Quantificational Logic, part 2

Review of Part 1

			Valid
	1	(x)(Fx · Gx)	
		[∴ (x)Fx	
*	2	┌ asm: ~(x)Fx	
*	3	│ ∴ (∃x)~Fx	{from 2}
	4	│ ∴ ~Fa	{from 3}
	5	│ ∴ (Fa · Ga)	{from 1}
	6	└ ∴ Fa	{from 5}
	7	∴ (x)Fx	{from 2; 4 contradicts 6}

			Invalid
	1	(x)(Lx ⊃ Fx)	
*	2	(∃x)Lx	a, b
		[∴ (x)Fx	
*	3	asm: ~(x)Fx	La, Fa ~Lb, ~Fb
*	4	∴ (∃x)~Fx	{from 3}
	5	∴ La	{from 2}
	6	∴ ~Fb	{from 4}
*	7	∴ (La ⊃ Fa)	{from 1}
*	8	∴ (Lb ⊃ Fb)	{from 1}
	9	∴ Fa	{from 5 and 7}
	10	∴ ~Lb	{from 6 and 8}

1. Reverse squiggles.
2. Drop initial existentials, using a new letter each time.
3. Lastly, drop initial universals, using all the old letters. (Only use a new letter if you've done everything else possible, including further assumptions if needed, and still have no old letters.)
4. If you can't get a contradiction, construct a refutation.

LogiCola I (EI & EC)

Pages 112 to 114

Identity Logic

$r=l$	=	Romeo is the lover of Juliet. (identity)
Ir	=	Romeo is Italian. (predication)
$(\exists x)Ix$	=	There are Italians. (existence)

The result of writing a small letter and then "=" and then a small letter is a wff.

LogiCola H (IM & IT)

Romeo isn't the lover of Juliet	=	$\sim r{=}l$
Someone besides Romeo is Italian Someone who isn't Romeo is Italian	=	$(\exists x)(\sim x{=}r \cdot Ix)$
Romeo alone is Italian Romeo is Italian but no one else is	=	$(Ir \cdot \sim(\exists x)(\sim x{=}r \cdot Ix))$

There is exactly one Italian	=	$(\exists x)(Ix \cdot \sim(\exists y)(\sim y{=}x \cdot Iy))$
There are at least two Italians	=	$(\exists x)(\exists y)((Ix \cdot Iy) \cdot \sim x{=}y)$
There are exactly two Italians	=	$(\exists x)(\exists y)(((Ix \cdot Iy) \cdot \sim x{=}y) \cdot$ $\sim(\exists z)((\sim z{=}x \cdot \sim z{=}y) \cdot Iz))$

$1 + 1 = 2$

If exactly one being is F and exactly one being is G and nothing is F-and-G, then exactly two beings are F-or-G.

$$((((\exists x)(Fx \cdot \sim(\exists y)(\sim y{=}x \cdot Fy)) \\ \cdot (\exists x)(Gx \cdot \sim(\exists y)(\sim y{=}x \cdot Gy))) \\ \cdot \sim(\exists x)(Fx \cdot Gx)) \supset \\ (\exists x)(\exists y)(((Fx \vee Gx) \cdot (Fy \vee Gy)) \cdot (\sim x{=}y \\ \cdot \sim(\exists z)((\sim z{=}x \cdot \sim z{=}y) \cdot (Fz \vee Gz)))))$$

Identity Principles

Self-identity axiom	$a=a$
Substitute-equals rule	$Fa,\ a=b \quad \rightarrow \quad Fb$

Can also use: $a = b,\ b = c \rightarrow a = c$

$a = b \rightarrow b = a$

There's more than one being. (pluralism)

∴ It's false that there's exactly one being. (monism)

*	1	$(\exists x)(\exists y)\sim x=y$	Valid
		$[\therefore \sim(\exists x)(y)y=x$	
*	2	asm: $(\exists x)(y)y=x$	
*	3	$\therefore(\exists y)\sim a=y$ {from 1}	
	4	$\therefore \sim a=b$ {from 3}	
	5	$\therefore(y)y=c$ {from 2}	
	6	$\therefore a=c$ {from 5}	
	7	$\therefore b=c$ {from 5}	
	8	$\therefore a=b$ {from 6 and 7}	
	9	$\therefore \sim(\exists x)(y)y=x$ {from 2; 4 contradicts 8}	

LogiCola I DV

LogiCola Set I (DV) - Score (level 8) = 0

File Options Tools Help

	1	Jf
	2	s=f
	3	~Es
		[∴ ~(x)~(Jx • ~Ex)
	4	asm: (x)~(Jx • ~Ex)
	5	∴ ~Ef {from 2 and 3}
	6	∴ Js {from 1 and 2}
	7	∴ f=f {from 2 and 2}
	8	∴ s=s {from 2 and 7}
*	9	∴ ~(Jf • ~Ef) {from 4}
	10	∴ ~(Js • ~Es) {from 2 and 9}
	11	∴ Ef {from 1 and 9}
	12	∴ ~(x)~(Jx • ~Ex) {from 4; 5 contradicts 11}

Valid

This proof shows that the argument is valid.

