Inferring Meaning From Disfluencies in an Incremental Dialogue Framework

Julian Hough and David Schlangen

Dialogue Systems Group, Bielefeld University

IMC Workshop, IWCS 2015, Queen Mary University of London

Universität Bielefeld

1. Problem statement
2. Incremental dialogue framework: IU, DS-TTR and DyLan
3. Parsing disfluency
4. Probability and Order-theoretic Semantics
5. Interpreting disfluency in a dialogue system

1. Problem statement
2. Incremental dialogue framework: IU, DS-TTR and DyLan
3. Parsing disfluency
4. Probability and Order-theoretic Semantics
5. Interpreting disfluency in a dialogue system

Disfluency and other ‘dirty’ stuff

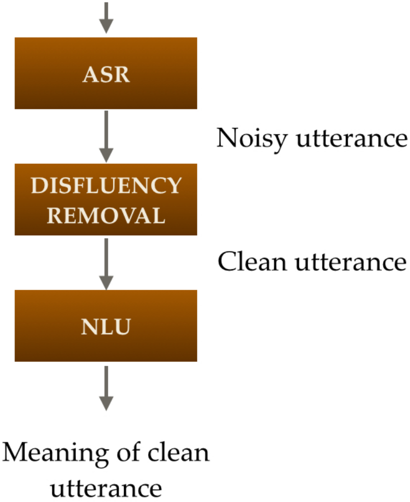
- Real dialogue is full of things like:
  - Filled pauses
  - Fillers (discourse markers, edit terms)
  - Self-repairs
  - Unfilled pauses (i.e. mid-turn silence/hesitation)
  - Laughter
  - Laughed speech
  - Exclamations (oh!, damn!, s***!)

Disfluency and other ‘dirty’ stuff

- Real dialogue is full of things like:
  - Filled pauses
  - Fillers (discourse markers, edit terms)
  - Self-repairs
  - Unfilled pauses (i.e. mid-turn silence/hesitation)
  - Laughter
  - Laughed speech
  - Exclamations (oh!, damn!, s***!)
- Are these problems, or *solutions*? [Clark, 1996]
- What do they *mean* in dialogue?

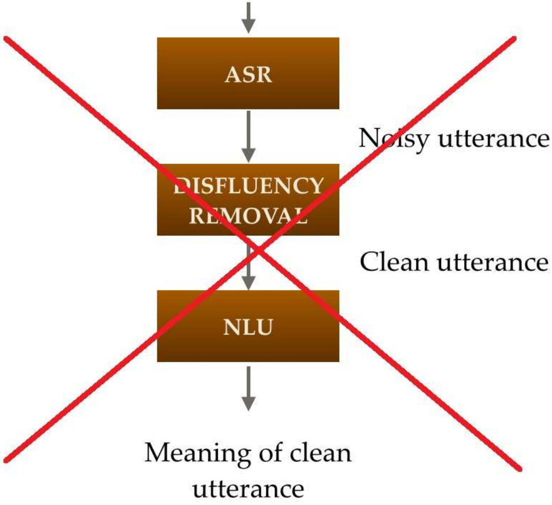
Disfluency processing: why do we care?

Dialogue systems (parsing speech)



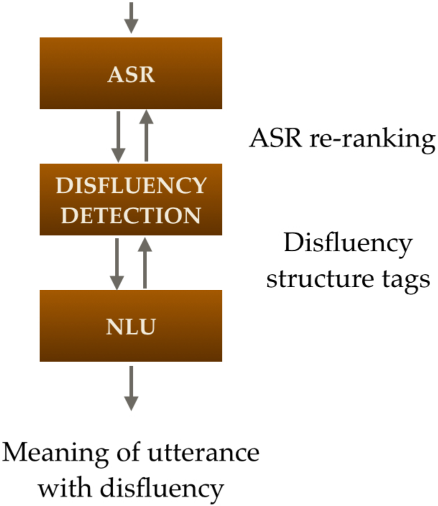
Disfluency processing: why do we care?

Dialogue systems (parsing speech)



Disfluency processing: why do we care?

Dialogue systems (parsing speech)



Self-repairs

"But one of **the, the** two things that I'm really..."

"Our situation is **just a little bit, kind of the** opposite of that"

"and you know it's like **you're, I mean,** employments are contractual by nature anyway"

[Switchboard examples]

Self-repairs: Annotation scheme

$$\text{John} \quad [\ \underbrace{\text{likes}}_{\text{reparandum}} \ + \ \underbrace{\{\text{uh}\}}_{\text{interregnum}} \ \underbrace{\text{loves}}_{\text{repair}}\ ]\ \text{Mary}$$

[Shriberg, 1994, onwards]

Terminology: *edit terms*, *interruption point* (+), *repair onset*

Self-repairs: classes

"But one of [the, + the] two things that I'm really..."

[repeat]

"Our situation is just [a little bit, + kind of the opposite] of that"

[substitution]

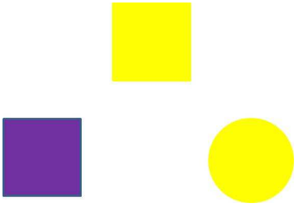
"and you know it's like [you're + {I mean}] employments are contractual by nature anyway"

[delete]

[Switchboard examples]

A familiar psycholinguistic experiment

- [Brennan and Schober, 2001]
  'Pick the yell-, uhh, purple square'



A familiar psycholinguistic experiment

- [Brennan and Schober, 2001] subjects use the reparandum and the presence of fillers to help make faster reference decisions:

  "Pick the, uh, purple square" *faster than fluent, no less accurate.*

  "Pick the yell-, uh, purple square" *faster than fluent, no less accurate.*

A familiar psycholinguistic experiment

- [Brennan and Schober, 2001] subjects use the reparandum and the presence of fillers to help make faster reference decisions:

  "Pick the, uh, purple square" *faster than fluent, no less accurate.*

  "Pick the yell-, uh, purple square" *faster than fluent, no less accurate.*

- "Pick the purple square, no, yellow" [Levelt, 1989, Ginzburg et al., 2014] *elliptical interpretation*

SUMMARY: what needs to be addressed...

- **Interpretation**: lack of fully incremental processing account of repairs. Deletion/ignoring of reparandum in self-repairs in automatic approaches. Lacks interface to discourse model.

SUMMARY: what needs to be addressed...

