Philosophy 240: Symbolic Logic Hamilton College
Fall 2008 Russell Marcus
Mondays, Wednesdays, Fridays: 9am - 9:50am rmarcus1@hamilton.edu

Class 38: Relational Predicates, Derivations (§8.6)

I. Deductions using Relational Predicates and Overlapping Quantifiers

Consider the argument with which we started studying relations:

Bob is taller than Charles.
Andrew is taller than Bob.
For any x, y and z, If x is taller than y and y is taller than z, then x is taller than z.
So, Andrew is taller than Charles.

1. Tbc
2. Tab
3. (x)(y)(z)[(Txy · Tyz) ⊃ Txz] / Tac

To derive the conclusion, we use the same rules of inference we used in monadic predicate logic.
Remove quantifiers one at a time.
Take care to make appropriate instantiations, to variables or constants.
There is only one exception, to UG, which we will note shortly.

1. Tbc	
2. Tab	
3. (x)(y)(z)[(Txy · Tyz) ⊃ Txz]	/ Tac
4. (y)(z)[(Tay · Tyz) ⊃ Taz]	3, UI
5. (z)[(Tab · Tbz) ⊃ Taz]	4, UI
6. (Tab · Tbc) ⊃ Tac	5, UI
7. (Tab · Tbc)	2, 1, Conj
8. Tac	6, 7, MP

QED

Here is another example, in which we take off the quantifiers in the middle of the proof, rather than at the beginning.

1. (∃x)[Hx · (y)(Hy ⊃ Lyx)]	/ (∃x)(Hx · Lxx)
2. Ha · (y)(Hy ⊃ Lya)	1, EI
3. Ha	2, Simp
4. (y)(Hy ⊃ Lya)	2, Com, Simp
5. Ha ⊃ Laa	4, UI
6. Laa	5, 3, MP
7. Ha · Laa	3, 6, Conj
8. (∃x)(Hx · Lxx)	7, EG

QED

II. **The Restriction on UG**:

Consider the following derivation, beginning with a proposition that can be interpreted as ‘Everything loves something’.

1. (x)(∃y)Lxy		
2. (∃y)Lxy	1, UI	
3. Lxa	2, EI	
4. (x)Lxa	3, UG	Note: this step is incorrect!
5. (∃y)(x)Lxy	4, EG	

Given our interpretation of line 1, line 5 reads, ‘There’s something that everything loves’.
It does not follow from the proposition that everything loves something that there is one thing that everything loves.
Imagine we ordered all the things in a circle, and everyone loved just the thing to its left.
Line 1 would be true, but line 5 would be false.
So, we should not be able to derive step 5 from step 1.

We can locate the problem in step 4 of the above derivation.
In line 2 we universally instantiated to some random object x.
So ‘x’ could have stood for any object.
It retained its universal character, even without a universal quantifier to bind it.
But, in line 3, when we existentially instantiated, we gave a name, ‘a’ to the thing which bore relation L to it, to the thing x loves.
Once we gave a name to the thing that x loves, x lost its universal character.
It could no longer be anything, which loves something.
It now had to be the thing that loves a.
‘x’ became as particular an object as ‘a’ is.
So, the generalization at line 4 must be blocked.
Variables lose their universal character if they are free when EI is used.
So, You may never UG on a variable when there’s a constant present, and the variable was free when the constant was introduced.
I.e. We can not UG In line 4, because ‘x’ was free in line 3 when ‘a’ was introduced.

Here is a derivation with an acceptable use of UG:

1. (∃x)(y)[(∃z)Ayz ⊃ Ayx]	
2. (y)(∃z)Ayz	/ (∃x)(y)Ayx
3. (y)[(∃z)Ayz ⊃ Aya]	1, EI
4. (∃z)Ayz ⊃ Aya	3, UI
5. (∃z)Ayz	2, UI
6. Aya	4, 5, MP
7. (y)Aya	6, UG
8. (∃x)(y)Ayx	7, EG

QED

Note that at line 7, UG is acceptable because ‘y’ was not free when ‘a’ was introduced in line3!
This restriction only applies to UG; all other rules are just as they are in monadic predicate logic.