LogiCola Set I (DI) - Score (level 8) = 0

File Options Tools Help

	1	Jk	= 1
*	2	~(∃x)~(~Ex ⊃ Jx)	= 1
	[∴	(~Ep ⊃ k=p)	= 0
*	3	asm: ~(~Ep ⊃ k=p)	
	4	∴ (x)(~Ex ⊃ Jx) {from 2}	
	5	∴ ~Ep {from 3}	
	6	∴ ~k=p {from 3}	
	7	∴ (~Ek ⊃ Jk) {from 4}	
*	8	∴ (~Ep ⊃ Jp) {from 4}	
	9	∴ Jp {from 5 and 8}	

REFUTE

Invalid

k, p

$$\boxed{\begin{array}{c}\text{Jk, ~k=p} \\ \text{~Ep, Jp}\end{array}}$$

These make premises true and conclusion false.

Relational Logic

Lrj	=	Romeo loves Juliet.
Bxyz	=	x is between y and z.

The result of writing a capital letter and then two or more small letters is a wff.

L(x) – property of x
Can be extended to relation…
L(r, j), B(x, y, z) are OK

L(juliet, romeo).

Juliet loves Romeo	=	Ljr
Juliet loves herself	=	Ljj
Juliet loves Romeo but not Paris	=	(Ljr • ~Ljp)
Juliet is between Paris and Romeo	=	Bjpr

Everyone loves him/herself	=	(x)Lxx
Someone loves himself	=	(∃x)Lxx
No one loves himself	=	~(∃x)Lxx

Put quantifiers *before* relations.	
Someone loves Juliet For some x, x loves Juliet $(\exists x)Lxj$	Juliet loves someone For some x, Juliet loves x $(\exists x)Ljx$

For some, use "·"
For all, use "⊃"

LogiCola H (RM & RT)

Everyone loves Juliet = $(x)Lxj$

For all x,	x loves Juliet

Juliet loves everyone = $(x)Ljx$

For all x,	Juliet loves x

No one loves Juliet = $\sim(\exists x)Lxj$

It is not the case that, for some x,	x loves Juliet

Juliet loves no one = $\sim(\exists x)Ljx$

It is not the case that, for some x,	Juliet loves x

Some Montague loves Juliet $=$ $(\exists x)(Mx \cdot Lxj)$

For some x,	x is a Montague and	x loves Juliet

Started with "All" => ⊃.

All Montagues love Juliet $=$ $(x)(Mx \supset Lxj)$

For all x,	if x is a Montague then	x loves Juliet

Romeo loves some Capulet $=$ $(\exists x)(Cx \cdot Lrx)$

For some x,	x is a Capulet and	Romeo loves x

case.

Romeo loves all Capulets $=$ $(x)(Cx \supset Lrx)$

For all x,	if x is a Capulet then	Romeo loves x

All Montagues love themselves = $(x)(Mx \supset Lxx)$

For all x,	x is a Montague then	x loves x

Some Montague besides Romeo loves Juliet = $(\exists x)((Mx \cdot \sim x=r) \cdot Lxj)$

For some x,	x is a Montague and x isn't Romeo and	x loves Juliet

Romeo loves all Capulets who love themselves = $(x)((Cx \cdot Lxx) \supset Lrx)$

For all x,	if x is a Capulet and x loves x then	Romeo loves x

These have two relations

All who know Juliet love Juliet = $(x)(Kxj \supset Lxj)$

For all x,	if x knows Juliet then	x loves Juliet

All who know themselves love themselves = $(x)(Kxx \supset Lxx)$

For all x,	if x knows x then	x loves x

These have two quantifiers

Someone loves someone = (∃x)(∃y)Lxy

For some x and for some y,	x loves y

Not the same person so two variables

Everyone loves everyone = (x)(y)Lxy

For all x and for all y,	x loves y

Everyone loves everyone else = (x)(y)(~x=y ⊃ Lxy)

For all x and for all y,	if x isn't y then	x loves y

Some Montague hates some Capulet = $(\exists x)(\exists y)((Mx \cdot Cy) \cdot Hxy)$

For some x and for some y,	x is a Montague and y is a Capulet and	x hates y

Every Montague hates every Capulet = $(x)(y)((Mx \cdot Cy) \supset Hxy)$

For all x and for all y,	if x is a Montague and y is a Capulet then	x hates y

Everyone loves someone.	There's someone who everyone loves.
For all x there's some y, such that x loves y.	There's some y such that, for all x, x loves y.
$(x)(\exists y)Lxy$	$(\exists y)(x)Lxy$

Variables are just dummy names
Variables should always be quantified
Order and scope of quantifiers are important

For all x, x is a C, there is M y that Lxy.