- **Interpretation**: lack of fully incremental processing account of repairs. Deletion/ignoring of reparandum in self-repairs in automatic approaches. Lacks interface to discourse model.
- **Generation**: lack of full integration with dialogue manager (incremental access to representations and discourse model)- needs inter-changeability with parsing.

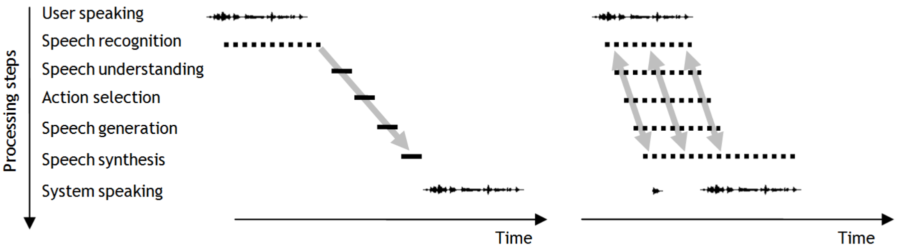
SUMMARY: what needs to be addressed...

- **Interpretation**: lack of fully incremental processing account of repairs. Deletion/ignoring of reparandum in self-repairs in automatic approaches. Lacks interface to discourse model.
- **Generation**: lack of full integration with dialogue manager (incremental access to representations and discourse model)- needs inter-changeability with parsing.
- **Dialogue models/dialogue management**: a nice model of forward and backward looking disfluency [Ginzburg et al., 2014], but lack of integration with incremental semantic grammars, parsers and generators.

  Needs probabilistic information to model realistic dialogue situations (*relevance*)

Incrementality (1 kind)

Non-incremental vs. Incremental Dialogue Systems



[Schlangen and Skantze, 2011]

DyLan Incremental stuff at all levels

Dynamic Syntax + TTR + IU Framework/Jindigo

[Purver et al., 2011]

- An incremental grammar formalism
  - *Dynamic Syntax* [Kempson et al., 2001]
- Interface between incremental representations and domain semantics
  - *Type Theory with Records (TTR)* [Cooper, 2005]
- An incremental dialogue framework which can store procedural context
  - *Incremental Unit (IU)* framework [Schlangen and Skantze, 2009]

DyLan Incremental stuff at all levels

Dynamic Syntax + TTR + IU Framework/Jindigo + Lattice and probability theory

[Purver et al., 2011]

- An incremental grammar formalism
  - *Dynamic Syntax* [Kempson et al., 2001]
- Interface between incremental representations and domain semantics
  - *Type Theory with Records (TTR)* [Cooper, 2005]
- An incremental dialogue framework which can store procedural context
  - *Incremental Unit (IU)* framework [Schlangen and Skantze, 2009]
- A dialogue model providing likelihood and relevance measures
  - *Lattice theory inquiry calculus* [Knuth, 2005] and Probabilistic TTR [Cooper et al., 2014]

TTR

$s : T$

where s can be a record and T can be a record type [Cooper, 2005] with fields of type judgements

- RTs are *inhabited* or *witnessed* by records

TTR

$$R_1 = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \\ l_3 & : & T_3(l_1) \end{bmatrix} \quad R_2 = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_{2'} \end{bmatrix} \quad R_3 = []$$

Figure : Example TTR record types

TTR

$$S_1 = \begin{bmatrix} l_1 & = & a \\ l_2 & = & b \\ l_3 & = & c \end{bmatrix} \quad S_2 = \begin{bmatrix} l_1 & = & a \\ l_2 & = & b' \end{bmatrix} \quad S_3 = []$$

Figure : Example TTR records

Record type check:
For a record s and and record type R, $s : R$ is true iff for every field $\left[\ l\ :\ T\ \right]$ in R there is a field $\left[\ l\ =\ v\ \right]$ in s such that $v : T$.

Subtype relation check:
For record types R_1 and R_2, $R_1 \sqsubseteq R_2$ is true iff for each field $\left[\ l\ :\ T_2\ \right]$ in R_2 there is a field $\left[\ l\ :\ T_1\ \right]$ in R_1 such that $T_1 \sqsubseteq T_2$. The $\sqsubseteq$ relation is reflexive and transitive.

DS-TTR

- Recent DS variant uses TTR *record types* on the trees [Purver et al., 2011].
- Record type compilation for *partial trees* [Hough, 2011] allows strong incremental interpretation [Milward, 1991].
- Incrementally constructed structures can be compared to domain concepts in word-by-word *subtype* relation checking.
- In generation, a goal tree in DS generation [Purver and Kempson, 2004] can be a TTR *goal concept* (record type) [Hough, 2011]- less tied to DS, interface with dialogue state possible.

Parsing *Robin arrives*:

Incremental DS-TTR parsing

Parsing *Robin arrives*:

$$\begin{bmatrix} x & : & e \\ p & : & t \end{bmatrix}$$

Incremental DS-TTR parsing

Parsing *Robin arrives*:

Robin

$$\begin{bmatrix} x_{=robin} & : & e \\ p & : & t \end{bmatrix}$$

Incremental DS-TTR parsing

Parsing *Robin arrives*:

Robin arrives

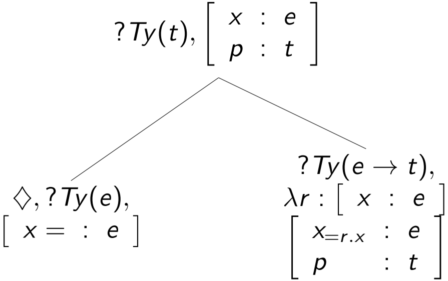
$$\left[\begin{array}{lll} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{array}\right]$$

Incremental DS-TTR parsing

Parsing *Robin arrives*:

Incremental DS-TTR parsing

Parsing *Robin arrives*:



Incremental DS-TTR parsing

Parsing *Robin arrives*:

Robin

$$?Ty(t), \left[\begin{array}{lcl} x_{=robin} & : & e \\ p & : & t \end{array}\right]$$

$$\Diamond, Ty(e), \left[\begin{array}{lcl} x =_{robin} & : & e \end{array}\right]$$

$$?Ty(e \rightarrow t), \lambda r : \left[\begin{array}{lcl} x & : & e \end{array}\right] \left[\begin{array}{lcl} x_{=r.x} & : & e \\ p & : & t \end{array}\right]$$

Incremental DS-TTR parsing

Parsing *Robin arrives*:

Robin

$$?Ty(t), \left[\begin{array}{lcl} x_{=robin} & : & e \\ p & : & t \end{array}\right]$$

$$Ty(e), \left[\begin{array}{lcl} x =_{robin} & : & e \end{array}\right]$$

$$\diamondsuit, ?Ty(e \rightarrow t), \lambda r : \left[\begin{array}{lcl} x & : & e \end{array}\right] \left[\begin{array}{lcl} x_{=r.x} & : & e \\ p & : & t \end{array}\right]$$

Parsing *Robin arrives*:

Robin arrives

$$\Diamond, Ty(t), \begin{bmatrix} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{bmatrix}$$

$$Ty(e), \begin{bmatrix} x =_{robin} & : & e \end{bmatrix}$$

$$Ty(e \rightarrow t), \lambda r : \begin{bmatrix} x & : & e \end{bmatrix} \begin{bmatrix} x_{=r.x} & : & e \\ p_{=arrive(x)} & : & t \end{bmatrix}$$

Incremental DS-TTR generation

Generating *Robin arrives*:

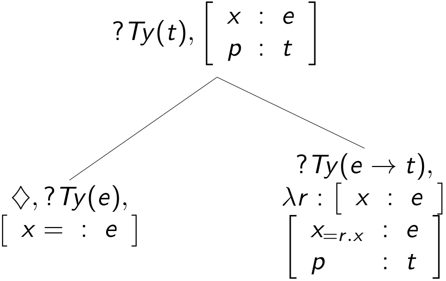
Incremental DS-TTR generation

Generating *Robin arrives*:

GOAL :

$$\begin{bmatrix} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{bmatrix}$$

SUBTYPE



Incremental DS-TTR generation

Generating *Robin arrives*:

Robin

GOAL :

$$\begin{bmatrix} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{bmatrix}$$

SUBTYPE

$$?Ty(t), \begin{bmatrix} x_{=robin} & : & e \\ p & : & t \end{bmatrix}$$

$$\Diamond, Ty(e), \begin{bmatrix} x =_{robin} & : & e \end{bmatrix}$$

$$?Ty(e \rightarrow t), \lambda r : \begin{bmatrix} x & : & e \end{bmatrix} \begin{bmatrix} x_{=r.x} & : & e \\ p & : & t \end{bmatrix}$$

Incremental DS-TTR generation

Generating *Robin arrives*:

Robin

GOAL :

$$\begin{bmatrix} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{bmatrix}$$

SUBTYPE

$$?Ty(t), \begin{bmatrix} x_{=robin} & : & e \\ p & : & t \end{bmatrix}$$

$$Ty(e), \begin{bmatrix} x =_{robin} & : & e \end{bmatrix}$$

$$\diamondsuit, ?Ty(e \rightarrow t), \lambda r : \begin{bmatrix} x & : & e \end{bmatrix} \begin{bmatrix} x_{=r.x} & : & e \\ p & : & t \end{bmatrix}$$

Incremental DS-TTR generation

Generating *Robin arrives*:

Robin arrives

GOAL :

$$\left[\begin{array}{lcl} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{array}\right]$$

MATCHES!

$$\Diamond, Ty(t), \left[\begin{array}{lcl} x_{=robin} & : & e \\ p_{=arrive(x)} & : & t \end{array}\right]$$

$$Ty(e), \left[\begin{array}{lcl} x =_{robin} & : & e \end{array}\right]$$

$$Ty(e \rightarrow t), \lambda r : \left[\begin{array}{lcl} x & : & e \end{array}\right] \left[\begin{array}{lcl} x_{=r.x} & : & e \\ p_{=arrive(x)} & : & t \end{array}\right]$$

DyLan dialogue system

- `DyLan` NLU [Purver et al., 2011] and NLG [Hough, 2011] modules in Jindigo [Skantze and Hjalmarsson, 2010], based on the IU framework [Schlangen and Skantze, 2009]
- Uses the graph-based input and output buffers.
- Uses a DS-TTR parsing DAG, shared by generation and parsing
- The notions of *GroundedIn* links to IUs in different modules, can *add*, *commit*, and *revoke* IUs.

DyLan dialogue system

- DyLan NLU [Purver et al., 2011] and NLG [Hough, 2011] modules in Jindigo [Skantze and Hjalmarsson, 2010], based on the IU framework [Schlangen and Skantze, 2009]
- Uses the graph-based input and output buffers.
- Uses a DS-TTR parsing DAG, shared by generation and parsing
- The notions of *GroundedIn* links to IUs in different modules, can *add*, *commit*, and *revoke* IUs. Gives us the requisite *incremental representation* for any given substring ('repairables').

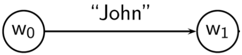
DyLan dialogue system

- **NLU module**:
  - *Input IUs:* Word graph from ASR
  - *Processing:* Increments a DS-TTR parsing DAG, GroundedIn corresponding word IUs
  - *Output IUs:* TTR record types (concepts) to dialogue manager, GroundedIn corresponding IUs of the DS-TTR DAG

DyLan parsing

John

WORD GRAPH
(INPUT)

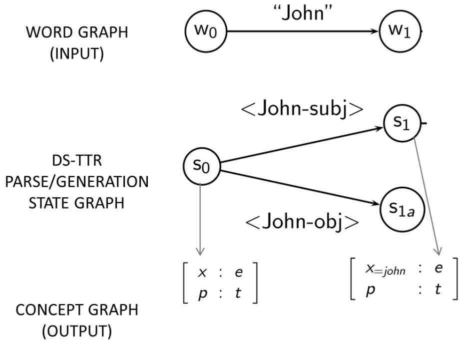


DS-TTR
PARSE/GENERATION
STATE GRAPH

CONCEPT GRAPH
(OUTPUT)

