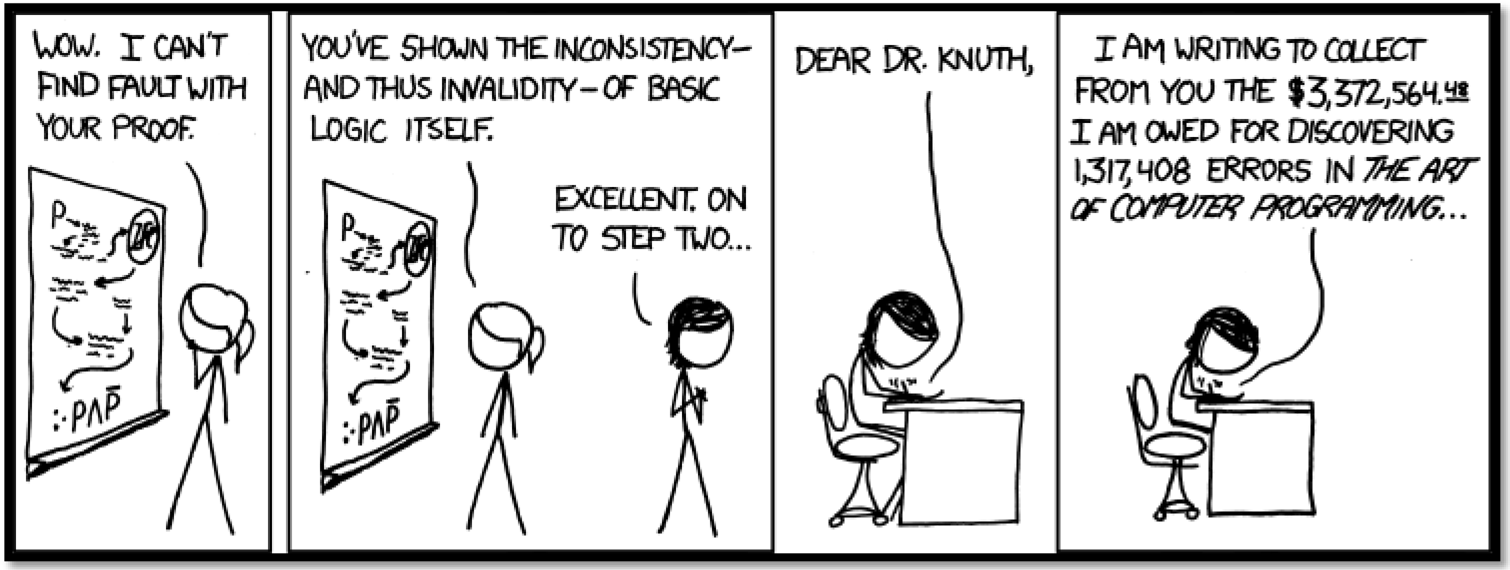
Fall 2015

Lecture 6: Predicate Logic, Logical Inference



quantifiers

$\forall x\ P(x)$

P(x) is true for every x in the domain

read as "for all x, P of x"

$\exists x\ P(x)$

There is an x in the domain for which P(x) is true

read as "there exists x, P of x"

- not every positive integer is prime

$$\neg \forall x \, \mathrm{Prime}(x) \equiv \exists x \, \neg \mathrm{Prime}(x)$$

- some positive integer is not prime

$$\exists x \, \neg \mathrm{Prime}(x)$$

- prime numbers do not exist

$$\forall x \, \neg \mathrm{Prime}(x) \equiv \neg \exists x \, \mathrm{Prime}(x)$$

- every positive integer is not prime

$$\forall x \, \neg \mathrm{Prime}(x)$$

negations of quantifiers

$\forall x$ PurpleFruit(x)

Domain:
Fruit

PurpleFruit(x)

Which one is equal to $\neg\forall x$ PurpleFruit(x)?

