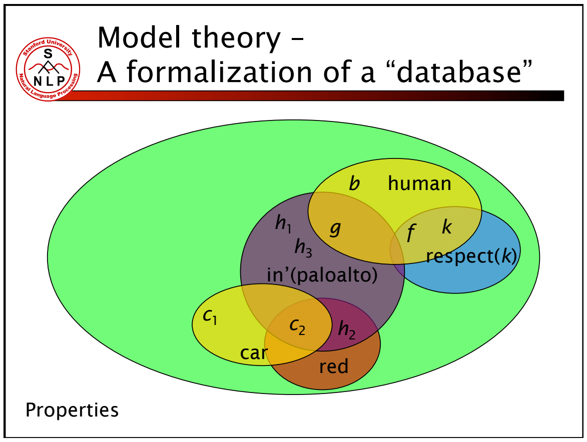
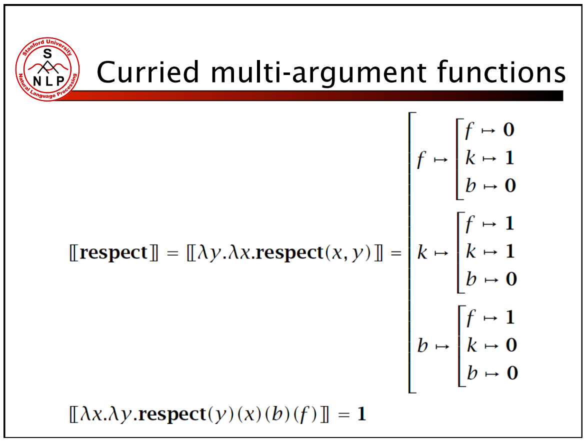


Model theory – A formalization of a "database"

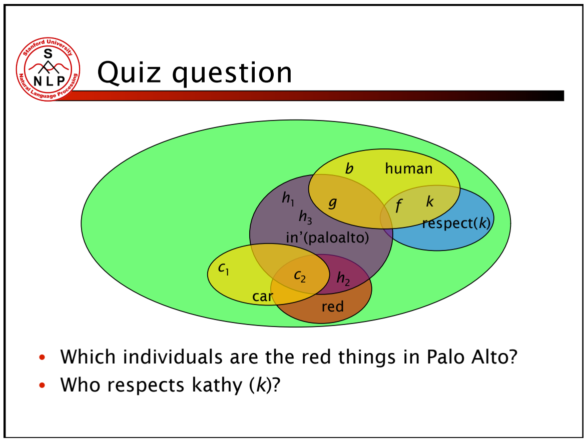
Properties



Curried multi-argument functions

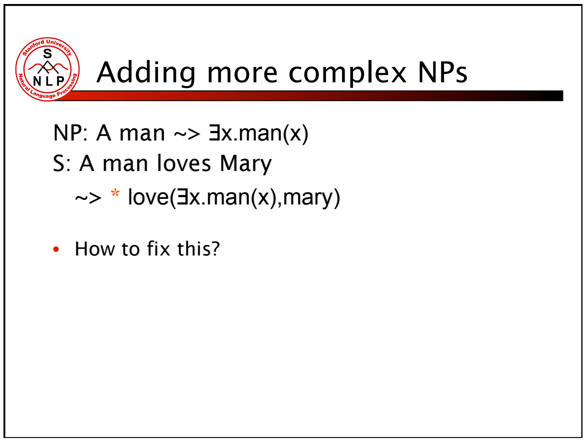
$$[\![\mathbf{respect}]\!] = [\![\lambda y.\lambda x.\mathbf{respect}(x,y)]\!] = \begin{bmatrix} f \mapsto \begin{bmatrix} f \mapsto \mathbf{0} \\ k \mapsto \mathbf{1} \\ b \mapsto \mathbf{0} \end{bmatrix} \\ k \mapsto \begin{bmatrix} f \mapsto \mathbf{1} \\ k \mapsto \mathbf{1} \\ b \mapsto \mathbf{0} \end{bmatrix} \\ b \mapsto \begin{bmatrix} f \mapsto \mathbf{1} \\ k \mapsto \mathbf{0} \\ b \mapsto \mathbf{0} \end{bmatrix} \end{bmatrix}$$

$$[\![\lambda x.\lambda y.\mathbf{respect}(y)(x)(b)(f)]\!] = \mathbf{1}$$



Quiz question

- Which individuals are the red things in Palo Alto?
- Who respects kathy (k)?



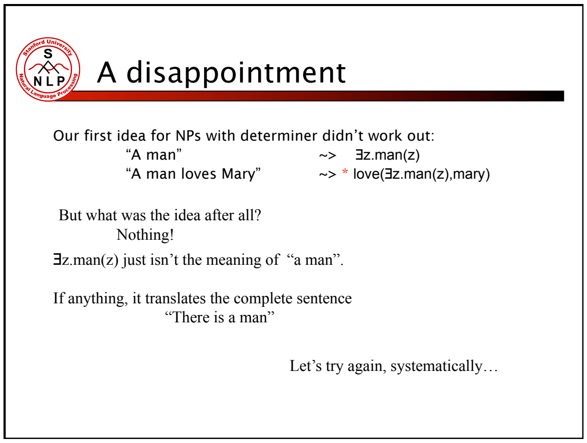
Adding more complex NPs

NP: A man ~> ∃x.man(x)

S: A man loves Mary

~> * love(∃x.man(x),mary)

- How to fix this?



A disappointment

Our first idea for NPs with determiner didn't work out:

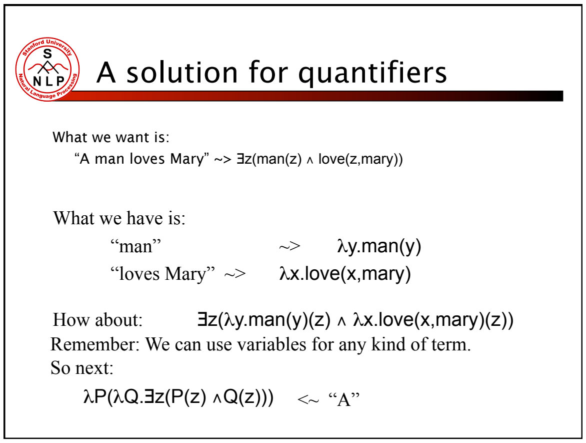
"A man"	~>	∃z.man(z)
"A man loves Mary"	~>	* love(∃z.man(z),mary)

But what was the idea after all?
Nothing!

∃z.man(z) just isn't the meaning of "a man".

If anything, it translates the complete sentence
"There is a man"

Let's try again, systematically…



A solution for quantifiers

What we want is:
"A man loves Mary" ~> ∃z(man(z) ∧ love(z,mary))

What we have is:

"man"	~>	λy.man(y)
"loves Mary"	~>	λx.love(x,mary)

How about: ∃z(λy.man(y)(z) ∧ λx.love(x,mary)(z))
Remember: We can use variables for any kind of term.
So next:

λP(λQ.∃z(P(z) ∧Q(z))) <~ "A"