CENTRE FOR LANGUAGE TECHNOLOGY
MACQUARIE UNIVERSITY – SYDNEY

Text Planning in the Small: Referring Expressions

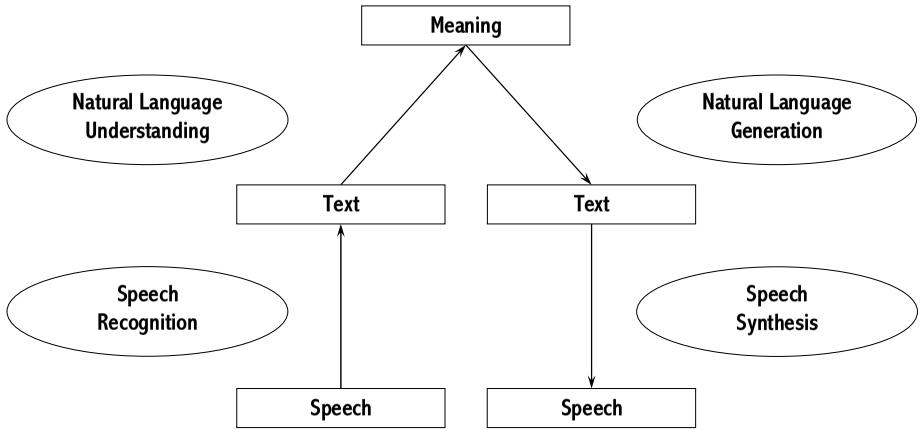
Robert Dale

rdale@ics.mq.edu.au
www.clt.mq.edu.au

Overview

- **Recap: Natural Language Generation**
- **Background: Reference and Anaphora**
- **Generating Referring Expressions**
- **Conclusions and Observations**

NLG vs NLU



Inputs and Outputs

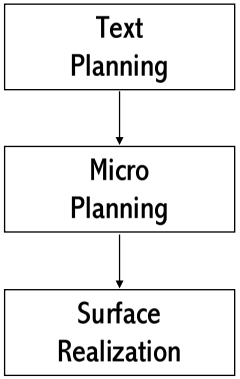
The inputs to NLG:

- **A knowledge source**
- **A communicative goal**
- **A user model**
- **A discourse model**

The output of NLG:

- **A text, possibly embodied as part of a document or within a speech stream**

The Architecture of an NLG System



- Content Determination
- Text Structuring
- Lexicalisation
- Aggregation
- Referring Expression Generation
- Syntax, morphology, orthography and prosody

Overview

- Recap: Natural Language Generation
- Background: Reference and Anaphora
- Generating Referring Expressions
- Conclusions and Observations

Reference

- Reference is a relationship between symbols and the world.
- A referring expression is a linguistic form (typically a noun phrase) which refers to some referent (an entity in some real or imaginary world).

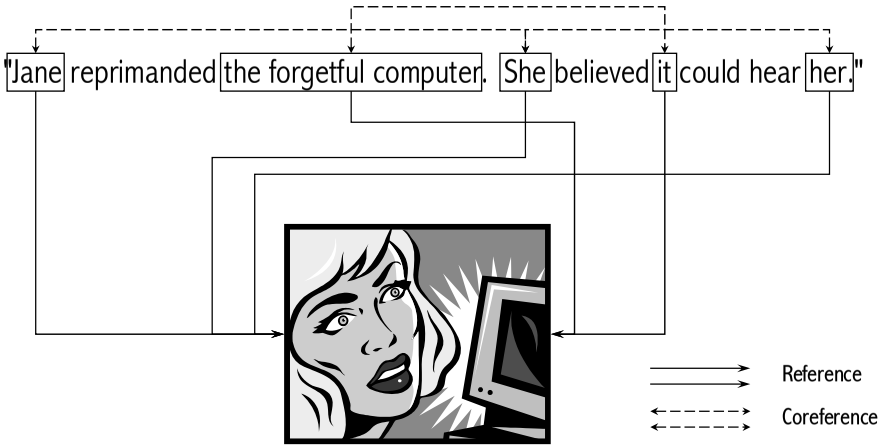
Anaphora

- The phenomenon: anaphora
- Instances of the phenomenon: anaphors
- An anaphor is …
  - an abbreviated linguistic form …
  - … where the entity referred to can only be determined by making use of contextual information …
  - … and not from the content of the linguistic form itself

Anaphora Resolution

- **The general problem:**
  - **Given an anaphoric expression, how do we work out which entity it refers to?**

Some Terminology



Anaphors and Antecedents

- **"Jane" is the antecedent of the two anaphors "she" and "her"**
- **"the forgetful computer" is the antecedent of the anaphor "it"**
- **Both the anaphor and its antecedent refer to something in some real or imaginary world**
- **The anaphor and the antecedent co-refer**
- **Both antecedent and anaphor are noun phrases**

Kinds of Anaphora

- **Pronominal Anaphora**
- **Definite Noun Phrase Anaphora**
- **One-Anaphora**

Kinds of Anaphora #1

- **Pronominal Anaphora**

  A plane has just landed.

