Recap: Semantic rules

Lambda abstraction

- Used when something moves

  John λx I like x = I like John

- Used for making relative clauses & questions

  Who λx x does it = set of people who do it

- Used for representing predicates

Not smoking is healthy = Healthy (λx ~smoke(x))

Semantic rules (cont'd)

Function application:

- Used to put predicates and arguments together

  John runs

  John λx I like x

  Someone runs

Conjunction and other 'connective' rules:

- Take predicates that you want to conjoin
- Fully saturate them using variables
- Conjoin the resulting sentences
- Lambda abstract over the variables to get the new predicate of correct type

Generalised Quantifiers

- Try applying conjunction schema to
  "John and Mary"
- What is the type of these expressions?
  "Every guy but John"
  "Some apples and this pear"

Generalised Quantifiers (cont'd)

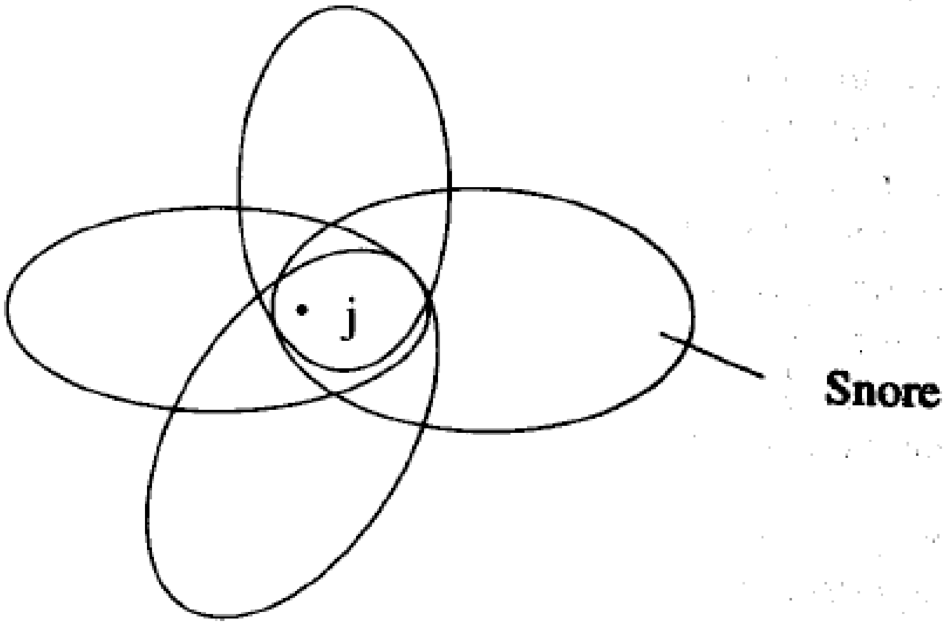
- Even worse: one might initially think that
  a unicorn
  is referential (refers to a particular individual)
  e.g., *A unicorn was there. He was beautiful.*

- However, as Bertrand Russel noted, indefinites are also non-referential:
  Nobody has seen a unicorn, because there aren't any.

Generalised Quantifier Theory

- Basic idea:
  All DPs (or as deSwart calls them, NPs) are of the same type – each is a **set of sets.**

[[Jane]] = {snore, run, talk, girl}



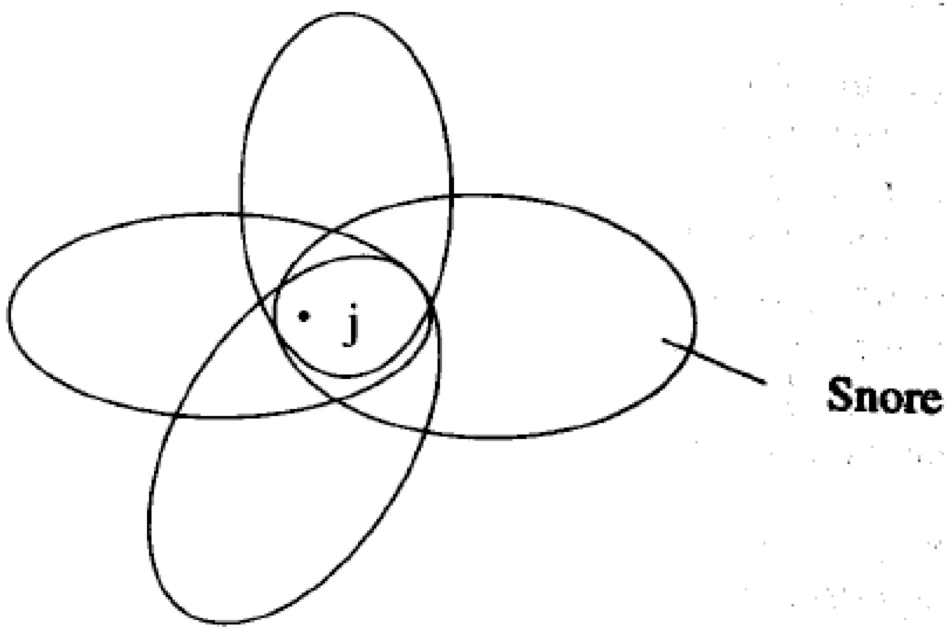
Jane snores

Generalised Quantifier Theory

- Basic idea:
  All DPs (or as deSwart calls them, NPs) are of the same type – each is a **set of sets.**

