Scope is useful!

Designing SQL queries using generalized quantifiers

Woodley Packard

DELPH-IN 2016

MRS — Scope?

- Labels, QEQ's, /EQ's, /H's, /NEQ's… they just get in the way, right?
- First thing most people do with an MRS is do their best to pretend it doesn't actually represent a logical form

Quick Review

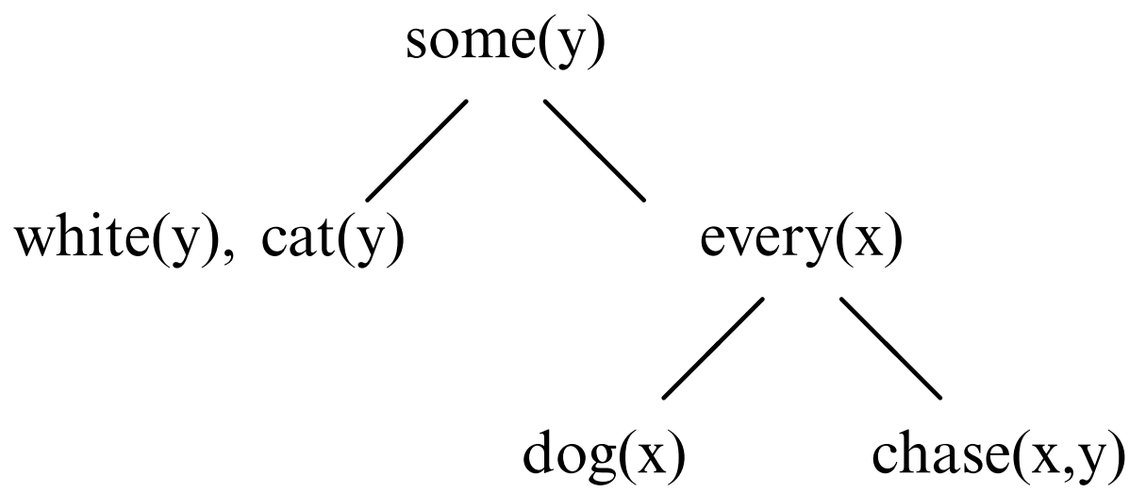
Every dog chases some white cat.

(18) $h1$: every(x, $h3$, $h8$), $h3$: dog(x), $h7$: white(y), $h7$: cat(y),
$h5$: some(y, $h7$, $h9$), $h4$: chase(x, y)

Diagrams from Copestake et al. 2005

Quick Review

(16) a. some(y, white(y) $\wedge$ cat(y), every(x, dog(x), chase(x, y)))
b.



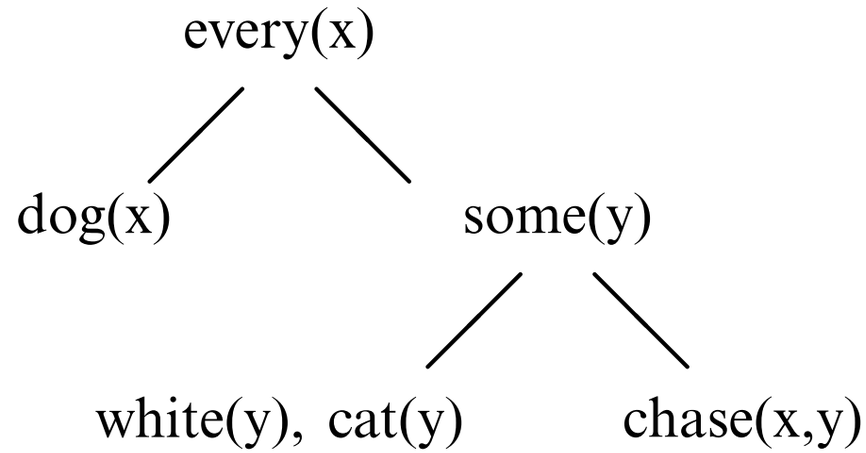
c. $h1$: every(x, $h3$, $h4$), $h3$: dog(x), $h7$: white(y), $h7$: cat(y),
$h5$: some(y, $h7$, $h1$), $h4$: chase(x, y)

Diagrams from Copestake et al. 2005

Quick Review

(17) a. every(x, dog(x), some(y, white(y) $\wedge$ cat(y), chase(x, y)))

b.



c. $h1$: every(x, $h3$, $h5$), $h3$: dog(x), $h7$: white(y), $h7$: cat(y),
$h5$: some(y, $h7$, $h4$), $h4$: chase(x, y)

Diagrams from Copestake et al. 2005

Scoping an MRS

- In general, N! scopings for N quantifiers
- Many or most are equivalent (lots of quantifiers commute), but real ambiguities are still common
- This demo: happy to find even one; heuristically aim for left-to-right precedence

Another review: Generalized Quantifiers

- Classical quantifiers: ∀, ∃
- Natural language has messier things
  - *seventeen*
  - *most*
  - *(almost but not quite all)*