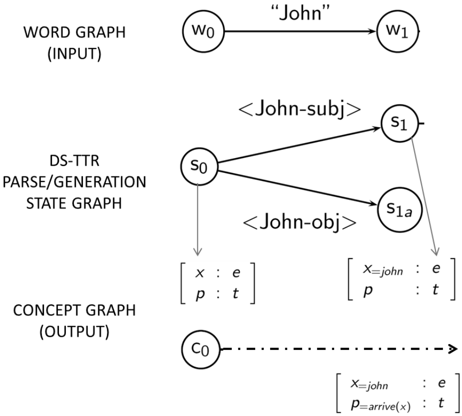
DyLan parsing

John



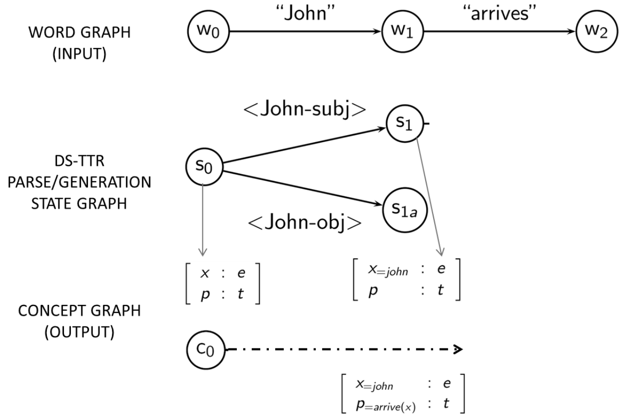
DyLan parsing

John



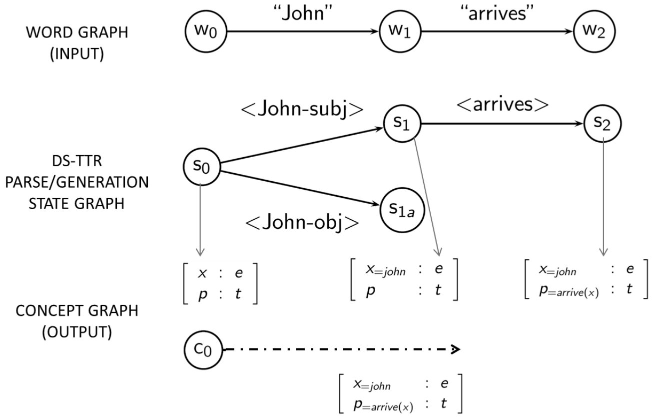
DyLan parsing

John arrives



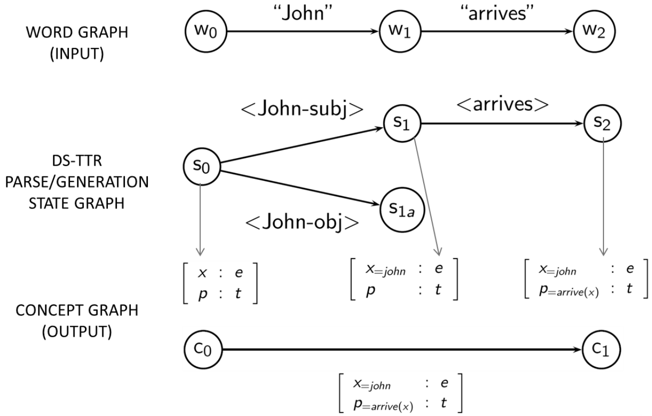
DyLan parsing

John arrives



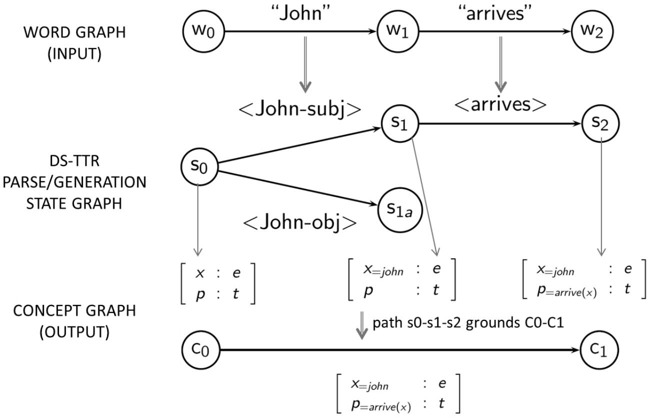
DyLan parsing

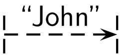
John arrives

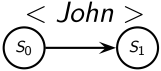


DyLan parsing

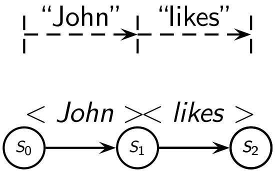
John arrives



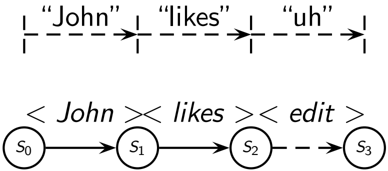




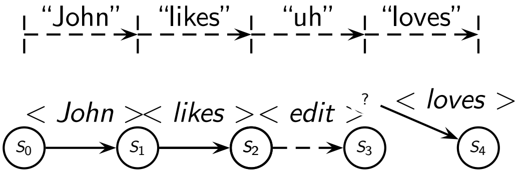
$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1 & : & e \\ x_{=John} & : & e \\ e & : & e_s \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & [Assert(User, \text{cont})] \end{array}\right]$$



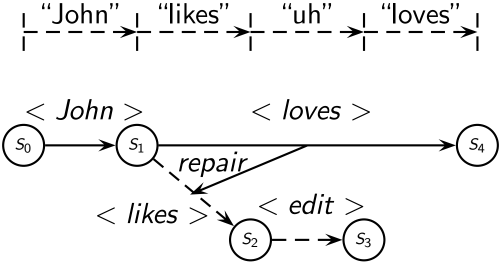
$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1 & : & e \\ x_{=John} & : & e \\ e_{=likes} & : & e_s \\ p1_{=obj(e,x1)} & : & t \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & [Assert(User, \text{cont})] \end{array}\right]$$



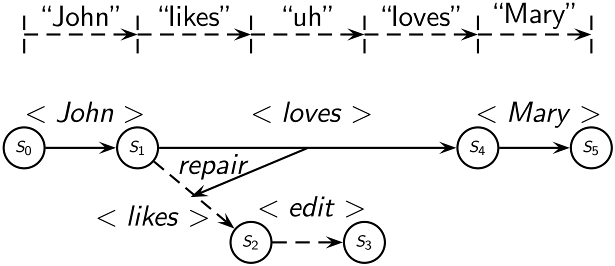
$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1 & : & e \\ x_{=John} & : & e \\ e_{=likes} & : & e_s \\ p1_{=obj(e,x1)} & : & t \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & \begin{array}{l} [Assert(User, \text{cont}), \\ FwdProblem(User, \text{cont})] \end{array} \end{array}\right]$$



$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1 & : & e \\ x_{=John} & : & e \\ e_{=likes} & : & e_s \\ p1_{=obj(e,x1)} & : & t \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & \begin{array}{l} [Assert(User, \text{cont}), \\ FwdProblem(User, \text{cont})] \end{array} \end{array}\right]$$