[[Jane]] = {snore, run, talk, girl}

$\lambda P.P(j)$



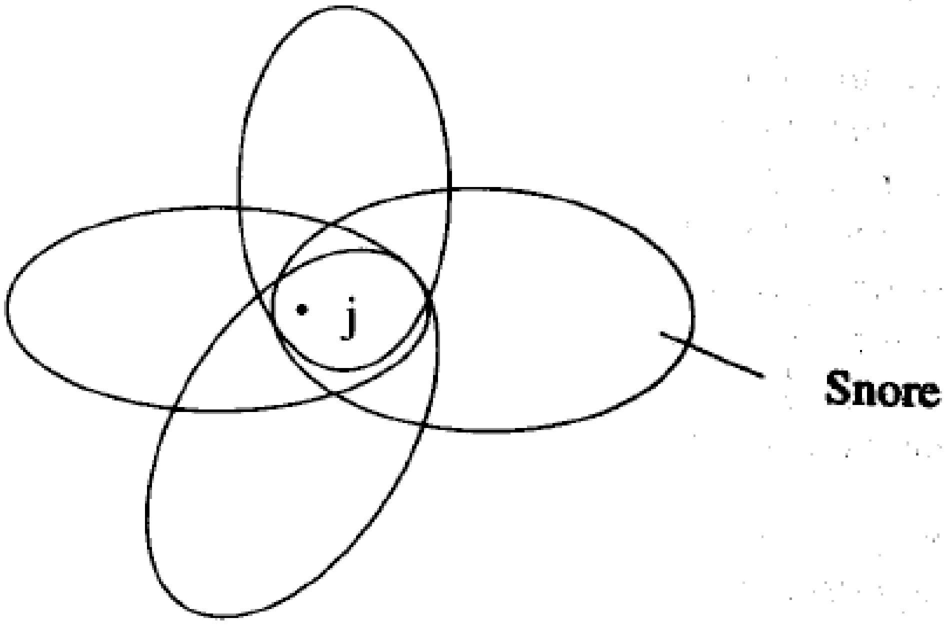
Jane snores

Generalised Quantifier Theory

- Basic idea:
  All DPs (or as deSwart calls them, NPs)
  are of the same type – each is a **set of sets.**

[[Jane]] = {snore, run, talk, girl}

$\lambda P.P(j)$ < et t >



Jane snores

Syntax and Semantics

S

DP ***VP***

⟦Jane snores⟧ = λP.P(j) (snore) =

<et t> <et>

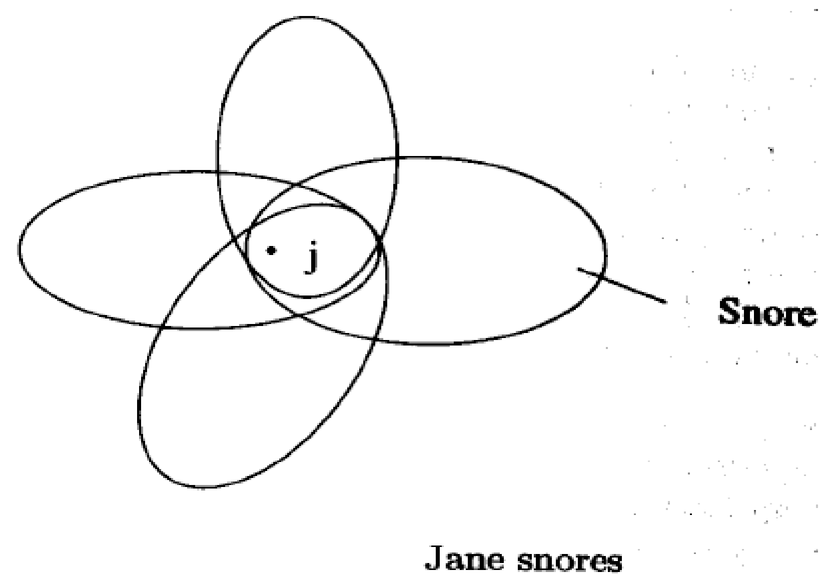
Syntax and Semantics

S

DP VP

$[\![\text{Jane snores}]\!] = \lambda P.P(j)\ (snore) = snore\ (j)$

$\langle et\ \ t\rangle \quad \langle et\rangle \quad t$



Syntax and Semantics

Mismatch for syntax and semantics:

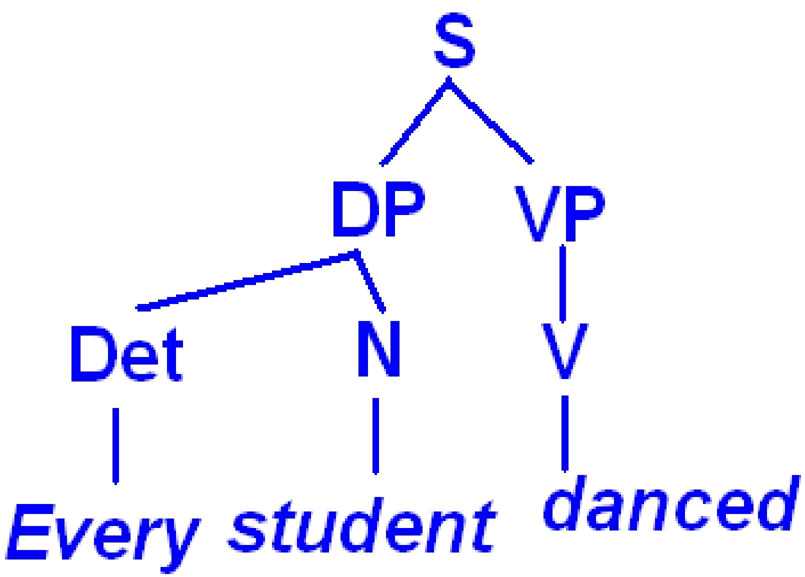
- What's the argument?
- What's the predicate?
- What is the constituent structure?
- Which individuals matter for the truth of S?

⟦Every student danced⟧ =

Every x [student(x) → danced(x)]

Syntax and Semantics

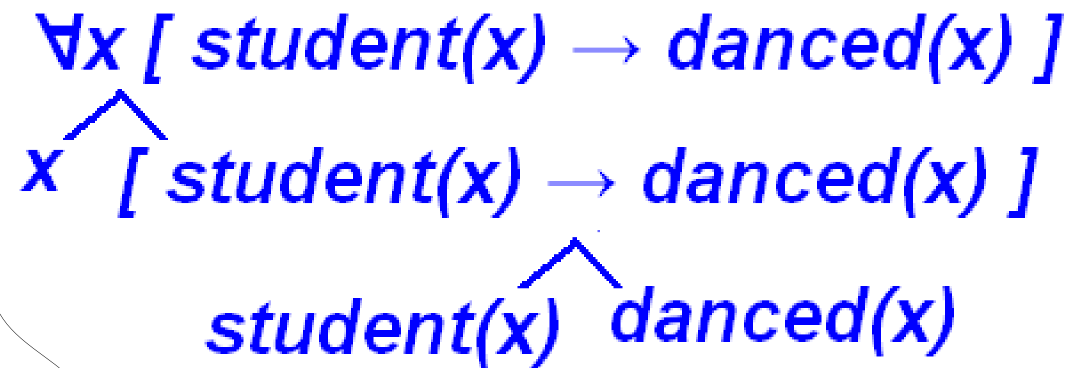
In English:



- “Every student” is a unit
- It combines with “danced”

In PC:

a totally different tree



- ⟦ every student ⟧ and ⟦danced⟧ are not constituents!
- “student danced” is a unit
- It combines with “every”

Syntax and Semantics

In English:

- Look in the **<u>set of students</u>**
  - If all members of this set danced – T
  - If not all members of this set danced – F

In PC:

- Look at all the entities in the **<u>universe</u>**
  - If the entity is not a student, ok
  - If the entity is a student,

  then if this entity danced – T

  otherwise - F

GQ Theory

- Try semantics which is more true to syntax:

⟦Every student⟧ = {dance, run, talk, student}

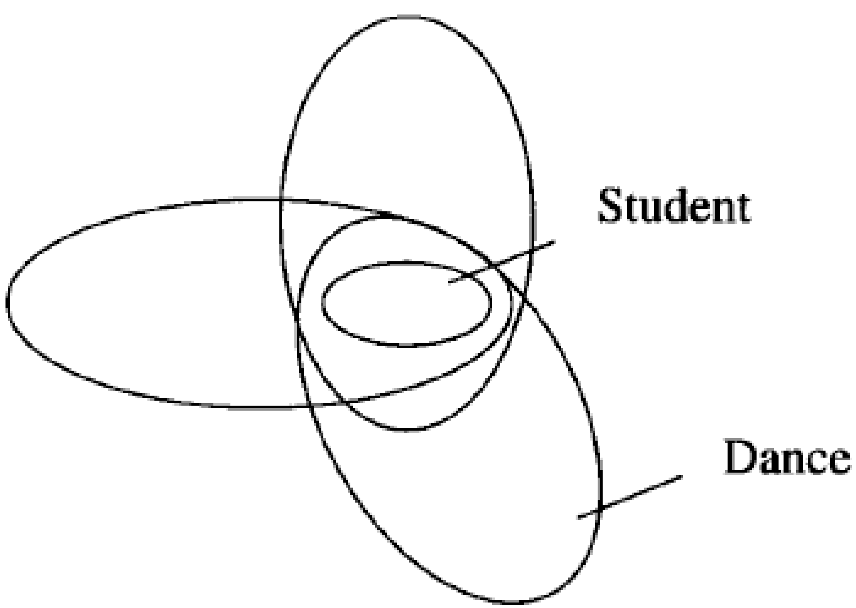
λP.Every(student)(P)   < et   t >

What's "***every***"?

Something that

combines with "***student***"

to make "***every student***"



Every student danced

GQ Theory

- What a determiner might mean:

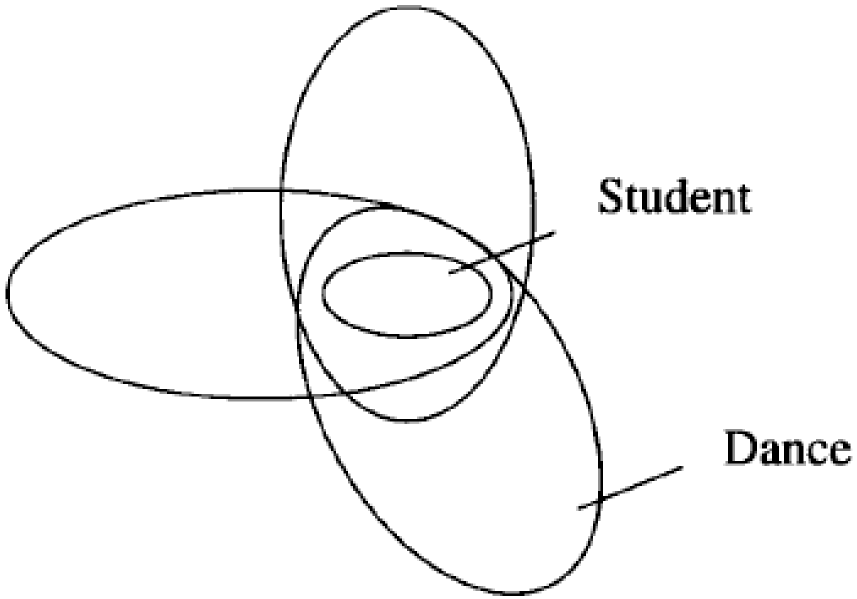
“***every***” - something that

combines with “***student***”

<et>

to make “***every student***”

<et t>



Every student danced

GQ Theory

- What a determiner might mean:

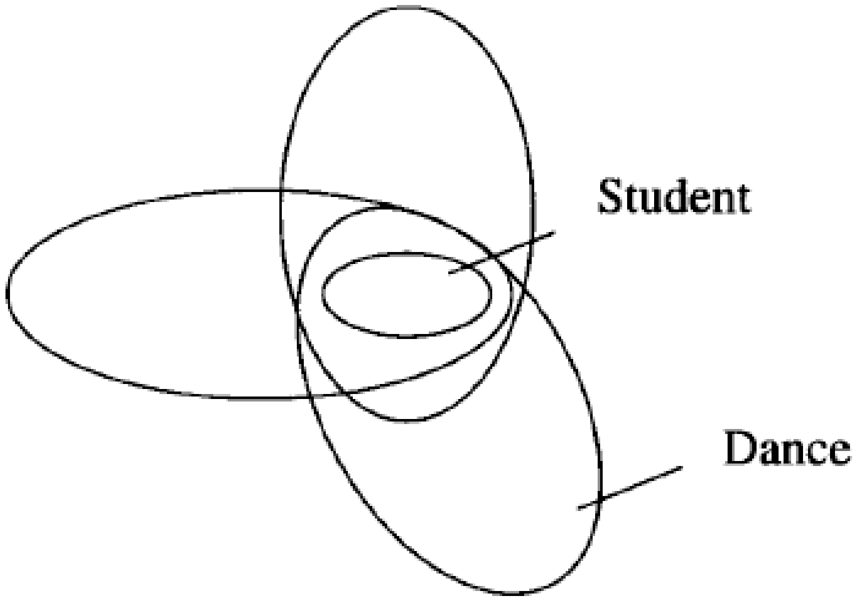
"***every***" - something that

combines with "***student***" ***<et>***

$\lambda Q_{<et>}.$

to make "***every student***"