Another review: Generalized Quantifiers

- Restriction and Body

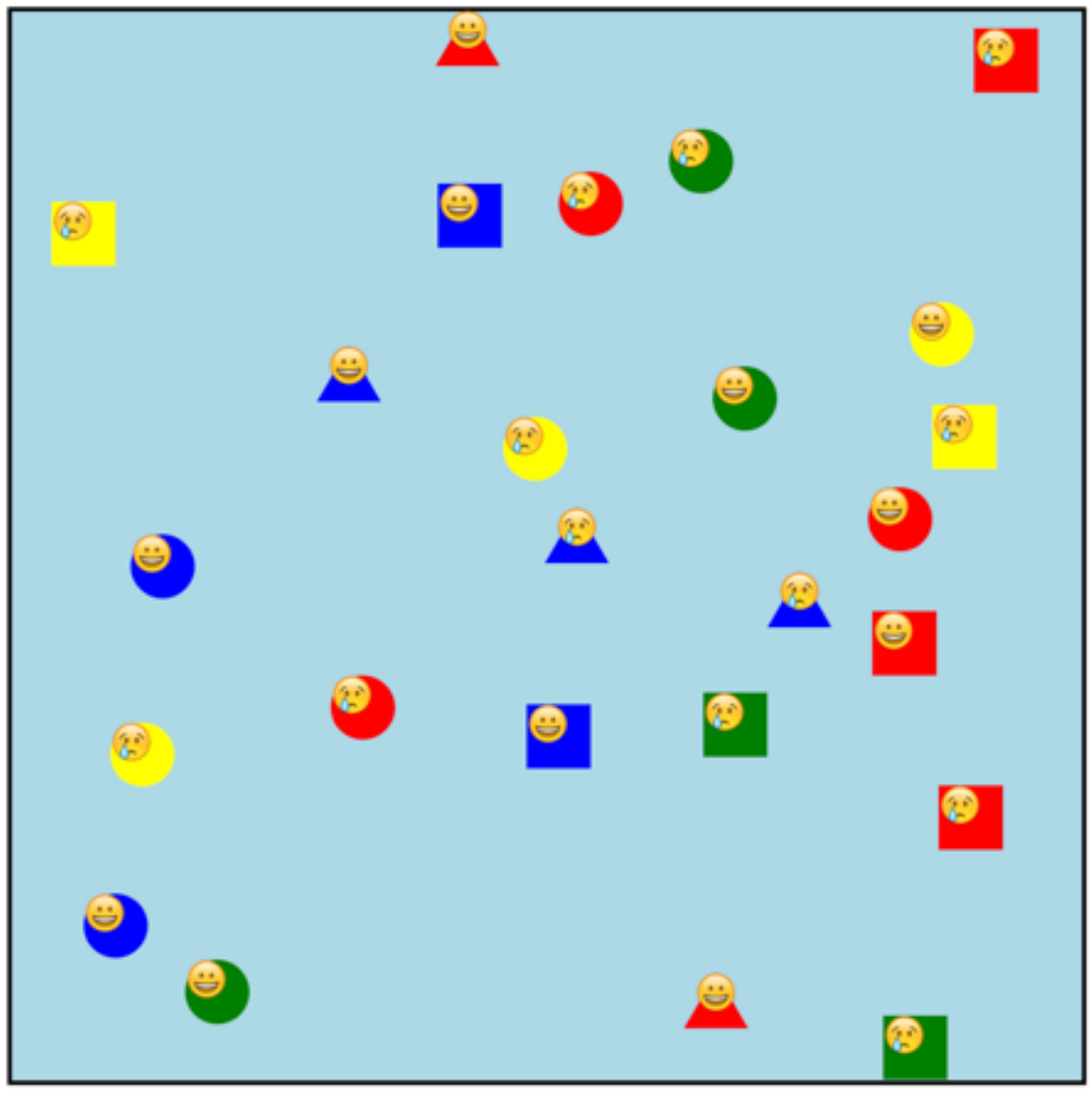
  [most x: dog(x)] bark(e, x)
  [most x: dog(x)][seventeen y: cat(y)] chase(e,x,y)

- Quantifier defines relationship between set denoted by restriction and set denoted by body

Interpreting scoped ERG-produced MRSes as SQL

- Toy world:
  squares, circles, triangles
  red, yellow, blue, green
  happy, sad
  above, below

- Single SQL relation:
  id, type, color, mood, x, y



MRS to SQL

- Recursively translate LF to SQL binary expressions (everything gets a truth value)

- Atomic examples:
  triangle_n_1(x) → x.type = "triangle_n_1"
  red_a_1(e,x) → x.color = "red_a_1"
  above_p(e,x,z) → x.y > z.y

MRS to SQL

- Quantifiers are more fun, but not complex:

  [∀x . R(x)] B(x) →

  ```
  (select count(*) from objects x where R(x) and B(x)) =
  (select count(*) from objects x where B(x))
  ```

  [∃x . R(x)] B(x) →

  ```
  (select count(*) from objects x where R(x)
  and B(x)) >= 1
  ```

MRS to SQL

- Language-y quantifiers:
  [most x . R(x)] B(x) →

```
2 * (select count(*) from objects x where
R(x) and B(x)) > (select count(*) from
objects x where B(x))
```

[seventeen x . R(x)] B(x) →

```
(select count(*) from objects x
where R(x) and B(x)) >= 17
```

MRS to SQL

- Negation
- Coordination
- Predicative NPs
  A red thing that is a square is happy.
  No squares are triangles.
- Demo!
  http://sweaglesw.org/linguistics/objects-demo/