CENTRE FOR LANGUAGE TECHNOLOGY
MACQUARIE UNIVERSITY - SYDNEY

Algorithms for Generating Referring Expressions: Do They Do What People Do?

Jette Viethen
jviethen@ics.mq.edu.au

Robert Dale
rdale@ics.mq.edu.au

The Message of This Talk

It is important to assess Referring Expression Generation algorithms against real human-produced data.

The algorithms don't always do what people do!

Outline

- The Project
- The Algorithms
- Domain and Data
- Knowledge Representation
- Results
- Future Work

The Project

Observations:

- Definition of a “good” referring expression is unclear
- Most existing algorithms not assessed against natural data

The Goal:

- Comparison of performance of three well-known algorithms to human-produced descriptions
- Can they generate the descriptions that humans produce?

The Algorithms Considered

- Full Brevity, greedy (Dale 1989)
- Incremental Algorithm (Dale and Reiter 1995)
- Relational Algorithm, greedy (Dale and Haddock 1991)

Full Brevity and Incremental Algorithms

Input: - The Intended Referent
- List of Distractors: all objects that fit the current description
- Knowledge Base of all facts true of the objects in the domain

1. Choose new property from the KB
2. Add the new property to the description
3. **IF** all distractors ruled out **THEN** Return description
   ELSEIF no properties left **THEN** Fail!
   ELSE go to Step 1

- Full Brevity: choose the property that rules out most distractors (greedy)
- Incremental Algorithm: choose the next property from a list that rules out any distractors

The Relational Algorithm

- Keeps referents on a stack
  - When choosing a relation, new object is pushed on the stack
  - Always works on the entity on top of the stack
- Uses a constraint network to keep track of distractors and chosen properties
  - The chosen properties are the constraints
  - Distractor sets are kept consistent with the property constraints in the current description

Domain and Task

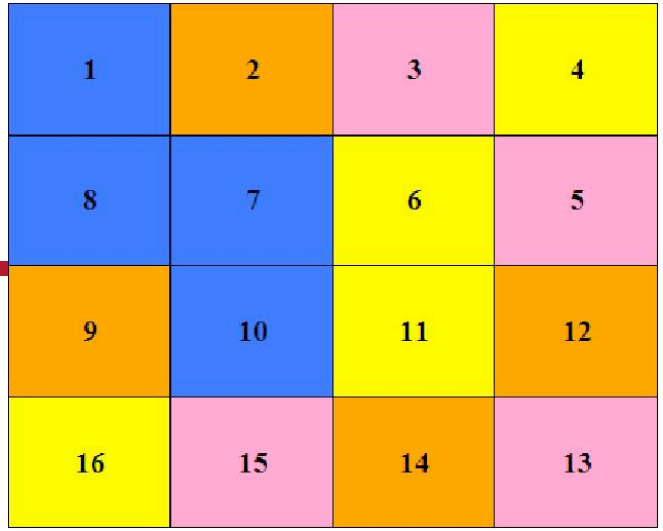
Domain:

- Grid of 4 × 4 filing cabinet drawers
- Each drawer has a number between 1 and 16
- 4 drawers each are blue, yellow, pink and orange

Task:

- Given the number of a drawer, describe it to an onlooker without mentioning any of the numbers.

The Data Set (1)



- 22 participants

  → 140 descriptions (3–12 per drawer)

- Filtering reduces this to 118 descriptions by excluding:
  - Reference to sets: The top pink drawer
  - Ambiguous descriptions: The orange one in the right corner
- Normalisation: lexical and syntactic variations

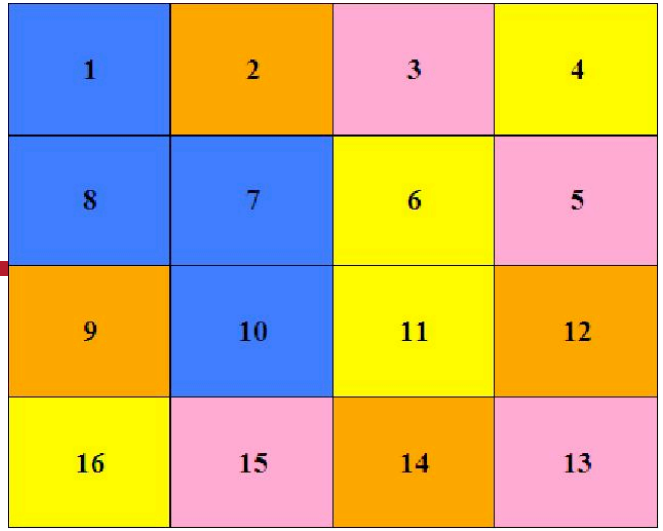
  The top drawer in the third filing cabinet

  → The top drawer in the third column

  The drawer that's in the top left corner

  → The drawer in the top left corner

The Data Set (2)



- Redundancy:
  - Minimal Descriptions: 75% (89)
    - d10: The blue drawer in the third row
  - Redundant Descriptions: 25% (29)
    - d1: The blue drawer in the top left corner
- Use of relations:
  - Absolute Properties Only: 87% (103)
  - Relational Descriptions: 13% (15)
    - d12: The orange drawer below the pink drawer