  It is operated by Qantas.

Kinds of Anaphora #2

- **Definite Noun Phrase Anaphora**

  A 737 and a 747 have both just landed.

  The 737 is operated by Qantas.

Kinds of Anaphora #3

- **One-Anaphora**

  Two planes have just landed.

  The smaller one is operated by Qantas.

Definite and Indefinite Reference

The simple model:

- **An indefinite reference is generally used to introduce an entity into a discourse.**
  - **I have a flight at 2pm.**
- **A definite reference is generally used to subsequently refer to an entity already introduced into the discourse.**
  - **Ok, I'll take it.**

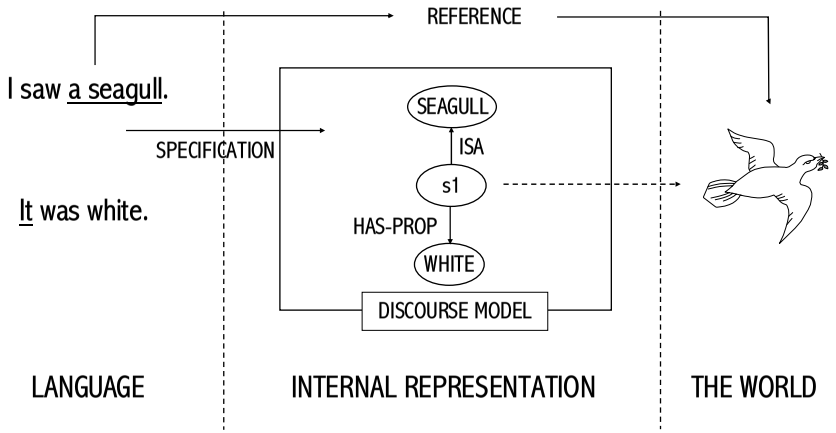
Definite Reference

- Definite noun phrases
  - I'll take the flight that goes to Adelaide via Melbourne.
- Demonstrative terms [exophoric reference]
  - That plane over there is leaving shortly.
- Proper names
  - That flight is operated by United Airlines.
- Pronouns
  - It departs in two hours' time.
- Definite one-anaphoric noun phrases
  - I'll take the last one.

The Problem from the NLU Perspective

- How do we model the process of reference resolution?
- General intuition:
  - When a speaker uses a referring expression, he or she is specifying an object in his or her discourse model, with the intent of causing the object in question to be introduced or identified in the hearer's discourse model.

Discourse Models



Using a Discourse Model

- When a new entity is introduced into the discourse, introduce a new symbol into the discourse model
- This symbol serves as a 'conceptual coathook': as new information is received about the entity, attach this information to the corresponding symbol

Overview

- Recap: Natural Language Generation
- Background: Reference and Anaphora
- Generating Referring Expressions
- Conclusions and Observations

The Problem from an NLG Perspective

- Referring Expression Generation:

  Natural language generation systems need to know how to refer to entities in order to pick them out for the hearer

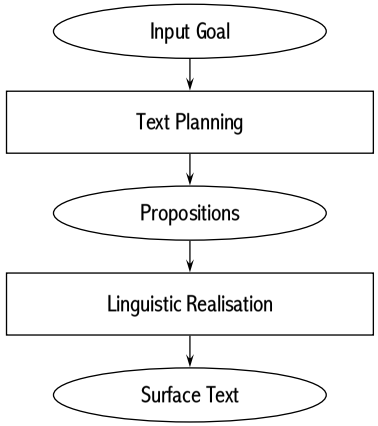
Referring Expressions

1a John has a red jumper.
1b He wears it on Sundays.

2a John has a red jumper and a blue jumper.
2b He wears the red one on Sundays.

3a John has a red jumper and blue cardigan.
3b He wears the jumper on Sundays.