- $\exists x$ PurpleFruit(x)?
- $\exists x\ \neg$PurpleFruit(x)?

de Morgan's laws for quantifiers

$\left[\forall x \, (\text{PayingAtt}(x) \to \text{MakesSense}(x)) \right]$

$$\boxed{\begin{array}{l} \neg\forall x \; P(x) \equiv \exists x \, \neg P(x) \\ \neg\exists x \; P(x) \equiv \forall x \, \neg P(x) \end{array}}$$

$D = \{x_1, x_2, x_3, \ldots\}$

$\forall x \, P(x) \equiv P(x_1) \wedge P(x_2) \wedge P(x_3) \wedge \ldots$

$\exists x \, P(x) \equiv P(x_1) \vee P(x_2) \vee P(x_3) \vee \ldots$

$\neg\forall x \, P(x) \equiv \neg(P(x_1) \wedge P(x_2) \wedge \ldots)$

$\equiv \neg P(x_1) \vee \neg P(x_2) \vee \ldots$

$\equiv \exists x \, \neg P(x)$

de Morgan's laws for quantifiers

$$\boxed{\begin{aligned} \neg\forall x\ P(x) &\equiv \exists x\, \neg P(x) \\ \neg\exists x\ P(x) &\equiv \forall x\, \neg P(x) \end{aligned}}$$

"There is no largest integer."

$$\begin{aligned} &\neg\exists x\ \forall y\ (x \geq y) \\ \equiv\ &\forall x\ \neg\forall y\ (x \geq y) \\ \equiv\ &\forall x\ \exists y\ \neg(x \geq y) \\ \equiv\ &\forall x\ \exists y\ (y > x) \end{aligned}$$

"For every integer there is a larger integer."

scope of quantifiers

$\forall x \, \exists y \, (y \geq x)$

example: Notlargest(x) $\equiv \exists$ y Greater (y, x)

$\equiv \exists$ z Greater (z, x)

truth value:

doesn't depend on y or z "bound variables"

does depend on x "free variable"

quantifiers only act on free variables of the formula they quantify

$\forall x \, (\exists y \, (P(x, y) \rightarrow \forall x \, Q(y, x)))$

$\equiv \forall x \, (\exists y \, P(x, y) \rightarrow \forall z \, Q(y, z))$

example:

Domain = positive integers

$\text{IsMultiple}(x, y) =$ "x is a multiple of y"

$$\forall x \left((x > 1 \land \neg(x = y)) \to \neg\text{IsMultiple}(y, x) \right)$$

$$\equiv \text{Prime}(y)$$

$$\forall x\, \exists y \left((x < y) \land \text{Prime}(y) \right)$$

$$\forall x\, \exists y \left((x < y) \land \left(\forall x \left((x > 1 \land \neg(x = y)) \to \neg\text{IsMultiple}(y, x) \right) \right) \right)$$

scope of quantifiers

example:

Domain = positive integers

$\mathrm{IsMultiple}(x, y) =$ “x is a multiple of y”

$$\forall x\ \big((x > 1 \wedge \neg(x = y)) \rightarrow \neg \mathrm{IsMultiple}(y, x)\big)$$

$$\equiv \mathrm{Prime}(y)$$

$$\forall x\ \exists y\ \big((x < y) \wedge \mathrm{Prime}(y) \wedge \mathrm{Prime}(y + 2)\big)$$

$$\forall x\ \exists y \left(\begin{array}{c} (x < y) \wedge \Big(\forall x\ \big((x > 1 \wedge \neg(x = y)) \rightarrow \neg \mathrm{IsMultiple}(y, x)\big)\Big) \\ \wedge \Big(\forall x\ \big((x > 1 \wedge \neg(x = \overset{y+2}{\cancel{y}})) \rightarrow \neg \mathrm{IsMultiple}(\overset{y+2}{\cancel{y}}, x)\big)\Big) \end{array} \right)$$

example:

Domain = positive integers

$\text{IsMultiple}(x, y) =$ "x is a multiple of y"

$$\forall x\,\big((x > 1 \land \neg(x = y)) \to \neg\text{IsMultiple}(y, x)\big)$$

$$\equiv \text{Prime}(y)$$

$$\forall x\,\exists y\,\big((x < y) \land \text{Prime}(y) \land \text{Prime}(y + 2) \land (x < y^2)\big)$$

$$\forall x\,\exists y \begin{pmatrix} (x < y) \land \Big(\forall x\,\big((x > 1 \land \neg(x = y)) \to \neg\text{IsMultiple}(y, x)\big)\Big) \\ \land \Big(\forall x\,\big((x > 1 \land \neg(x = y)) \to \neg\text{IsMultiple}(y, x)\big)\Big) \land (x < y^2) \end{pmatrix}$$

$$\underbrace{\exists x\ (P(x) \land Q(x))}_{F} \quad \text{vs.} \quad \underbrace{(\exists x\, P(x)) \land (\exists x\, Q(x))}_{T}$$