Every Capulet loves some Montague

$= \quad (x)(Cx \supset x \text{ loves some Montague})$

$= \quad (x)(Cx \supset (\exists y)(My \cdot Lxy))$

Every Capulet loves someone

$= \quad (x)(Cx \supset x \text{ loves someone})$

$= \quad (x)(Cx \supset (\exists y)Lxy)$

Everyone loves some Montague

$= \quad (x)\ x \text{ loves some Montague}$

$= \quad (x)(\exists y)(My \cdot Lxy)$

Some Capulet loves every Montague
= $(\exists x)(Cx \cdot$ x loves every Montague$)$
= $(\exists x)(Cx \cdot (y)(My \supset Lxy))$

Some Capulet loves everyone
= $(\exists x)(Cx \cdot$ x loves everyone$)$
= $(\exists x)(Cx \cdot (y)Lxy)$

Someone loves every Montague
= $(\exists x)$ x loves every Montague
= $(\exists x)(y)(My \supset Lxy)$

There is an unloved lover

= (∃x)(no one loves x • x loves someone)

= (∃x)(~(∃y)Lyx • (∃y)Lxy)

Note y scopes differently; can use y, z

Everyone loves every lover

= (x)(x loves someone ⊃ everyone loves x)

= (x)((∃y)Lxy ⊃ (y)Lyx)

Romeo loves all and only those who don't love themselves

= (x)(Romeo loves x ≡ x doesn't love x)

= (x)(Lrx ≡ ~Lxx)

All who know any person love that person

= (x)(y)(x knows y ⊃ x loves y)

= (x)(y)(Kxy ⊃ Lxy)

Reflexive / Irreflexive

Everyone loves himself	=	(x)Lxx
No one loves himself	=	(x)~Lxx

Symmetrical / Asymmetrical

Universally, if x loves y then	=	(x)(y)(Lxy ⊃ Lyx)
y loves x [does not love x]	=	(x)(y)(Lxy ⊃ ~Lyx)

Transitive / Intransitive

Universally, if x loves y and y loves	=	(x)(y)(z)((Lxy · Lyz) ⊃ Lxz)
z, then x loves z [does not love z]	=	(x)(y)(z)((Lxy · Lyz) ⊃ ~Lxz)

1 (x)Lxx Valid

[∴ (x)(∃y)Lxy

* 2 ┌ asm: ~(x)(∃y)Lxy

* 3 │ ∴ (∃x)~(∃y)Lxy {from 2}

* 4 │ ∴ ~(∃y)Lay {from 3}

5 │ ∴ (y)~Lay {from 4}

6 │ ∴ ~Laa {from 5}

7 └ ∴ Laa {from 1}

8 ∴ (x)(∃y)Lxy {from 2; 4 contradicts 6}

LogiCola I (RV & BV)

> Relational proofs are often tricky, even though they use no new inference rules. When you have a string of quantifiers, as in lines 2 and 3 above, work on one at a time, starting from the outside. Drop only *initial* quantifiers!

	1	(x)Lxx	
		[∴ (∃x)(y)Lyx	
*	2	asm: ~(∃x)(y)Lyx	
	3	∴ (x)~(y)Lyx	{from 2}
	4	∴ Laa	{from 1}
*	5	∴ ~(y)Lya	{from 3}
*	6	∴ (∃y)~Lya	{from 5}
	7	∴ ~Lba	{from 6}
	8	∴ Lbb	{from 1}
*	9	∴ ~(y)Lyb	{from 3}
	10	∴ (∃y)~Lyb	{from 9} . . .

Invalid

a, b

Laa, Lbb
~Lba, ~Lab

➔ get c, d, . . .

LogiCola I (RI)

> If you see an infinite loop coming, break out of it and invent your own refutation.

Endless Loops

Since everyone loves someone	a loves someone, call this person b b loves someone, call this person c c loves someone, call this person d …

$(x)(\exists y)Lxy$	$(\exists y)Lay \Rightarrow Lab$ $(\exists y)Lby \Rightarrow Lbc$ $(\exists y)Lcy \Rightarrow Lcd$ …

Pages 143 to 144

Alonzo Church's Theorem (1931)

The problem of determining validity in relational logic cannot be reduced to an algorithm (a finite mechanical procedure).

Pages 143 to 144