Incrementality, Alignment and Split Utterances

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

March 11, 2010
Outline

1. Dialogue and Incrementality
   - Split Utterances and Alignment

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

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

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

1. Dialogue and Incrementality
   - Split Utterances and Alignment

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

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

4. Dynamic Syntax & Type Theory with Records (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]
  - inherently incremental grammar formalism
- 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]
  - inherently incremental grammar formalism
- Empirical studies using corpora and experiments
  - Non-sentential utterances
  - Clarification requests
  - Split utterances
  - Priming/alignment
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

- Treatment for one particular kind [Poesio and Rieser, 2010]
  - 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? Within constituents?

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

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

(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? Within constituents?

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

2. Hugh: Ruth visited Trecastle, to go to the
   Alex: farm shop
Split Utterances

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

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

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

- Splits can occur across syntactic/semantic dependencies:

(3) 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? Within constituents?

1. Hugh: Ruth visited
   Alex: Trecastle, to go to the farm shop
2. Hugh: Ruth visited Trecastle, to go to the
   Alex: farm shop
- Splits can occur across syntactic/semantic dependencies:

3. A: Have you read ...
   B: any of your chapters? Not yet.
- Not just a case of splitting a *string*
Split Utterances

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

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

2. Hugh: Ruth visited Trecastle, to go to the
   Alex: farm shop

- Splits can occur across syntactic/semantic dependencies:

3. A: Have you read ...
   B: any of your chapters? Not yet.

- Not just a case of splitting a string
- How common are these really?
- Where do splits really occur (how incremental must we be)?
Primed and/or Alignment

- Tendency to repeat previously used material
  - words
  - syntactic structures [Branigan et al., 2000]
  - multi-word expressions
  - ways of referring [Garrod and Anderson, 1987]
- Both self- and other- effects [Pickering and Ferreira, 2008]
- Interesting for models of incremental processing
  - (...especially in the case of split utterances ...)
  - what phenomena are primed/aligned? (and therefore represented)?
  - evidence for independence of lexicon/syntax/semantics?
- Most data from controlled experimental settings
- What does this tell us about real dialogue?
Outline

1. Dialogue and Incrementality
   - Split Utterances and Alignment

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

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

4. Dynamic Syntax & Type Theory with Records (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
DS Trees as semantic representations

- End product of parsing is a semantic tree
- Nodes decorated with $Ty()$ type and $Fo()$ formula labels
  
  "John likes Mary":

  $Ty(t),$
  $Fo(\text{like}(\text{john}, \text{mary}))$

  $Ty(e),$
  $Fo(\text{john})$

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

  $Ty(e),$
  $Fo(\text{mary})$

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

- 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

- Incremental tree growth driven by *requirements* e.g. $?Ty(t)$
- Node under development marked by *pointer* ♦
- Words induce sets of *lexical* actions: "like"

```plaintext
IF $?Ty(e \rightarrow t)$ THEN
make(⟨↓₁⟩); go(⟨↓₁⟩);
put($Fo(\lambda y \lambda x. like(x, y))$);
put($Ty(e \rightarrow (e \rightarrow t))$)
go(⟨↑₁⟩); make(⟨↓₀⟩);
go(⟨↓₀⟩); put($?Ty(e)$)
ELSE ABORT

<table>
<thead>
<tr>
<th>$?Ty(e)\rightarrow t$</th>
<th>$Ty(e \rightarrow (e \rightarrow t))$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Fo(\lambda y \lambda x. like(x, y))$</td>
<td></td>
</tr>
</tbody>
</table>

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

Processing *Someone fainted*

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

Processing *Someone fainted*

```
&Ty(t)

◊, ?Ty(e)       ?Ty(e → t)
```
Processing *Someone fainted*

```
?Ty(t)

