +Q, \]

\[ \lambda [x]. [ p : \text{leave}(x) ] \]

B: “John?”

\[ u_1 : \text{utt} - \text{event} \]
\[ b : spkr(u_1) \]
\[ x : john \]
An example: a “constituent” clarification request

- Add [Poesio and Traum, 1997]’s *micro-conversational events*

  A: “Did John . . .”
  
  \[ ?Ty(t), +Q \]

  \[
  \begin{array}{l}
  u_{01} : \text{utt} - \text{event} \\
  a : spkr(u_{01}) \\
  x : john \\
  \end{array}
  \]

  \[ ?Ty(e \rightarrow t) \]
An example: a “constituent” clarification request

- Add [Poesio and Traum, 1997]'s micro-conversational events
  
  A: “Did John . . .”
  
  \[ ?Ty(t), +Q \]

  \[
  \begin{array}{l}
  u_{01} : \text{utt} - \text{event} \\
  a : spkr(u_{01}) \\
  x : john
  \end{array}
  \]

  \[ ?Ty(e \rightarrow t) \]

  B: “John?”

  \[
  \begin{array}{l}
  u_{1} : \text{utt} - \text{event} \\
  b : spkr(u_{1}) \\
  x : john
  \end{array}
  \]
An example: a “constituent” clarification request

Add [Poesio and Traum, 1997]’s *micro-conversational events*

A: “Did John . . .”

\[ ?Ty(t), +Q \]

\[
\begin{array}{l}
u_{01} : \text{utt} - \text{event} \\
a : \text{spkr}(u_{01}) \\
x : \text{john}
\end{array}
\]

\[ ?Ty(e \rightarrow t) \]

B: “John?”

\[
\begin{array}{l}
u_{01} : \text{utt} - \text{event} \\
\ldots : \ldots \\
u_{1} : \text{utt} - \text{event} \\
b : \text{spkr}(u_{1}) \\
x : \text{john}
\end{array}
\]

\[
\begin{array}{l}
u_{01} : \text{utt} - \text{event} \\
q_{1} : \text{ask}(u_{1}, b, ?\text{content}(u_{01}, a, x))
\end{array}
\]
An example: a clarificational split utterance

A: “John …”

\[
+ Q, \begin{bmatrix}
  u_0 & : & \text{utt} - \text{event} \\
  a & : & \text{spkr}(u_0) \\
  x & : & \text{john} \\
  P & : & \text{META}(x)
\end{bmatrix}
\]

\[
\begin{array}{c}
  x : \text{john} \\
  ?\text{Ty}(e \rightarrow t)
\end{array}
\]
An example: a clarificational split utterance

A: “John . . .”

\[ u_0 : \text{utt} - \text{event} \]
\[ a : \text{spkr}(u_0) \]
\[ x : \text{john} \]
\[ P : \text{META}(x) \]
\[ x : \text{john} \]
\[ ?Ty(e \to t) \]
An example: a clarificational split utterance

A: “John . . .”  B: “left?”

\[
\begin{align*}
+ Q, \\
\begin{array}{l}
u_0: \text{utt} \rightarrow \text{event} \\
a: \text{spkr}(u_0) \\
x: \text{john} \\
P: \text{META}(x)
\end{array}
\end{align*}
\]

\[
\begin{array}{l}
x: \text{john} \\
? \text{Ty}(e \rightarrow t)
\end{array}
\]

\[
\begin{align*}
u_0: \text{utt} \rightarrow \text{event} \\
a: \text{spkr}(u_0) \\
x: \text{john} \\
P: \text{META}(x) \\
q_0: \text{ask}(u_0, a, ?P)
\end{align*}
\]
An example: a clarificational split utterance

A: “John . . .” B: “left?”

\[
\begin{align*}
+ Q, \quad & u_0 : \text{utt} - \text{event} \\
& a : \text{spkr}(u_0) \\
& x : \text{john} \\
& p : \text{leave}(x)
\end{align*}
\]

\[
\begin{align*}
& u_0 : \text{utt} - \text{event} \\
& a : \text{spkr}(u_0) \\
& x : \text{john} \\
& p : \text{leave}(x) \\
q_0 : \text{ask}(u_0, a, ?p)
\end{align*}
\]
An example: a clarificational split utterance

A: “John . . .” B: “left?”

\[ u_0 : \text{utt} \rightarrow \text{event} \]

\[ a : \text{spkr}(u_0) \]

\[ x : \text{john} \]

\[ p : \text{leave}(x) \]

\[ x : \text{john} \]

\[ \lambda [x]. [ p : \text{leave}(x) ] \]

\[ u_0 : \text{utt} \rightarrow \text{event} \]

\[ a : \text{spkr}(u_0) \]

\[ x : \text{john} \]

\[ p : \text{leave}(x) \]

\[ q_0 : \text{ask}(u_0, a, ?p) \]

\[ u_0 : \text{utt} \rightarrow \text{event} \]

\[ \ldots : \ldots \]

\[ u_1 : \text{utt} \rightarrow \text{event} \]

\[ b : \text{spkr}(u_1) \]

\[ q_1 : \text{ask}(u_1, b, ?\text{ask}(u_0, a, p)) \]
(Eventual) Conclusions

- Incrementality of DS with the flexibility of TTR
- Core grammar essentially as before
- Optional enrichment processes for speech act information
  - similarities to [Ginzburg and Cooper, 2004] et al.
  - similarities to [Asher and Lascarides, 2003] et al.
- A proper treatment of split utterances . . . ?
  - capturing insights of [Rieser and Poesio, prep]
  - more fundamentally incremental
Thanks!

And thanks to:
Pat Healey, Christine Howes, Graham White, Arash Eshghi, Greg Mills at QMUL
Ruth Kempson, Eleni Gregoromichelaki, Wilfried Meyer-Viol at KCL
Andrew Gargett in Saarbrücken, Yo Sato in Herts
*Logics of Conversation*.  
Cambridge University Press.

Extension of martin l¨of type theory with record types and subtyping.  
In Sambin, G. and Smith, J., editors, *25 Years of Constructive Type Theory*. Oxford University Press.

Linguistics, logic and finite trees.  

Syntactic co-ordination in dialogue.  
*Cognition*, 75:13–25.
Records and record types in semantic theory.

Using dependent record types in clarification ellipsis.

*Non-Sentential Utterances in Dialogue: Classification, Resolution and Use.*
PhD thesis, Department of Computer Science, King’s College London, University of London.

Saying what you mean in dialogue: A study in conceptual and semantic co-ordination.

Abstraction and ontology: questions as propositional abstracts in type theory with records.

Ginzburg, J. (prep).
The Interactive Stance: Meaning for Conversation.
Studies in Computational Linguistics. CSLI Publications. Draft chapters available from:
http://www.dcs.kcl.ac.uk/staff/ginzburg.

Clarification, ellipsis, and the nature of contextual updates in dialogue.


Computational Intelligence, 13(3).