$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1 & : & e \\ x_{=John} & : & e \\ e_{=loves} & : & e_s \\ p_{=obj(e,x1)} & : & t \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & \begin{array}{l} [Assert(User,\text{cont}), \\ Revoke(User,[\, e_{=likes} \ : \ e_s \,] \\ \wedge\neg[\, e_{=loves} \ : \ e_s \,])] \end{array} \end{array}\right]$$



$$\left[\begin{array}{lcl} \text{cont} & = & \left[\begin{array}{lcl} x1_{=Mary} & : & e \\ x_{=John} & : & e \\ e_{=loves} & : & e_s \\ p_{=obj(e,x1)} & : & t \\ p_{=subj(e,x)} & : & t \end{array}\right] \\ \text{ctxt} & = & \begin{array}{l} [Assert(User,\text{cont}), \\ Revoke(User,[\,e_{=likes} : e_s\,] \\ \wedge\neg[\,e_{=loves} : e_s\,])] \end{array} \end{array}\right]$$

Model: Where we're up to

- We have strong incremental interpretation and incremental representation [Milward, 1991] of repairs and edit terms
- Models forward looking and backward looking disfluency [Ginzburg et al., 2014]
- We have bi-directional parsing and generation model *But* . . .

Model: Where we're up to

- We have strong incremental interpretation and incremental representation [Milward, 1991] of repairs and edit terms
- Models forward looking and backward looking disfluency [Ginzburg et al., 2014]
- We have bi-directional parsing and generation model *But* ...

- No reasoning/discourse system- just simple matching of domain concept RTs
- We have the 'closed-world' view that parseablility is {false,true}

Model: Where we're up to

- We have strong incremental interpretation and incremental representation [Milward, 1991] of repairs and edit terms
- Models forward looking and backward looking disfluency [Ginzburg et al., 2014]
- We have bi-directional parsing and generation model *But* . . .

- No reasoning/discourse system- just simple matching of domain concept RTs
- We have the 'closed-world' view that parseablility is {false,true}
- Probabilistic reasoning? At which 'level'?

Probabilistic TTR

$s : T = \{0, 1\}$

[Cooper, 2005]

Probabilistic TTR

$p(s : T) = [0, 1]$

[Cooper et al., 2014]

Order theory essentials

- An ordering relation on a set of elements of form $x \leq y$ means 'y includes x'.

  If order defined between some pairs of elements: a partial order (*poset*).
- *Meet*, the greatest lower bound ($\wedge$) and *join*, the least upper bound ($\vee$) of two elements.

Order theory essentials

A poset with all elements closed under meet and join is a *lattice*.

Top ($\top$) and bottom ($\bot$) elements.

Complement of an element $\neg x$ such that:

$$x \wedge \neg x = \bot$$
$$x \vee \neg x = \top$$

Atoms are elements that cover (direct successors of) $\bot$.

Join-irreducible elements those not definable by join of two other elements.

Order theory essentials

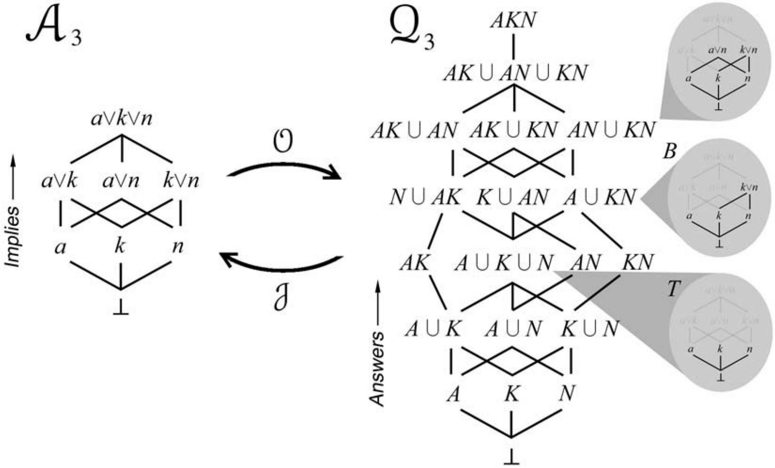
Atoms are elements that cover (direct successors of) $\bot$.

Join-irreducible elements those not definable by join of two other elements.

Distributed lattices can express any poset of sets ordered by the $\subset$ relation. Obey distributivity relations.

Complemented lattices express any lattice where every element x has a unique complement $\neg x$.

Previous work: Probability theory and lattices (Knuth)



Previous work: Probability theory and lattices (Knuth)

- Knuth's *Inquiry Calculus* generalises Boolean logic to probability and information theory through *distributed lattices*

Previous work: Probability theory and lattices (Knuth)

- Knuth's *Inquiry Calculus* generalises Boolean logic to probability and information theory through *distributed lattices*
- Boolean operators $\wedge$ and $\vee$ and $\neg$ happily coincide with the order-theoretic relations
- Derives probabilities from function on the lattice $Z(x, y)$, the degree to which x includes/implies y:

$$p(x \mid y) = Z(x, y) = \begin{cases} 1 & \text{if } y \rightarrow x \\ 0 & \text{if } x \wedge y = \perp \\ p & \text{otherwise, where } 0 \leq z \leq 1 \end{cases}$$

- Normal probability theory applies: sum rule, product rule, Bayes theorem

Previous work: Probability theory and lattices (Knuth)

- Knuth's *Inquiry Calculus* generalises Boolean logic to probability and information theory through *distributed lattices*
- Boolean operators $\wedge$ and $\vee$ and $\neg$ happily coincide with the order-theoretic relations
- Derives probabilities from function on the lattice $Z(x, y)$, the degree to which x includes/implies y:

$$p(x \mid y) = Z(x, y) = \begin{cases} 1 & \text{if } y \rightarrow x \\ 0 & \text{if } x \wedge y = \bot \\ p & \text{otherwise, where } 0 \leq z \leq 1 \end{cases}$$

- Normal probability theory applies: sum rule, product rule, Bayes theorem
- Question lattice: a question's *relevance* to the central issue