◊, Ty(e)     ?Ty(e → t)
ε, x, person(x)
```
Processing *Someone fainted*

\[
\begin{array}{c}
\text{?Ty}(t) \\
\text{Ty}(e) \quad \text{?Ty}(e \to t), \Diamond \\
\epsilon, x, \text{person}(x)
\end{array}
\]
Unfolding then building up the tree

Processing *Someone fainted*

\[
\begin{align*}
\text{Processing } & \text{Someone fainted} \\
& \quad \begin{array}{c}
? Ty(t) \\
\quad Ty(e) \\
\quad Ty(e \rightarrow t), \diamond \\
\epsilon, x, \text{person}(x) \\
\quad \lambda y. \text{faint}(y)
\end{array}
\end{align*}
\]
Unfolding then building up the tree

Processing *Someone fainted*

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

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

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

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

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

\[ \text{faint}(\epsilon, x, \text{person}(x)) \]
\[ \text{Ty}(t) \]
\[ \text{Ty}(e) \quad \text{Ty}(e \rightarrow t) \]
\[ \epsilon, x, \text{person}(x) \quad \lambda y.\text{faint}(y) \]
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**

\[
\begin{align*}
Ty(t), & \\
\text{faint}(\epsilon, \text{person}(x)) & \\
\end{align*}
\]

**TEST TREE**

\[
\begin{align*}
?Ty(t), & \\
\end{align*}
\]

\[
\begin{align*}
Ty(e), & \\
\epsilon, x, \text{person}(x) & \\
\end{align*}
\]

\[
\begin{align*}
Ty(e \rightarrow t) & \\
\lambda y.\text{faint}(y) & \\
\end{align*}
\]
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**

\[
Ty(t), \diamond \\
\text{faint}(\varepsilon, \text{person}(x))
\]

\[
Ty(e), \quad Ty(e \rightarrow t) \\
\varepsilon, x, \text{person}(x), \quad \lambda y. \text{faint}(y)
\]

**TEST TREE**

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

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

**TEST TREE**

\[
\begin{align*}
& ?Ty(t), \diamond, Ty(e) \\
& ?Ty(e \rightarrow t), \epsilon, x, person(x) \\
\end{align*}
\]

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**

\[
Ty(t), \diamond
\]

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

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

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

**TEST TREE**

\[
?Ty(t),
\]

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

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

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**

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

**TEST TREE**

\[
\text{Gen: “Someone fainted”}
\]
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**

\[
Ty(t), \diamond
\]

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

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

**TEST TREE**

\[
Ty(t), \diamond
\]

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

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

Gen: “Someone fainted”
Underspecification: structural

- “Unfixed” nodes - building underspecified tree relations

```
?Ty(t), Tn(0)

Ty(e), Mary'  ?Ty(e)  ?Ty(e → t)
```

- 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:

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

```latex
\begin{array}{c}
\text{\small ?Ty}(t) \\
\hline
\text{\small U,} \\
\text{\small ?}\exists x \text{Fo}(x) \\
\hline
\text{\small \lambda y. faint}(y)
\end{array}
```