The Generation Process



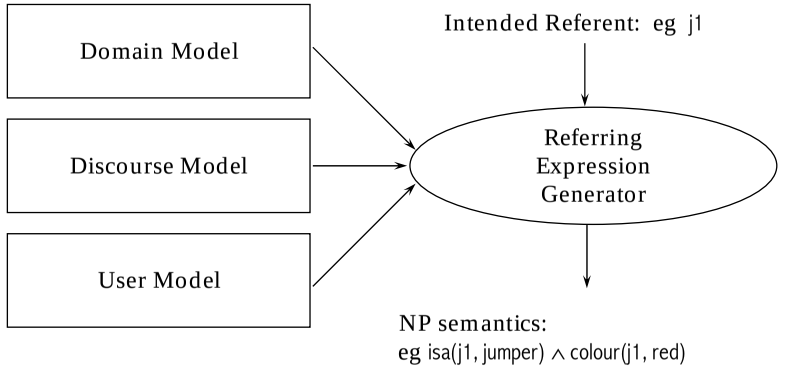
The Input Propositions

1a owns(j, j1)
1b wears(j, j1, s)

2a owns(j, {j1, j2})
2b wears(j, j1, s)

3a owns(j, {j1, c1})
3b wears(j, j1, s)

Knowledge Sources



The Generation Process

- Input Propositions
  - owns(j, {j1, c1}), wears(j, j1, s)
- Domain Knowledge
  - isa(j1, jumper), colour(j1, red), isa(c1, cardigan), colour(c1, blue)
- Output
  - John owns a red jumper and a blue cardigan.
  - He wears the jumper on Sundays.

The Domain Model

- The set of entities and eventualities that make up the domain
- Every entity e_i defined in terms of a set of attributes and values
- $D = \langle\langle e_1, a_m, v_{1m}\rangle, \langle e_1, a_n, v_{1n}\rangle, \ldots, \langle e_i, a_j, v_{ij}\rangle\rangle$

The Discourse Model

The discourse model is a triple ⟨C, P, H⟩ where:

- C is the set of entities referred to in the current clause (the *current clause contents*)
- P is the set of entities referred to in the current clause (the *previous clause contents*)
- H is a list of all the entities mentioned in the discourse before the previous clause (the *history list*)

Content Determination

- The Principle of Efficiency
- The Principle of Adequacy
- The Principle of Sensitivity

Content Determination

- Given:
  - a set of knowledge base entities C (the context set)
  - an intended referent r in C
  - a knowledge base of properties P
- find a set of properties in P which are together true of r but of no other entity in C.

Distinguishing Descriptions

A distinguishing description of *r* is the linguistic realisation of this set of properties.

A minimal distinguishing description of *r* is the linguistic realisation of the smallest such set of properties.

Distinguishing Descriptions

Let *r* be the intended referent, and *C* be the contrast set; then, a set *L* of attribute–value pairs will represent a distinguishing description if:

- Every attribute–value pair in *L* applies to *r*: that is, every element of *L* specifies an attribute–value that *r* possesses.
- For every member *c* of *C*, there is at least one element *l* of *L* that does not apply to *c*: i.e., there is an *l* in *L* that specifies an attribute–value that *c* does not possess.
- *l* is said to rule out *c*.

An Example

- Intended referent: a small black dog
- The contrast set: a large white dog and a small black cat
- Choose *black* to rule out the white dog, and *dog* to rule out the cat: the result is *the black dog*
- *The small dog* is also a distinguishing description.

Computing Distinguishing Descriptions

Three steps which are repeated until a successful description has been constructed:

- Check whether the description constructed so far is successful in picking out the intended referent from the context set.
- If it's not sufficient, choose the most useful fact that will contribute to the description.
- Extend the description with this fact, and reduce the context set accordingly.

Computing Distinguishing Descriptions

Initial Conditions:

$C_r = \langle$all entities$\rangle$; $P_r = \langle$all properties true of r$\rangle$; $L_r = \{\}$

1 Check Success

if $|C_r| = 1$ then return L_r as a distinguishing description

elseif $P_r = 0$ then return L_r as a non-dd

else goto Step 2.