Domain = "sea creature"

P(x) = "x has fins"

Q(x) = "x has a shell"

✓

- Bound variable names don't matter

$$\forall x \, \exists y \, P(x, y) \equiv \forall a \, \exists b \, P(a, b)$$

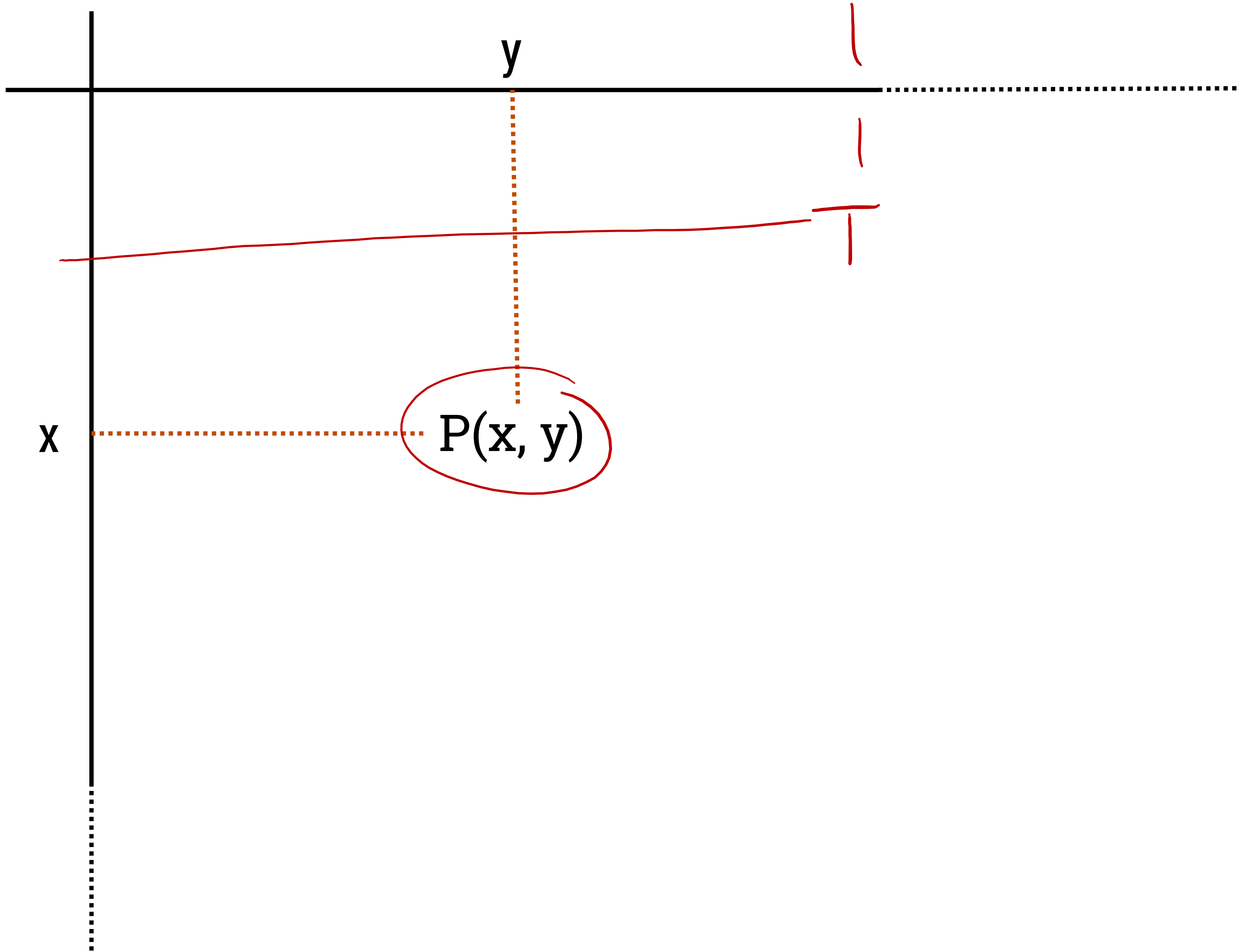
- Positions of quantifiers can sometimes change

$$\forall x \, (Q(x) \land \exists y \, P(x, y)) \equiv \forall x \, \exists y \, (Q(x) \land P(x, y))$$