```
\begin{array}{c}
\text{\small SUBSTITUTION}
\end{array}
```
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.
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:

\[
\begin{align*}
\text{smoke}(john) \\
\text{john} & \quad \lambda y.\text{smoke}(y) \\
\text{bill} & \quad \exists x \text{Fo}(x)
\end{align*}
\]

SUBSTITUTION

Matthew Purver et al.  OU Computing Dept, 11/03/10
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:

$$\text{smoke}(\text{john})$$

$$\lambda y. \text{smoke}(y)$$

$$\text{substitution}$$

- 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.

  $\text{Smoke}'(\text{Bill}') \land \text{Faint}'(\text{Bill}')$

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

$\text{Bill}'$ $\text{Smoke}'$ $\text{Faint}'$
Appositions as **LINKed** 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*

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

\[\text{Ty}(e)\]
Appositions as \textit{LINKed} trees

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

- Partial tree as context with term enriched by \textit{LINKed} tree of same type
- Parsing \textit{A friend, a musician}

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

Evaluation of \textit{LINKed} nodes both of type \(e\) yields composite term:

\[
\epsilon, x, \text{Friend}'(x) \land \text{Musician}'(x)
\]

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

- DS seems well suited for split utterances
- Inherent word-by-word incrementality
- Well-defined partial structures at each point
- Same actions and partial structures in parsing and generation
- Grammatical constraints via semantics rather than “syntax”
  - Not licensing a string
  - Splits should be possible anywhere
DS and Split Utterances

- DS *seems* well suited for split utterances
- Inherent word-by-word incrementality
- Well-defined partial structures at each point
- Same actions and partial structures in parsing and generation
- Grammatical constraints via semantics rather than "syntax"
  - Not licensing a string
  - Splits should be possible anywhere

\[
\begin{align*}
\text{John} & \quad \text{Ty}(e \rightarrow t), \\
\text{?Ty}(t) & \quad \text{smoke}(\text{John}), \\
\lambda y.\text{smoke}(y)
\end{align*}
\]
DS and Split Utterances

- **DS seems** well suited for split utterances
- Inherent word-by-word incrementality
- Well-defined partial structures at each point
- Same actions and partial structures in parsing and generation
- Grammatical constraints via semantics rather than “syntax”
  - Not licensing a string
  - Splits should be possible anywhere
- Is it too general (what are the real constraints)?
- Is it too simplistic (what do split utterances *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 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])