$\lambda P_{<et>}.Every(student)(P)$



Every student danced

GQ Theory

- "***every***" - combines with

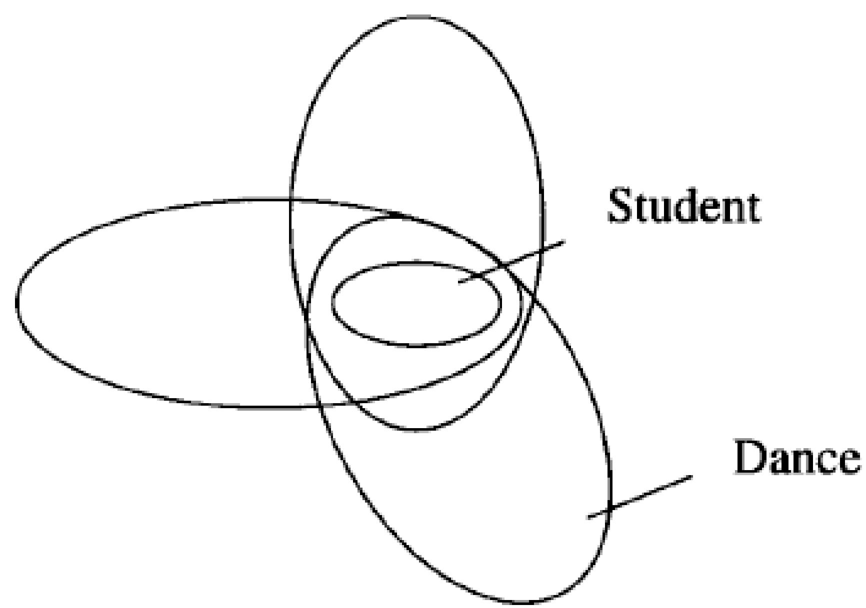
  "***student***" ***<e t>*** $\lambda Q_{<et>}.$ to make