- But: order is important...

$$\forall x \, \exists y \; P(x,y) \leftarrow \exists y \, \forall x \; P(x,y)$$

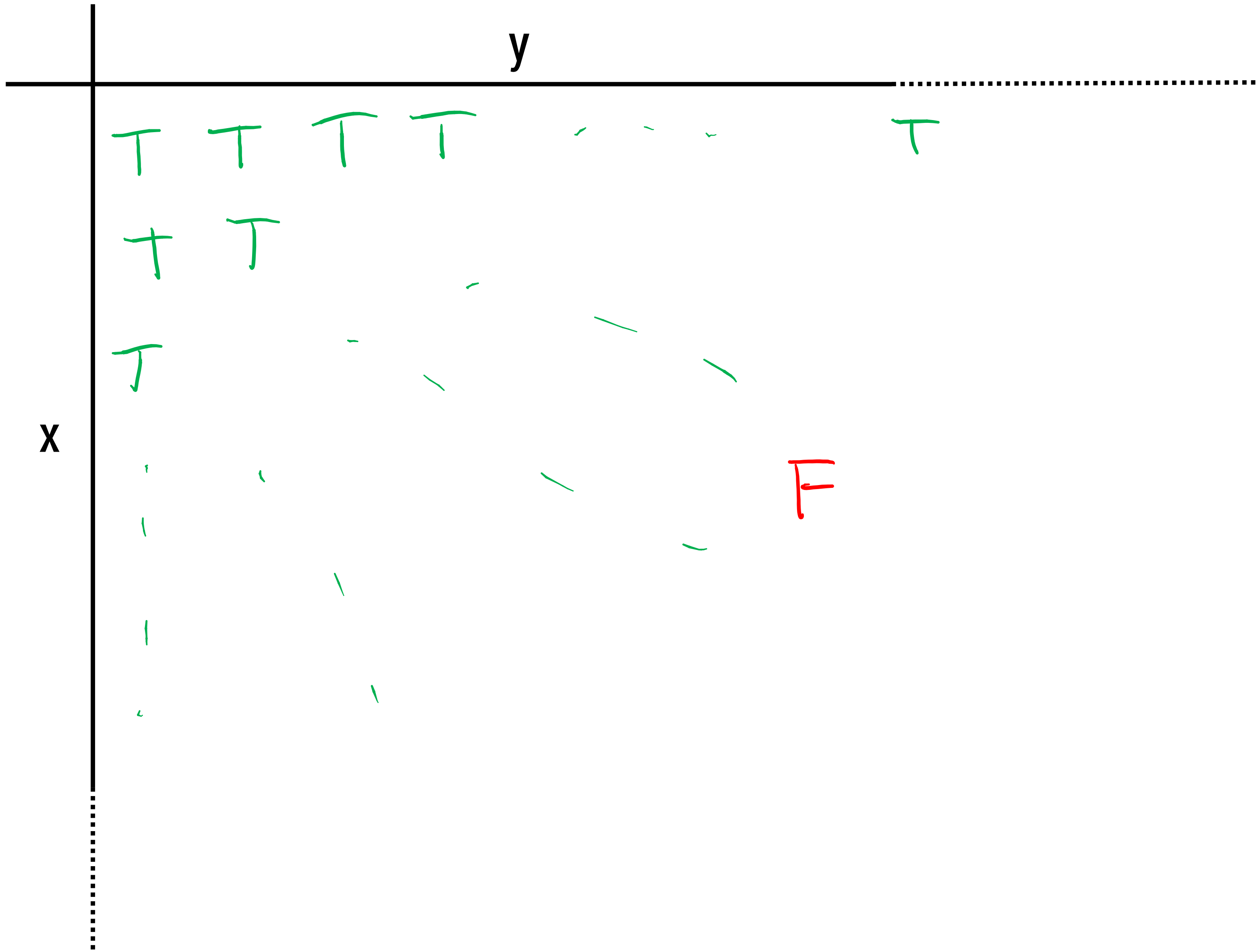
predicate with two variables



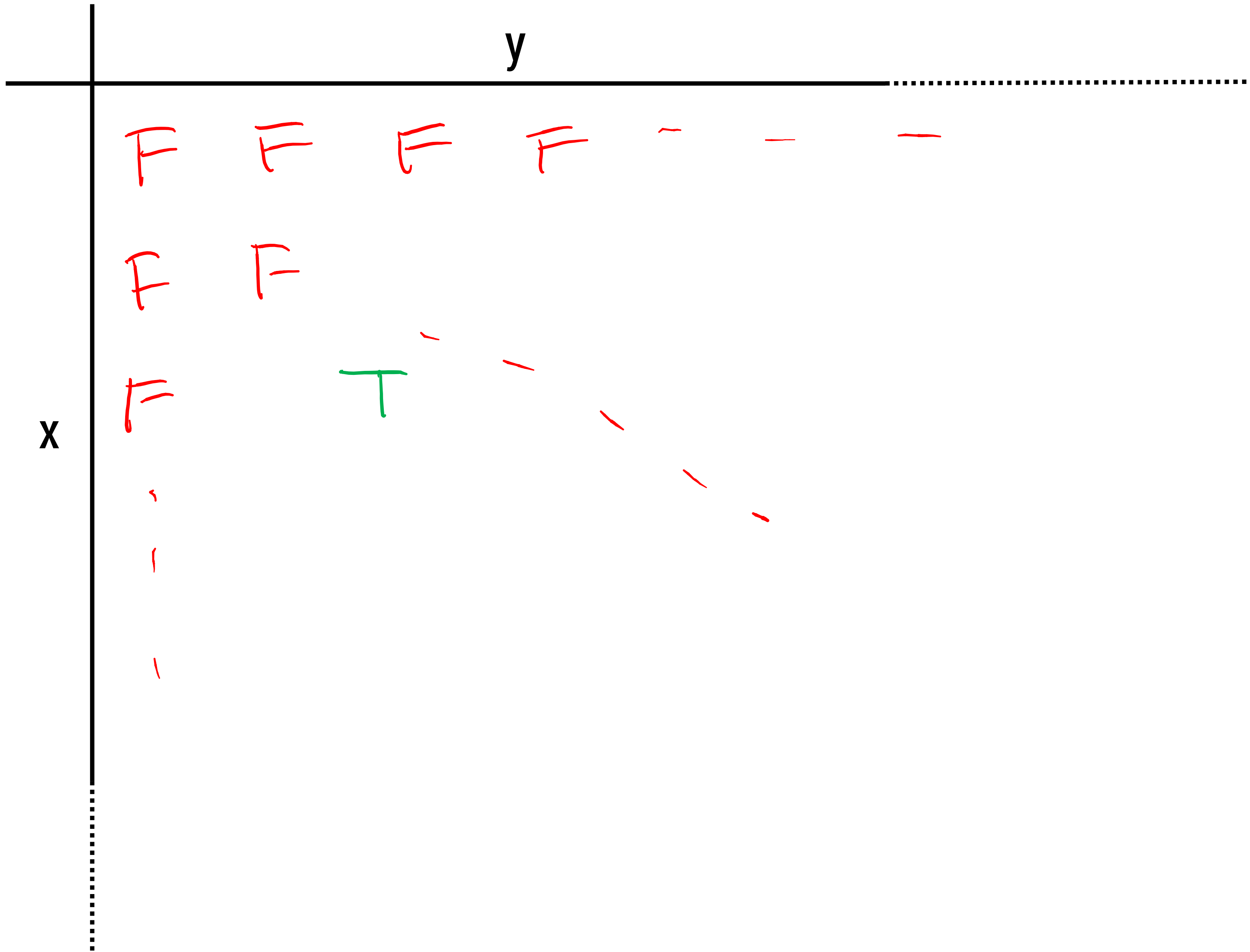
quantification with two variables

expression	**when true**	**when false**
$\forall x \forall y\, P(x, y)$		
$\exists x \exists y\, P(x, y)$		
$\forall x \exists y\, P(x, y)$		
$\exists x \forall y\, P(x, y)$		