```
IF ?Ty(e → t)
THEN make(⟨↓₁⟩); go(⟨↓₁⟩);
    put(Fo(λy λx.like(x, y)));
    put(Ty(e → (e → t)))
    go(⟨↑₁⟩); make(⟨↓₀⟩);
    go(⟨↓₀⟩); put(?Ty(e))
ELSE ABORT
```
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])
- Does this really explain general (non-lexical) effects?
  - Branigan et al found weaker cross-verb effects
  - (re-use of *computational* action sequences?)
- Do we see them in real dialogue?
Outline

1. Dialogue and Incrementality
   - Split Utterances and Alignment

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

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

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

- What do these phenomena really look like?
Empirical Investigations

- What do these phenomena really look like?
- What’s the deal with lexical and syntactic priming?
  - Do we see them in ordinary dialogue?
  - Can we tell which effect is greater?
Empirical Investigations

- What do these phenomena really look like?
- What’s the deal with lexical and syntactic priming?
  - Do we see them in ordinary dialogue?
  - Can we tell which effect is greater?
- Do split utterances really behave the way we think?
  - How common are they?
  - Where does the split happen?
  - What do they mean?
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
<table>
<thead>
<tr>
<th>A: Hello</th>
<th>A’: Hi</th>
</tr>
</thead>
<tbody>
<tr>
<td>B: Hi</td>
<td>B’: Hello</td>
</tr>
<tr>
<td>A: How are you?</td>
<td>A’: What’s up?</td>
</tr>
<tr>
<td>B: Fine - you?</td>
<td>B’: Not much</td>
</tr>
<tr>
<td>A: Yeah fine thanks</td>
<td>A’: Me neither</td>
</tr>
<tr>
<td>B: Uh-huh</td>
<td>B’: Uh-huh</td>
</tr>
</tbody>
</table>
Corpus experiment: Method

A: Hello

A: How are you?

A: Yeah fine thanks

B’: Hello

B’: Not much

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: Method

A: Hello
B': Uh-huh
A: How are you?
B': Hello
A: Yeah fine thanks
B': Not much
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 $S_{lex} = \frac{N_{AB}}{\sqrt{N_{AA}N_{BB}}}$

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

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

- Lexical similarity expressed via word pair kernel:
  - number of matching word pairs between turns $A$ and $B = N_{AB}$
  - similarity $S_{lex} = \frac{N_{AB}}{\sqrt{N_{AA}N_{BB}}}$
- Real dialogues mean other-person similarity
  $S_{lex} = 0.094 \ (SD = 0.04)$
- Control dialogues mean other-person similarity
  $S_{lex} = 0.059 \ (SD = 0.03)$
- ANOVA for real vs. control shows difference is reliable:
  $F(1,253) = 106.55, \ p = 0.00$
Corpus experiment: Syntactic results (1)

- 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 (1)

- 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 $S_{syn} = \frac{N_{AB}}{\sqrt{N_{AA} N_{BB}}}$

- 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 (1)

- 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 $S_{syn} = \frac{N_{AB}}{\sqrt{N_{AA}N_{BB}}}$
- Real dialogues mean other-person similarity
  $S_{syn} = 0.19 \ (SD = 0.06)$
- Control dialogues mean other-person similarity
  $S_{syn} = 0.18 \ (SD = 0.06)$
- ANOVA for real vs. control shows difference not reliable:
  $F_{(1,253)} = 1.32, \ p = 0.25$
- But: a reliable effect of genre ($F_{(2,237)} = 20.13, \ p = 0.00$):

<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>
Corpus experiment: Syntactic results (2)

- What’s the influence of lexical similarity on syntactic similarity?
- Linear Mixed Model analysis can tell us:
  - subject, dialogue as random factors
  - real/control type as fixed factor
  - lexical similarity as covariate
What’s the influence of lexical similarity on syntactic similarity?

- Linear Mixed Model analysis can tell us:
  - subject, dialogue as random factors
  - real/control type as fixed factor
  - lexical similarity as covariate

- Parameter estimate for $S_{syn}$ negative

- Marginal (“corrected”) means:
  - $S_{syn} = 0.184$ real, $S_{syn} = 0.211$ control

- Reliable difference: $p = 0.01$

- i.e. $S_{syn}$ is lower than chance when $S_{lex}$ taken into account
What’s the influence of lexical similarity on syntactic similarity?

Linear Mixed Model analysis can tell us:
- subject, dialogue as random factors
- real/control type as fixed factor
- lexical similarity as covariate

Parameter estimate for $S_{syn}$ negative

Marginal ("corrected") means:
$S_{syn} = 0.184$ real, $S_{syn} = 0.211$ control

Reliable difference: $p = 0.01$

i.e. $S_{syn}$ is lower than chance when $S_{lex}$ taken into account

Checked on BNC spoken portion (bigger but not parsed)
- parsed using C&C CCG parser, Stanford CFG parser
- results the same
Corpus experiment: Conclusions

- We can measure the effect of lexical priming
- We can’t measure the effect of syntactic priming
  - It appears to be negative when lexical effect taken into account
  - Even if it exists, it must be 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
Corpus experiment: Conclusions

- We can measure the effect of lexical priming
- We can’t measure the effect of syntactic priming
  - It appears to be negative when lexical effect taken into account
  - Even if it exists, it must be 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
- A grammar which associates syntactic effects with lexical entries might be on the right track . . .
- We’d like to know more about individual phenomena . . .
Split Utterances: 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, Szczepek, 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
  - 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

  CONTINUES

- A1: in getting to know
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