Record Type lattices

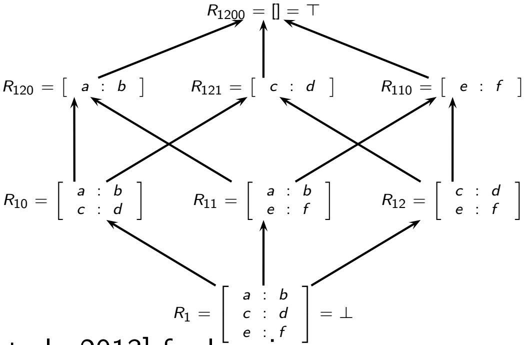
- RT lattice G ordered by the relation ‘is a subtype of’ $x \sqsubseteq y$
- Meet is maximal common subtype $x \dot{\wedge} y$
- Join is minimal common supertype $x \dot{\vee} y$

Record Type lattices

- RT lattice G ordered by the relation 'is a subtype of'
  $x \sqsubseteq y$
- Meet is maximal common subtype $x \curlywedgedot y$
- Join is minimal common supertype $x \curlyveedot y$
- Guaranteed to be *distributive* as long as it has a $\bot$ and $\top$, often the empty type [ ], not generally *complemented*

$$x \curlywedgedot (y \curlyveedot z) = (x \curlywedgedot y) \curlyveedot (x \curlywedgedot z) \quad \text{(D1. Distributivity of } \curlywedgedot \text{ over } \curlyveedot \text{)}$$
$$x \curlyveedot (y \curlywedgedot z) = (x \curlyveedot y) \curlywedgedot (x \curlyveedot z) \quad \text{(D2. Distributivity of } \curlyveedot \text{ over } \curlywedgedot \text{)}$$

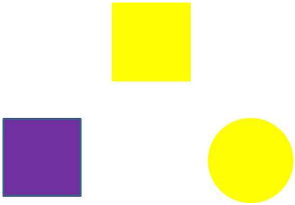
Record Type lattices



[Eshghi et al., 2013] for learning

A familiar psycholinguistic experiment

- [Brennan and Schober, 2001]
  'Pick the yell-, uh, purple square'



A familiar psycholinguistic experiment

Disjunction of final situations are *the atoms*.

Overall probability mass in lattice L is $P(L)$ global denominator.

$$\begin{bmatrix} x & : & ind \\ col_p & : & purple(x) \\ shp_{sq} & : & square(x) \end{bmatrix} \frac{\|PSq\|}{P(L)} \qquad \begin{bmatrix} x & : & ind \\ col_y & : & yellow(x) \\ shp_{sq} & : & square(x) \end{bmatrix} \frac{\|YSq\|}{P(L)} \qquad \begin{bmatrix} x & : & ind \\ col_y & : & yellow(x) \\ shp_c & : & circle(x) \end{bmatrix} \frac{\|YC\|}{P(L)}$$

$$\bot = 0$$

A familiar psycholinguistic experiment

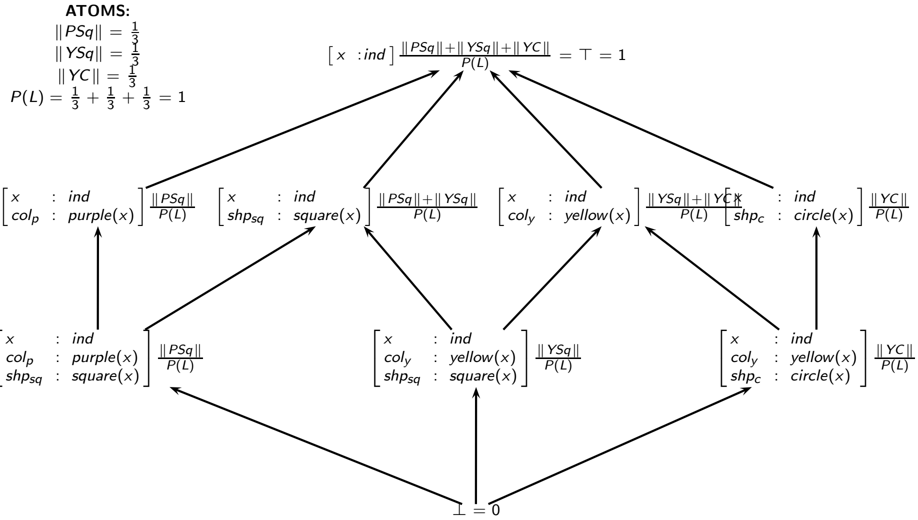


Figure : Record type lattice with initial uniform prior probablities

A familiar psycholinguistic experiment

ATOMS:

$\|PSq\| = \frac{1}{3}$

$\|YSq\| = \frac{1}{3}$

$\|YC\| = \frac{1}{3}$

$P(L) = \frac{1}{3} + \frac{1}{3} + \frac{1}{3} = 1$

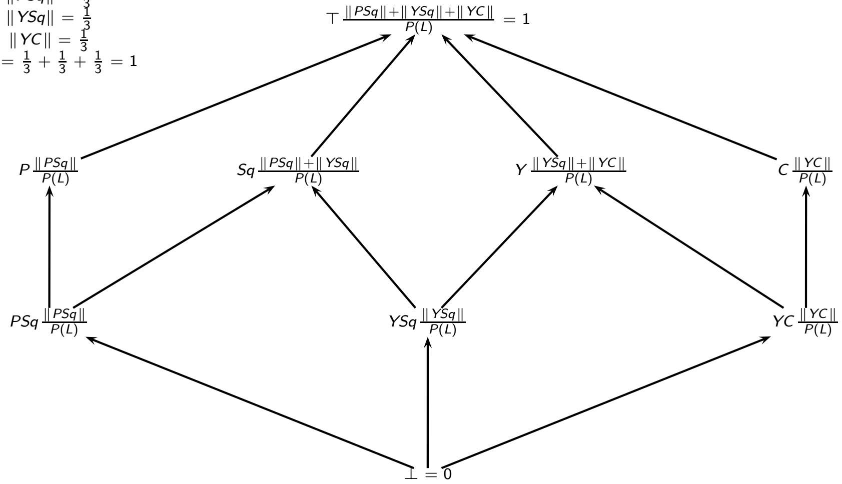
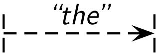
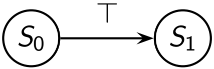


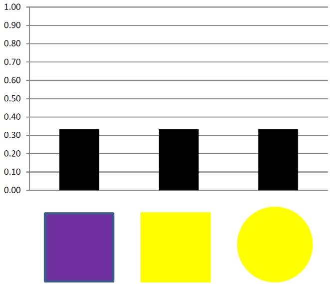
Figure : Record type lattice with initial uniform prior probablities

Self-repair:
IF parse(W) at vertex S_n unlikely OR IF $p(s : R_x \mid W)$ for $R_x \in G$ is unlikely
THEN (1) backtrack: parse(W) from vertex S_{n-1}. IF successful (2) `add` a new edge to the top path ELSE set $n = n - 1$ and repeat (1).