Knowledge Representation

- Encoded properties are chosen based on the properties explicit in the human data
- Absolute Properties: colour, row, column, corner
- Relations: above, below, right-of, left-of, next-to

Representation of Cornerhood

- inferred from row and column?
- explicit property in its own right?
  - Implementation as explicit property due to its salience
- more likely: corners are entities
  - drawers are spatially related to corners

Results

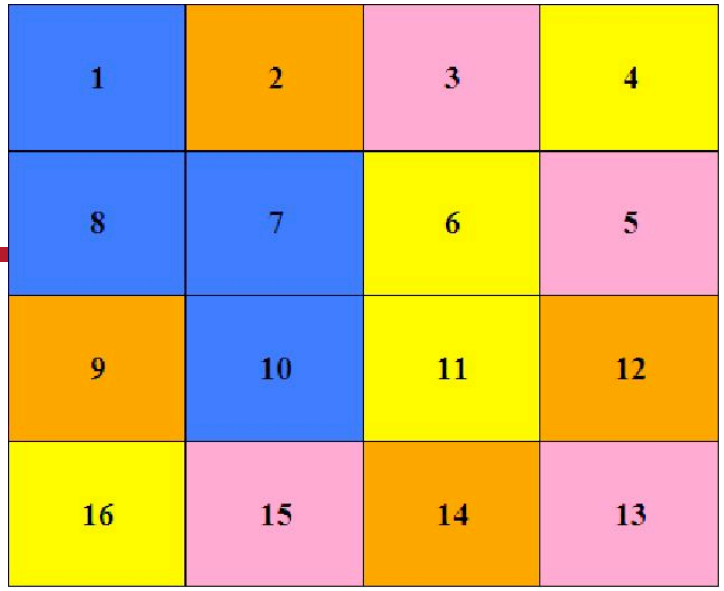
- Preference Ordering of Properties
  - Always consequential for Incremental Algorithm
  - Consequential for Greedy Algorithms if properties are equally discriminating
  - Allows the possibility of modelling description strategies

  → We test with all preference orderings appearing in the natural data set

- Coverage

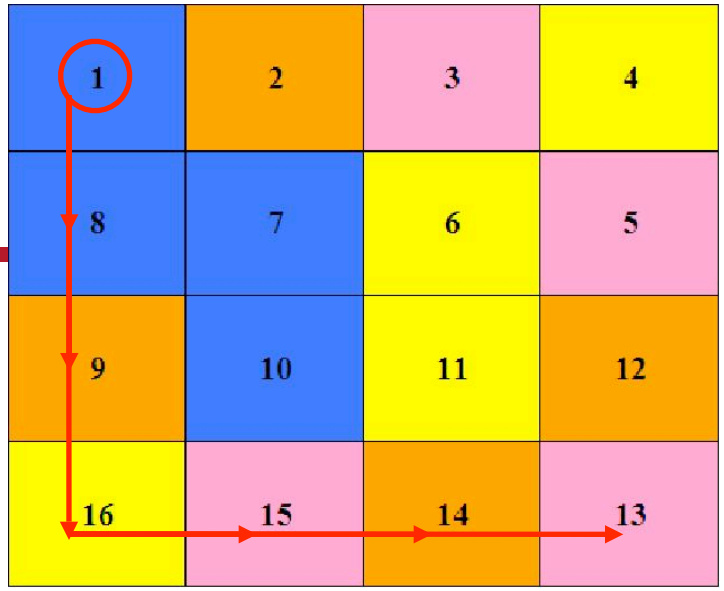
Algorithm \ Description type	Overall	Minimal	Redundant
Greedy	79.6% (103)	100% (79)	31.0% (24)
Incremental	95.1% (103)	100% (79)	82.8% (24)
Relational	0% (118)	0% (89)	0% (29)

Relational Results (1)



- In this domain, relational properties initially rule out more (all) distractors than absolute properties.
- Relational properties are treated like absolute properties; choice is based on the property value, for example:
  - colour(blue) rules out all but 3 distractors
  - left-of(d2) rules out all distractors

Relational Results (2)



In most cases the algorithm does a “corner run”:

Preference ordering:

(above – left_of – colour – row – column – corner – next_to – right_of – below)

((D1 ABOVE D8) (D8 ABOVE D9) (D9 ABOVE D16)

(D16 LEFT_OF D15) (D15 LEFT_OF D14)

(D14 LEFT_OF D13))

Observations

1. The Relational Algorithm doesn't work as advertised!
2. A lot depends on how you decide to represent knowledge.
3. Contrary to what algorithms implicitly assume, there is not one best referring expression for every object.

Future Work

- Referring Expressions involving relations
  - Graph-based algorithm (Krahmer, van Erk, Verleg 2003) shows promising results
- Gather psycholinguistic evidence
  - Review psycholinguistic literature on Referring Expression Generation
  - Examine larger and more complex natural data sets from naturalistic settings
  - Identify reasons why humans describe the same thing differently;

    e.g. reference strategies, underlying knowledge representation, salience of properties
- Model human reference behaviour in GRE
  - Integrate reference strategies into an algorithm
  - Use more sophisticated preference system for properties, e.g. graph costs
  - Allow multiple representations