\[ \text{START-COMPLETE} = \text{N} \quad \text{END-COMPLETE} = \text{Y} \]
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)
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
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:
  1. F: We are going to call you the
     U: Wallering
  2. 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

- 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
  2. S: N?
  3. J: N for name, it’ll let you type in the document name.

- The antecedent for clarification is often incomplete . . .
  - (hard to establish propositional content/intention of antecedent)
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

...
Corpus Study: Conclusions

- Some conclusions play right into DS’s hands . . .
  - Splits happen within syntactic/semantic “constituents”
  - Not always collaborative as per [Poesio and Rieser, 2010]
  - Intervening turns use incomplete antecedents (partial trees)
Some conclusions play right into DS’s hands . . .
- Splits happen within syntactic/semantic “constituents”
- Not always collaborative as per [Poesio and Rieser, 2010]
- Intervening turns use incomplete antecedents (partial trees)

. . . but some don’t:
- Repair
- Clarifications
Corpora tell us nothing about processing questions
- SUs may be common, but are they easy/hard to process?

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 (DS)
Empirical Investigations
Dynamic Syntax & Type Theory with Records (TTR)

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

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

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

turn typed by sam intercepted by server
First part of SU relayed to E ...
Dialogue and Incrementality
Dynamic Syntax (DS)
Empirical Investigations
Dynamic Syntax & Type Theory with Records (TTR)

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

Matthew Purver et al.
OU Computing Dept, 11/03/10

... and cyn
Dialogue and Incrementality
Dynamic Syntax (DS)
Empirical Investigations
Dynamic Syntax & Type Theory with Records (TTR)

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
  Bancil: 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’t observe time to start typing)
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’t observe time to start typing)
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 *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

- 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)
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 . . .
  - . . . and we’d really like to know about onset delay . . .
  - . . . a character-by-character version is almost complete
Outline

1. Dialogue and Incrementality
   - Split Utterances and Alignment

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

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

4. Dynamic Syntax & Type Theory with Records (TTR)
   - Adding TTR to DS
   - Fragments & Split Utterances in DS/TTR
So far, we’re happy that we’re going in roughly the right direction:

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

For a proper treatment of NSUs and SUs, DS needs:
- Utterance function (speech acts?)
- Responsibility for a (sub-)utterance (speaker, hearer?)
- So we need more structured representations

Want to avoid *forcing* this into all representations . . .
- What should really be in the grammar?
Type Theory With Records

- 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:
    \[
    v_1 : T_1, \quad v_2 : T_2, \quad v_3 : T_3
    \]
Type Theory With Records

- Types can be dependent on earlier (higher-up) types:
  \[
  \begin{array}{l}
  l_1 : T_1 \\
  l_2 : T_2(l_1) \\
  l_3 : T_3(l_1, l_2)
  \end{array}
  \]

- We can have nested records and record types:
  \[
  \begin{array}{l}
  l_1 : T_1 \\
  l_2 : \begin{array}{l}
  l'_1 : T'_1 \\
  l'_2 : T'_2
  \end{array} \\
  l_3 : T_3(l_1, l_2.l'_1, l_2.l'_2)
  \end{array}
  \]

- We can have functional record types:
  \[
  \lambda r : \begin{array}{l}
  l_1 : T_1 \\
  l_2 : T_2
  \end{array} (\begin{array}{l}
  l_3 : T_3 \\
  l_4 : T_4(r.l_1, r.l_2)
  \end{array})
  \]
Type Theory With Records

- Used for sentential semantics, e.g. [Cooper, 2005]
  - “A man left”: \[ x : \text{man} \\
                             p : \text{leave}(x) \]
  - for truth: \( x \) must be a man, \( p \) a proof that \( x \) left

- “Every man left”: \( \lambda r : [ x : \text{man} ] ([ p : \text{leave}(r.x) ] ) \)

- Similarities to DRT representation:
  
<table>
<thead>
<tr>
<th>( x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{man}(x) )</td>
</tr>
<tr>
<td>( \text{leave}(x) )</td>
</tr>
</tbody>
</table>

- 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?

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

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

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

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

\[ \diamond, \left[ \begin{align*}
x & : \text{john} \\
e & : \text{leave}(x)
\end{align*} \right] \]

\[ \left[ \begin{align*}
x & : \text{john} \\
\lambda [x].\left[ \begin{align*}
p & : \text{leave}(x) \end{align*} \right]
\end{align*} \right] \]
A simple version

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

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

$$\Diamond, \ ?Ty(t)$$

- $Ty(e)$, $john$
- $Ty(e \rightarrow t)$, $\lambda x.\ leave(x)$

IF $\ ?Ty(e)$
THEN put($Ty(e)$)
put($Fo(john)$)
ELSE abort
A simple version

- Replace $\text{Fo()}$ epsilon-calculus labels with TTR record types

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

IF $?Ty(e)$
THEN put($Ty(e)$)
ELSE put($[x : john]$)
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