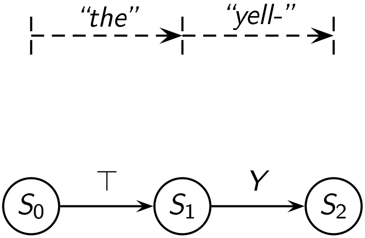
Interpreting disfluencies incrementally

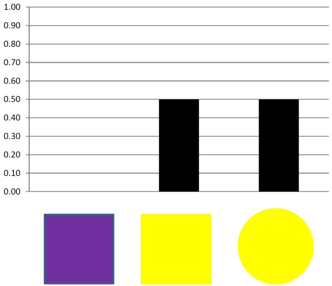




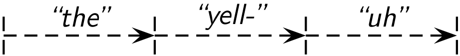


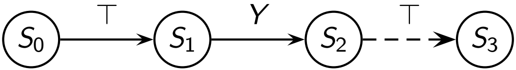
Interpreting disfluencies incrementally

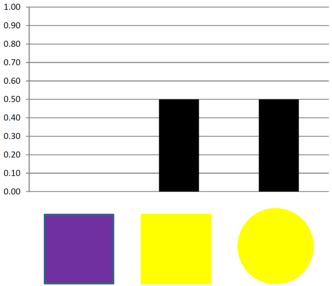




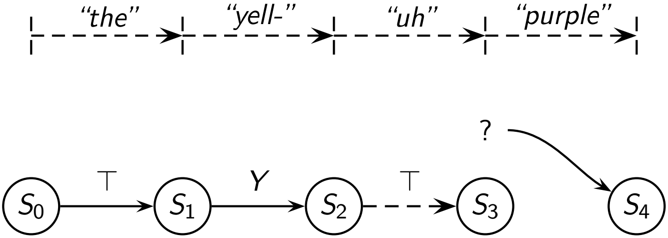
Interpreting disfluencies incrementally

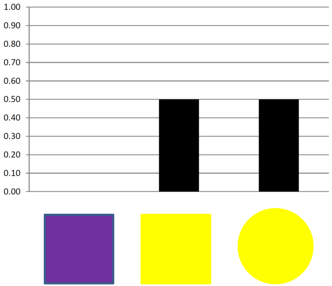




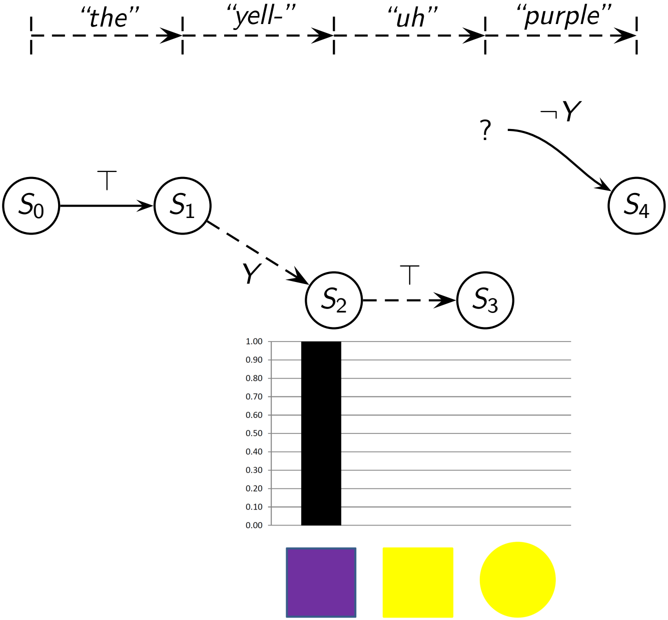


Interpreting disfluencies incrementally

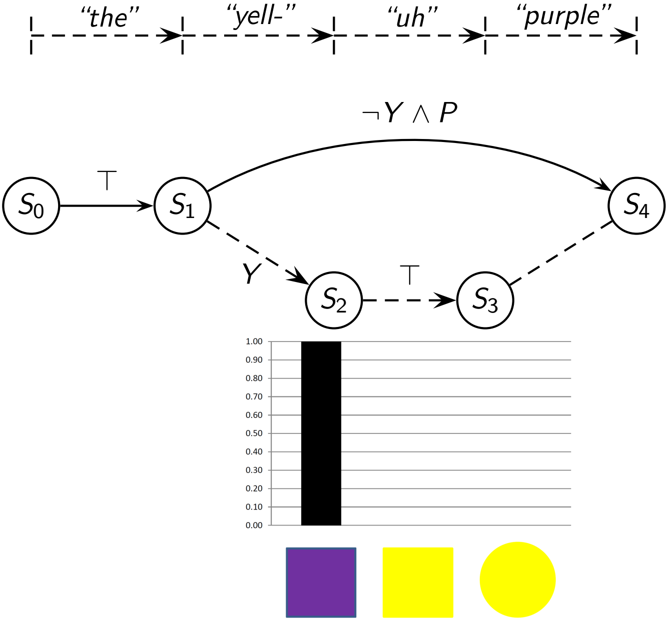




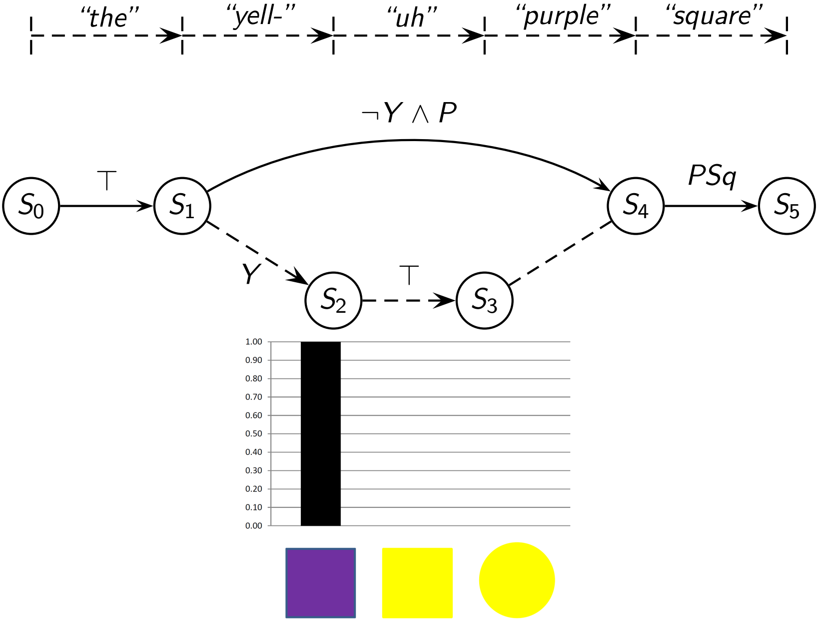
Interpreting disfluencies incrementally



Interpreting disfluencies incrementally



Interpreting disfluencies incrementally



Discussion

- Probabilistic inference can be formally characterized in prob TTR lattices.
- They can also be extended for efficient on-line learning.