2 Choose Property

for each $p_i \in P_r$ do: $C_{r_i} \leftarrow C_r \cap \{x \mid p_i(x)\}$

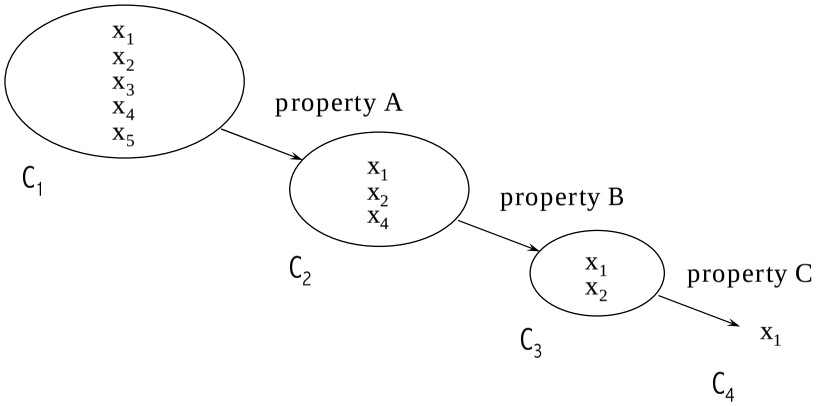
Chosen property is p_j, where C_{r_j} is smallest set.

goto Step 3.

3 Extend Description (wrt the chosen p_j)

$L_r \leftarrow L_r \cup \{p_j\}$; $C_r \leftarrow C_{r_j}$; $P_r \leftarrow P_r - \{p_j\}$; goto Step 1.

How It Works



An Example

Suppose we know:

- **cup(c1), cup(c2), cup(c3),**
- **blue(c1), red(c2), red(c3),**
- **big(c1), small(c2), big(c3)**

and we want to refer to c3.

Chosen properties:

- **red(c3) (or big(c3))**
- **big(c3) (or red(c3))**

Problems

The algorithm assumes that all properties are equal: it is only the relative discriminatory power, and nothing else, that causes a particular property to be selected.

Some properties are more useful than other properties which have the same discriminatory power.

A: "Which stop do I want for the cinema?"

B: "You should take the stop before mine."

What People Do

- **Context Set = b1, c1, c2**
- **Intended Referent = b1**
- **Domain Model:**
  - **bird(b1), white(b1)**
  - **cup(c1), black(c1)**
  - **cup(c2), white(c2)**
- **Typical description: "the white bird"**

A Revised Algorithm

- The algorithm iterates through the attributes in an ordered list of preferred-attributes
- The elements of this list and their order will vary with the domain, and should be determined by empirical investigation
- For each attribute, we check if specifying a value for it would rule out at least one member of the contrast set that has not already been ruled out; if so, this attribute is added to the set of properties to be used

The Algorithm Continued

- The process of adding attribute–value pairs continues until a referring expression has been formed that rules out every member of the contrast set
- A head noun (i.e., a value for the type attribute) is always included, even if it has no discriminatory power

Script-Based Reference

- Initial Conditions:
  - $C_r = \langle$all entities$\rangle$; P = $\langle$preferred attributes$\rangle$; $L_r = \{\}$
- 1 Check Success
  - if $|C_r| = 1$ then return L_r as a distinguishing description
  - elseif P = 0 then return L_r as a non-dd
  - else goto Step 2.
- 2 Evaluate Next Property
  - get next $p_i \in P$ such that userknows$(p_i(r))$
  - if $|\{x \in C_r \mid p_i(x)\}| < |C_r|$ then goto Step 3
  - else goto Step 2.
- 3 Extend Description (wrt the chosen p_j)
  - $L_r \leftarrow L_r \cup \{p_j\}$; $C_r \leftarrow C_{rj}$; goto Step 1.

Advantages of this Algorithm

- Fast: its run-time is linear in the number of distractors, and independent of the number of possible modifiers
- Sensitive to human preferences: it attempts to use easily perceivable attributes
- Domain-independent: the core algorithm should work in any domain, once an appropriate knowledge base and user model has been set up.

Overview

- Recap: Natural Language Generation
- Background: Reference and Anaphora
- Generating Referring Expressions
- Conclusions and Observations

Recap: Text Planning at Different Levels

- Discourse Structure:
  - Schema-based: 'text grammars'
  - Relation-based: relational notions of coherence
- Referring Expressions:
  - Domain-independent property choice
  - Domain-dependent script-based

Ways Forward for Referring Expression Generation

- Observation:
  - Text planning selects propositions that contribute to the discourse and integrates them into the ongoing discourse structure that is being developed
  - Referring expression generation selects properties that contribute to the reference and integrates them into the ongoing semantic structure that is being developed
- So: borrow ideas from discourse structure
  - Incorporate properties into descriptions by taking account of what they contribute to the ongoing discourse

Summary

- Discourse structure and referring expression generation are seen as two distinct problems in NLG
- A range of algorithms and solutions have been proposed for both
- There are many unanswered research questions in both domains
- One way forward: since both problems are concerned with selecting and organising content, look for a unified framework that covers both