```
◊, ?Ty(t)

Ty(e),
[ x : john ]

Ty(e → t),
λx. leave(x)
```

```
IF ?Ty(e)
THEN put(Ty(e))
put([ x : john ])
ELSE abort
```
A simple version

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

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

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

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

\[
\lambda [ x : e ] . \begin{bmatrix} x : e \\ p : leave(x) \end{bmatrix}
\]
Adding in LINK relations

- For LINKed trees, we need conjunction

"Bill, who fainted, smokes."

\[
\begin{align*}
\text{smoke(bill)} & \land \text{faint(bill)} \\
\text{bill} & \quad \lambda x.\text{smoke}(x) \\
\text{faint(bill)} & \\
\text{bill} & \quad \lambda x.\text{faint}(x)
\end{align*}
\]
Adding in LINK relations

- For LINKed trees, we need conjunction
- Use extension: \( \oplus \) where \( r_1 \oplus r_2 \) adds \( r_2 \) to the end of \( r_1 \)
  - (for distinct labels; identical fields collapse [Cooper, 1998])

“Bill, who fainted, smokes.”

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

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

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

\[
\text{smoke}(x) \quad \lambda x.\text{smoke}(x)
\]
For LINKed trees, we need conjunction

Use extension: $r_1 \oplus r_2$ adds $r_2$ to the end of $r_1$

(for distinct labels; identical fields collapse [Cooper, 1998])

“Bill, who fainted, smokes.”

\[
\begin{align*}
\text{smoke(bill)} \land \text{faint(bill)} & \quad \left[ \begin{array}{l}
p : \text{smoke(bill)} \\
q : \text{faint(bill)}
\end{array} \right] \\
\text{faint(bill)} & \quad \left[ \begin{array}{l}
x : \text{bill} \\
\lambda [x]. [p : \text{smoke(x)}]
\end{array} \right] \\
\text{bill} \lambda x. \text{smoke}(x) & \quad \left[ \begin{array}{l}
x : \text{bill} \\
\lambda [x]. [q : \text{faint(x)}]
\end{array} \right] \\
\text{faint(bill)} & \quad \left[ \begin{array}{l}
x : \text{bill} \\
\lambda [x]. [q : \text{faint(x)}]
\end{array} \right]
\end{align*}
\]
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”

\[
\begin{align*}
\Diamond, Ty(t), & \\
\left[ x : john \right], & \\
\left[ p : leave(x) \right], & \\
\end{align*}
\]

\[
\begin{align*}
Ty(e), & \\
\left[ x : john \right], & \\
\lambda [x]. \left[ p : leave(x) \right], & \\
\end{align*}
\]
LINK as optional enrichment process

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

“John left”

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

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

\text{Ty}(e \rightarrow t), \begin{cases}
  u_0 : \text{utt} - \text{event} \\
  a : \text{spkr}(u_0) \\
  x : \text{john} \\
  p : \text{leave}(x)
\end{cases}\]
LINK as optional enrichment process

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

“John left”

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

\[
Ty(e), \\
\begin{bmatrix}
  x & : & john
\end{bmatrix}
\]

\[
\lambda [x]. [p : leave(x)]
\]

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

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

“John left”

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

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

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

- Speech act inferences conditional on syntax/semantics

"Did John leave?"

\[ \text{TTR}(e) \quad \text{Ty}(e \to t), \quad \text{Ty}(e), \quad +Q, \text{Ty}(t), \quad \Diamond \]

\[
\begin{array}{l}
\left[ \begin{array}{l}
u_0 : \text{utt} \to \text{event} \\
a : \text{spkr}(u_0) \\
x : \text{john} \\
p : \text{leave}(x)
\end{array} \right]
\end{array}
\]

\[
\text{TTR}(e), \quad \left[ \begin{array}{l}
x : \text{john}
\end{array} \right]
\]

\[
\lambda [x]. \left[ \begin{array}{l}
p : \text{leave}(x)
\end{array} \right]
\]
LINK as optional enrichment process

- Speech act inferences conditional on syntax/semantics

“Did John leave?”

\[ \diamond, + Q, Ty(t), \]
\[ u_0 : \text{utt} \rightarrow \text{event} \]
\[ a : \text{spkr}(u_0) \]
\[ x : \text{john} \]
\[ p : \text{leave}(x) \]

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

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

\[ u_0 : \text{utt} \rightarrow \text{event} \]
\[ a : \text{spkr}(u_0) \]
\[ x : \text{john} \]
\[ p : \text{leave}(x) \]
\[ q : \text{ask}(u_0, a, ?p) \]
**LINK as optional enrichment process**

- Speech act inferences conditional on syntax/semantics

\[
\text{“Did John leave?”} \quad \begin{aligned}