III. More examples, and a warning about accidental binding

When quantifying, i.e. when using UG or EG, watch for accidental binding.
Consider :

(Pa · Qa) ⊃ (Fx ∨ Gx)

If you try to quantify over the ‘a’ using EG with the variable ‘x’, you accidentally bind the latter two terms:

(∃x)[(Px · Qx) ⊃ (Fx ∨ Gx)]

That is a legitimate use of existential generalization, but it may not mean what you want.
Instead, use a ‘y’:

(∃y)[(Py · Qy) ⊃ (Fx ∨ Gx)]

Now the latter terms remain free.
We can still bind them with either a universal quantifier or an existential quantifier, later.

(x)(∃y)[(Py · Qy) ⊃ (Fx ∨ Gx)]
(∃x)(∃y)[(Py · Qy) ⊃ (Fx ∨ Gx)]

Here is a conditional proof using relational predicates:

1. (x)[Ax ⊃ (y)Bxy]	
2. (x)[Ax ⊃ (∃y)Dyx]	/(x)[Ax ⊃ (∃y)(Bxy · Dyx)]
	3. Ax	ACP
	4. Ax ⊃ (y)By	1, UI
	5. Ax ⊃ (∃y)Dyx	2, UI
	6. (y)Bxy	4, 3, MP
	7. (∃y)Dyx	5, 3 MP
	8. Dax	7, EI
	9. Bxa	6, UI
	10. Bxa · Dax	9, 8, Conj
	11. (∃y)(Bxy • Dyx)	10, EG
12. Ax ⊃ (∃y)(Bxy · Dyx)	3-11, CP
13. (x)[Ax ⊃ (∃y)(Bxy · Dyx)]	12, UG

QED

Here is a more complex proof:

1. (x)(Wx ⊃ Xx)	
2. (x)[(Yx · Xx) ⊃ Zx]	
3. (x)(∃y)(Yy · Ayx)	
4. (x)(y)[(Ayx · Zy) ⊃ Zx]	/(x)[(y)(Ayx ⊃ Wy) ⊃ Zx]
│5. (y)(Ayx ⊃ Wy)	ACP
│6. (∃y)(Yy · Ayx)	3, UI
│7. Ya · Aax	6, EI
│8. Aax ⊃ Wa	5, UI
│9. Aax	7, Com, Simp
│10. Wa	8,9, MP
│11. Wa ⊃ Xa	1, UI
│12. Xa	11, 10, MP
│13. Ya	7, Simp
│14. Ya · Xa	13, 12, Conj
│15. (Ya · Xa) ⊃ Za	2, UI
│16. Za	15, 14, MP
│17. (y)[(Ayx · Zy) ⊃ Zx]	4, UI
│18. (Aax · Za) ⊃ Zx	17, UI
│19. Aax · Za	9, 16, Conj
│20. Zx	18, 19, MP
21. (y)(Ayx ⊃ Wy) ⊃ Zx	5-20, CP
22. (x)[(y)(Ayx ⊃ Wy) ⊃ Zx]	21, UG

QED

Notes:
At line 16, you might be tempted to discharge your assumption and finish your CP.
But, you wouldn't be able to UG over the 'Za'.
We have to UI at line 17, retaining a variable for the predicate 'Z'.

IV. **Exercises**. Derive the conclusions of each of the following arguments.

1.
 1. (x)(Cax ⊃ Dxb)
 2. (∃x)Dxb ⊃ (∃y)Dby / (∃x)Cax ⊃ (∃y)Dby

2.
 1. (x)[Ex ⊃ (y)(Fy ⊃ Gxy)]
 2. (∃x)[Ex · (∃y)~Gxy] / (∃x)~Fx

3.
 1. (∃x)Ax ⊃ (∃x)Bx / (∃y)(x)(Ax ⊃ By)

4.
 1. (x)[Mx ⊃ (y)(Ny ⊃ Oxy)]
 2. (x)[Px ⊃ (y)(Oxy ⊃ Qy)] / (∃x)(Mx · Px) ⊃ (y)(Ny ⊃ Qy)