"***every student***" $\lambda P_{<et>}.Every(student)(P)$

- SO: ***[[Every]] =***

  $\lambda Q \lambda P.Every(Q)(P)$

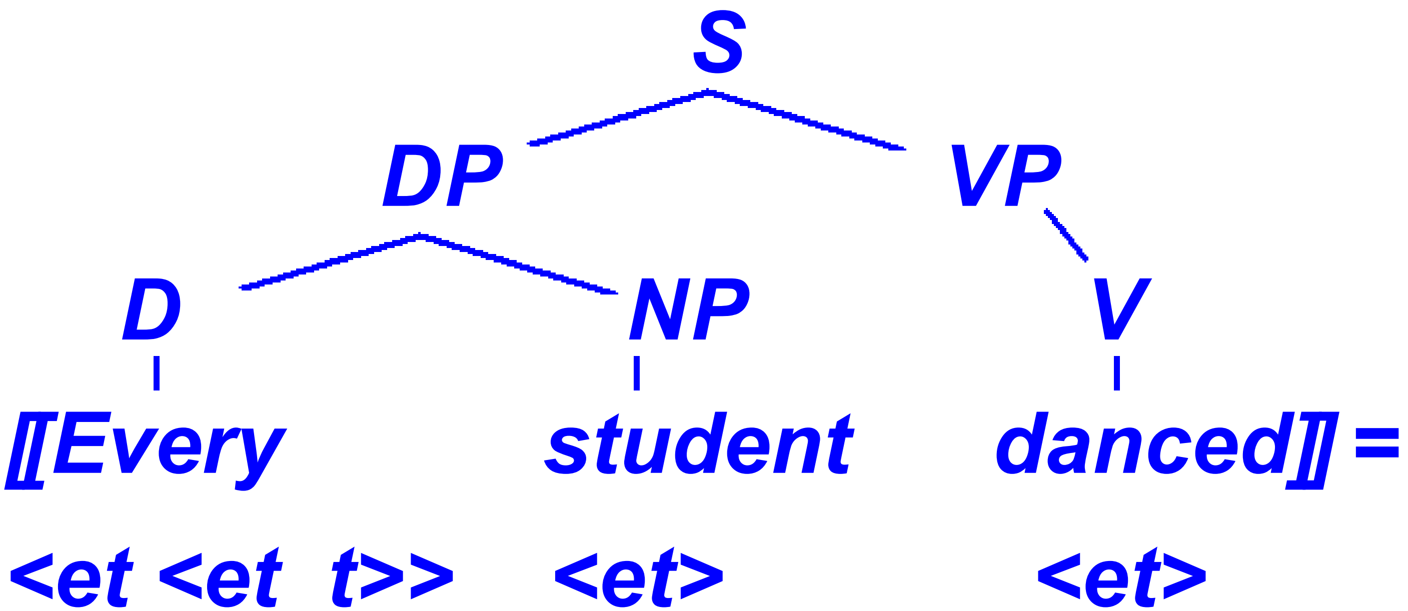
  <et <et t>>



Every student danced

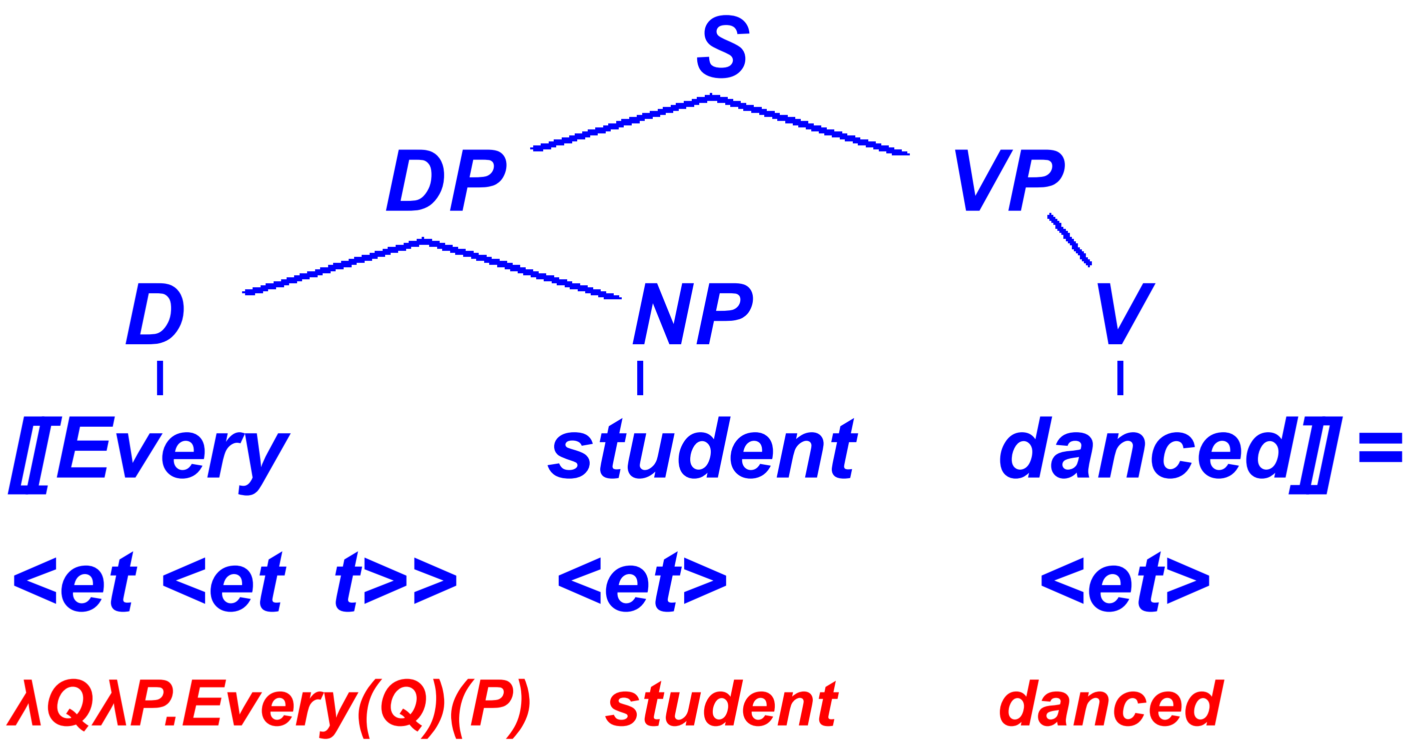
GQ Theory

- Try semantics which is more true to syntax:



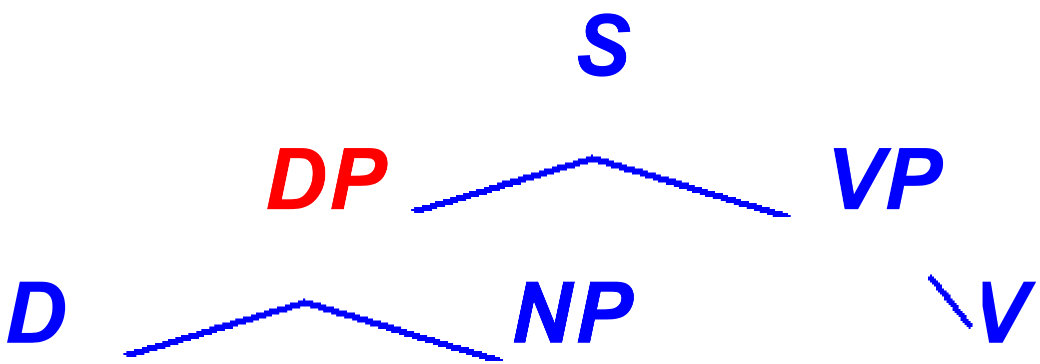
GQ Theory

- Try semantics which is more true to syntax:



GQ Theory

- Try semantics which is more true to syntax:



⟦Every student⟧ ⟦danced⟧ =

<et <et t>> <et> <et>

λQλP.Every(Q)(P) student danced

⟦DP⟧ = λP.Every(student)(P)