\diamond, +Q, Ty(t), \quad &\begin{cases}
u_0 &\text{: utt → event} \\
a &\text{: spkr}(u_0) \\
x &\text{: john} \\
p &\text{: leave}(x)
\end{cases} \\
&\begin{cases}
u_0 &\text{: utt → event} \\
a &\text{: spkr}(u_0) \\
x &\text{: john} \\
p &\text{: leave}(x) \\
q &\text{: ask}(u_0, a, ?p)
\end{cases}
\end{aligned}
\]

\[
\begin{aligned}
&\begin{array}{c}
Ty(e), \\
\lambda[x].[p : \text{leave}(x)]
\end{array} \quad Ty(e \rightarrow t),
\end{aligned}
\]

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

A: “Did John leave?”

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

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

A: “Did John leave?”

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

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

A: “Did John leave?”

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

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?”

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

B: “John?”

\[
\begin{align*}
&u_1 : \text{utt} \rightarrow \text{event} \\
&b : \text{spkr}(u_1) \\
&x : \text{john}
\end{align*}
\]
An example: a “constituent” clarification request

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

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

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

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

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

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

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

B: “John?”

\[
\begin{align*}
u_1 &: \text{utt} - \text{event} \\
b &: \text{spkr}(u_1) \\
x &: \text{john}
\end{align*}
\]
An example: a "constituent" clarification request

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

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

B: “John?”

\[
\begin{align*}
  u_{1} & : \text{utt} - \text{event} \\
  b & : \text{spkr}(u_{1}) \\
  x & : \text{john}
\end{align*}
\]

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

A: “John . . .”

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

\[ \text{?Ty}(e \rightarrow t) \]
An example: a clarificational split utterance

A: “John . . . ”

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

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

A: “John . . .”  B: “left?”

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

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

A: “John . . .”  B: “left?”

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

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

A: “John . . .”  B: “left?”

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

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

\[ u_0 : \text{utt} - \text{event} \]
\[ a : \text{spkr}(u_0) \]
\[ x : \text{john} \]
\[ p : \text{leave}(x) \]
\[ q_0 : \text{assert}(u_0, a, p) \]

\[ u_0 : \text{utt} - \text{event} \]
\[ \ldots : \ldots \]
\[ u_1 : \text{utt} - \text{event} \]
\[ b : \text{spkr}(u_1) \]
\[ q_1 : \text{ask}(u_1, b, ?\text{assert}(u_0, a, p)) \]
(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 [Poesio and Rieser, 2010]
  - 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, Ronnie Cann in Edinburgh, 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.
Information states, attitudes and dependent record types.

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, King’s College London, University of London.
Saying what you mean in dialogue: A study in conceptual and semantic co-ordination.

_The Interactive Stance: Meaning for Conversation._
Draft chapters available from:
http://www.dcs.kcl.ac.uk/staff/ginzburg.

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

Integrating conversational move types in the grammar of conversation.


Dynamic Syntax: The Flow of Language Understanding.
Blackwell.

Collaborative turn sequences.

Making tree kernels practical for natural language learning.

Structural priming: A critical review.

Completions, coordination, and alignment in dialogue.
Dialogue and Discourse, 1:1–89.

Conversational actions and discourse situations.
Computational Intelligence, 13(3).

Dialogue systems as proof editors.

Priming of syntactic rules in task-oriented dialogue and spontaneous conversation.

Conversation in context: a corpus-driven approach.
Continuum.


Interaction and Linguistic Structures (InLiSt),
http://www.uni-potsdam.de/u/inlist/issues/17/.