Discussion

- Probabilistic inference can be formally characterized in prob TTR lattices.
- They can also be extended for efficient on-line learning.
- *but* Conversation participants in the wild cannot observe the entire situation.

  Local lattices generated with issues [Larsson, 2002] the way forward?
- A different more direct classification approach [Kennington and Schlangen, 2014]

Discussion

- Probabilistic inference can be formally characterized in prob TTR lattices.
- They can also be extended for efficient on-line learning.
- *but* Conversation participants in the wild cannot observe the entire situation.

  Local lattices generated with issues [Larsson, 2002] the way forward?
- A different more direct classification approach [Kennington and Schlangen, 2014]

Conclusion: Disfluency and other ‘dirty’ stuff

- Real dialogue is full of things like:
  - **Filled pauses**
  - **Fillers (discourse markers, edit terms)**
  - **Self-repairs**
  - Unfilled pauses (i.e. mid-turn silence/hesitation)
  - Laughter
  - Laughed speech
  - Exclamations (oh!, damn!, s***!)

Conclusion: Disfluency and other ‘dirty’ stuff

- Real dialogue is full of things like:
  - **Filled pauses**
  - **Fillers (discourse markers, edit terms)**
  - **Self-repairs**
  - Unfilled pauses (i.e. mid-turn silence/hesitation)
  - Laughter
  - Laughed speech
  - Exclamations (oh!, damn!, s***!)
- Are these problems, or *solutions*? [Clark, 1996]
- What do they *mean* in dialogue?

Conclusion: Disfluency and other ‘dirty’ stuff

- Real dialogue is full of things like:
    - **Filled pauses**
    - **Fillers (discourse markers, edit terms)**
    - **Self-repairs**
    - Unfilled pauses (i.e. mid-turn silence/hesitation)
    - Laughter
    - Laughed speech
    - Exclamations (oh!, damn!, s***!)
- Are these problems, or *solutions*? [Clark, 1996]
- What do they *mean* in dialogue?
- The DUEL project will tell us!

Thanks!

especially to:

- Matt Purver
- DUEL project (Bielefeld University and Paris 7, DFG and ANR)

Brennan, S. and Schober, M. (2001).
How listeners compensate for disfluencies in spontaneous speech* 1.
Journal of Memory and Language, 44(2):274–296.

Clark, H. H. (1996).
Using Language.
Cambridge University Press.

Cooper, R. (2005).
Records and record types in semantic theory.
Journal of Logic and Computation, 15(2):99–112.

Cooper, R., Dobnik, S., Lappin, S., and Larsson, S. (2014).
A probabilistic rich type theory for semantic interpretation.
In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS)*, Gothenburg, Sweden. Association for Computational Linguistics.

Eshghi, A., Hough, J., and Purver, M. (2013).
Incremental grammar induction from child-directed dialogue utterances.
In *Proceedings of the Fourth Annual Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, pages 94–103, Sofia, Bulgaria. Association for Computational Linguistics.

Ginzburg, J., Fernández, R., and Schlangen, D. (2014).
Disfluencies as intra-utterance dialogue moves.
Semantics and Pragmatics, 7(9):1–64.

Hough, J. (2011).
Incremental semantics driven natural language generation with self-repairing capability.
In *Proceedings of the Student Research Workshop associated with RANLP 2011*, pages 79–84, Hissar, Bulgaria.

Kempson, R., Meyer-Viol, W., and Gabbay, D. (2001).
Dynamic Syntax: The Flow of Language Understanding.
Blackwell, Oxford.

Kennington, C. and Schlangen, D. (2014).
Situated incremental natural language understanding using markov logic networks.
Computer Speech & Language, 28(1):240–255.

Knuth, K. H. (2005).
Lattice duality: The origin of probability and entropy.
Neurocomputing, 67:245–274.

Larsson, S. (2002).
Issue-based Dialogue Management.
PhD thesis, Göteborg University.
Also published as Gothenburg Monographs in Linguistics 21.

Levelt, W. (1989).
Speaking: From intention to articulation.
Mit Pr.

Milward, D. (1991).
Axiomatic Grammar, Non-Constituent Coordination and Incremental Interpretation.
PhD thesis, University of Cambridge.

Purver, M., Eshghi, A., and Hough, J. (2011).
Incremental semantic construction in a dialogue system.
In Bos, J. and Pulman, S., editors, *Proceedings of the 9th International Conference on Computational Semantics*, pages 365–369, Oxford, UK.

Purver, M. and Kempson, R. (2004).
Incremental context-based generation for dialogue.
In Belz, A., Evans, R., and Piwek, P., editors, *Proceedings of the 3rd International Conference on Natural Language Generation (INLG04)*, number 3123 in Lecture Notes in Artifical Intelligence, pages 151–160, Brockenhurst, UK. Springer.

Schlangen, D. and Skantze, G. (2009).
A general, abstract model of incremental dialogue processing.
In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 710–718, Athens, Greece. Association for Computational Linguistics.

Schlangen, D. and Skantze, G. (2011).
A general, abstract model of incremental dialogue processing.
Dialogue and Discourse, 2(1):83–111.

Shriberg, E. (1994).
Preliminaries to a Theory of Speech Disfluencies.
PhD thesis, University of California, Berkeley.

Skantze, G. and Hjalmarsson, A. (2010).
Towards incremental speech generation in dialogue systems.
In *Proceedings of the SIGDIAL 2010 Conference*, pages 1–8, Tokyo, Japan. Association for Computational Linguistics.