<et t>

GQ Theory

- Try semantics which is more true to syntax:

S

DP VP

D NP V

⟦Every student⟧ ⟦danced⟧ =

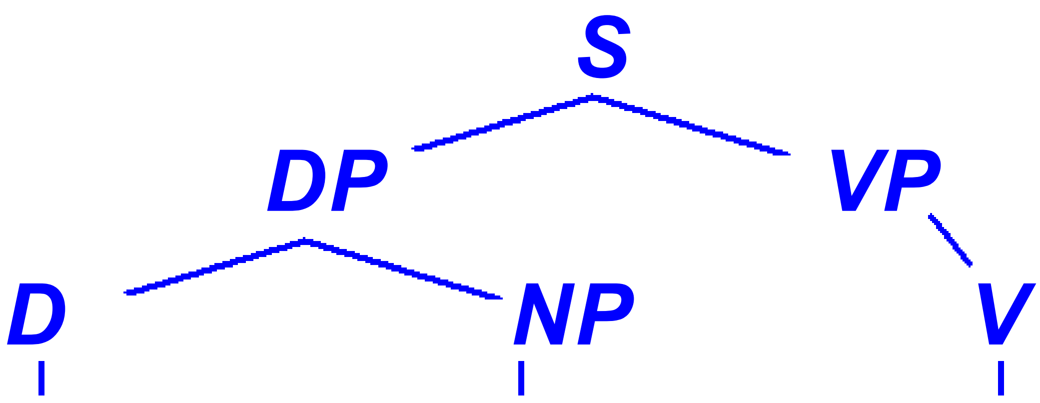
<et <et t>> <et> <et>

λP.Every(student)(P) danced

<et t>

GQ Theory

- Try semantics which is more true to syntax:



⟦Every student⟧ ⟦danced⟧ =

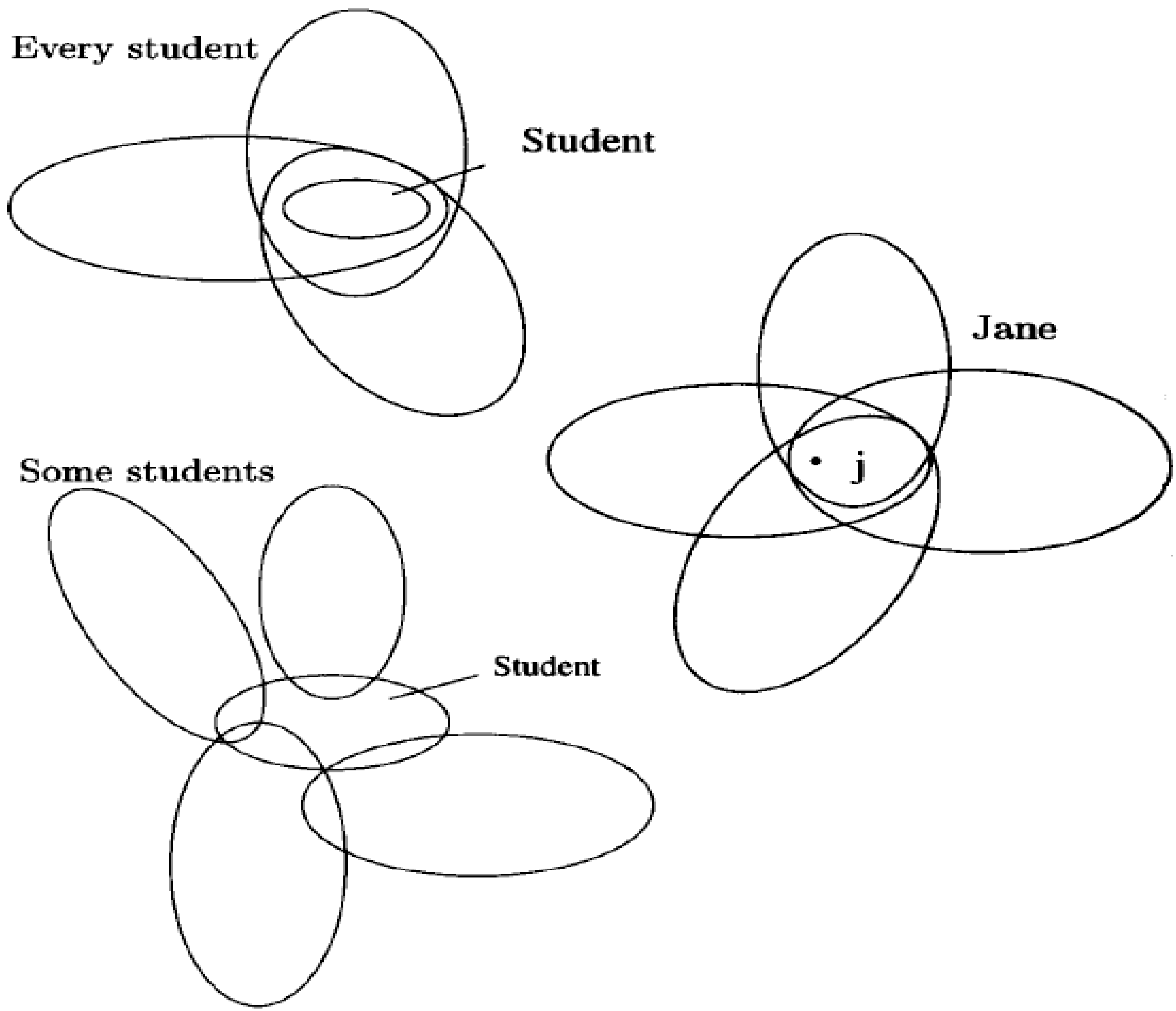
<et <et t>> <et> <et>

λP.Every(student)(P) danced

⟦S⟧ = Every(student)(danced)

t

Several GQs



Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧=
- ⟦Some NP⟧=
- ⟦No NP⟧=
- ⟦At least 5 NP⟧=
- ⟦Most NP⟧=

Several Determiners

⟦All (A,B)⟧=

⟦Some (A,B)⟧=

⟦No (A,B)⟧=

⟦At least 5 (A,B)⟧=

⟦Most (A,B)⟧=

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧=

All Ling 130 students are smart .

Several Determiners

⟦All (A,B)⟧=

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧= $\{X \subseteq U \mid A \subseteq X\}$

- ⟦Some NP⟧=

Several Determiners

⟦All (A,B)⟧= $1 \text{ iff } A \subseteq B$

⟦Some (A,B)⟧=

Some Brandeis students commute.

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧= $\{X \subseteq U \mid A \subseteq X\}$
- ⟦Some NP⟧= $\{X \subseteq U \mid X \cap A \neq \emptyset\}$
- ⟦No NP⟧=

Several Determiners

⟦All (A,B)⟧= $1 \text{ iff } A \subseteq B$

⟦Some (A,B)⟧= $1 \text{ iff } A \cap B \neq \emptyset$

⟦No (A,B)⟧=

No boy(s) came to the party.

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧= $\{X \subseteq U \mid A \subseteq X\}$
- ⟦Some NP⟧= $\{X \subseteq U \mid X \cap A \neq \emptyset\}$
- ⟦No NP⟧= $\{X \subseteq U \mid X \cap A = \emptyset\}$
- ⟦At least 5 NP⟧=

Several Determiners

⟦All (A,B)⟧= $1 \text{ iff } A \subseteq B$

⟦Some (A,B)⟧= $1 \text{ iff } A \cap B \neq \emptyset$

⟦No (A,B)⟧= $1 \text{ iff } A \cap B = \emptyset$

⟦At least 5 (A,B)⟧=

At least 5 ballerinas danced there.

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧= $\{X \subseteq U \mid A \subseteq X\}$
- ⟦Some NP⟧= $\{X \subseteq U \mid X \cap A \neq \emptyset\}$
- ⟦No NP⟧= $\{X \subseteq U \mid X \cap A = \emptyset\}$
- [At least 5 NP]= $\{X \subseteq U \mid |X \cap A| \geq 5\}$
- ⟦Most NP⟧=

Several Determiners

⟦All (A,B)⟧= 1 iff $A \subseteq B$

⟦Some (A,B)⟧= 1 iff $A \cap B \neq \emptyset$

⟦No (A,B)⟧= 1 iff $A \cap B = \emptyset$

[At least 5 (A,B)]= 1 iff $|A \cap B| \geq 5$

⟦Most (A,B)⟧=

Most Brandeis students live on campus.

Generalised Quantifier Theory

Several GQs

- ⟦All NP⟧= $\{X \subseteq U \mid A \subseteq X\}$
- ⟦Some NP⟧= $\{X \subseteq U \mid X \cap A \neq \emptyset\}$
- ⟦No NP⟧= $\{X \subseteq U \mid X \cap A = \emptyset\}$
- **[**At least 5 NP**]**=$\{X \subseteq U \mid |X \cap A| \geq 5\}$
- **[**Most NP**]**= $\{X \subseteq U \mid |X \cap A| > 1/2|A|\}$
  $\{X \subseteq U \mid |X \cap A| > |X - A|\}$

Several Determiners

⟦All (A,B)⟧= $1 \text{ iff } A \subseteq B$

⟦Some (A,B)⟧= $1 \text{ iff } A \cap B \neq \emptyset$

⟦No (A,B)⟧= $1 \text{ iff } A \cap B = \emptyset$

[At least 5 (A,B)**]**=$1 \text{ iff } |A \cap B| \geq 5$

[Most (A,B)**]**= $1 \text{ iff } |A \cap B| > |A - B|$