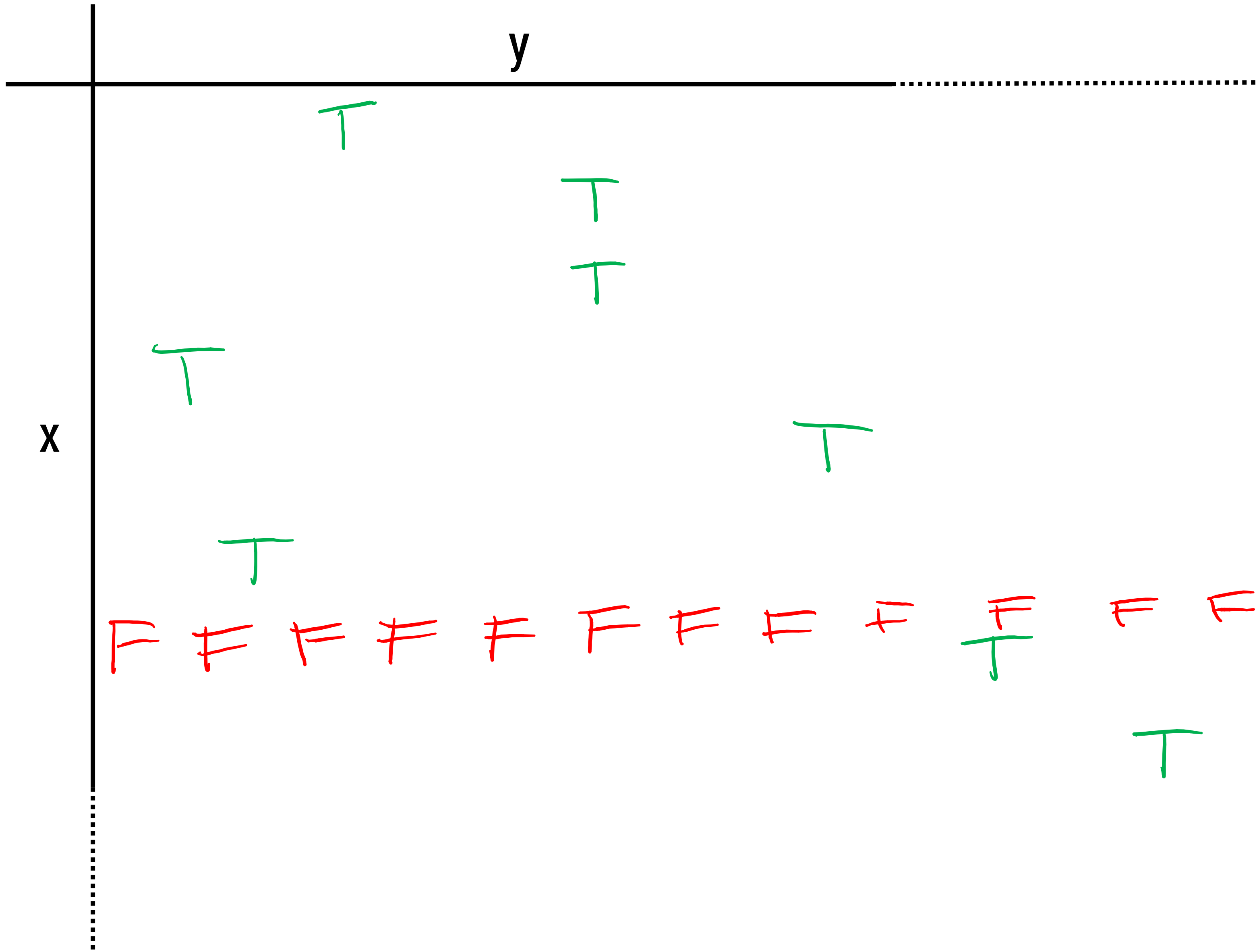
$\forall x \, \forall y \, P(x, y)$



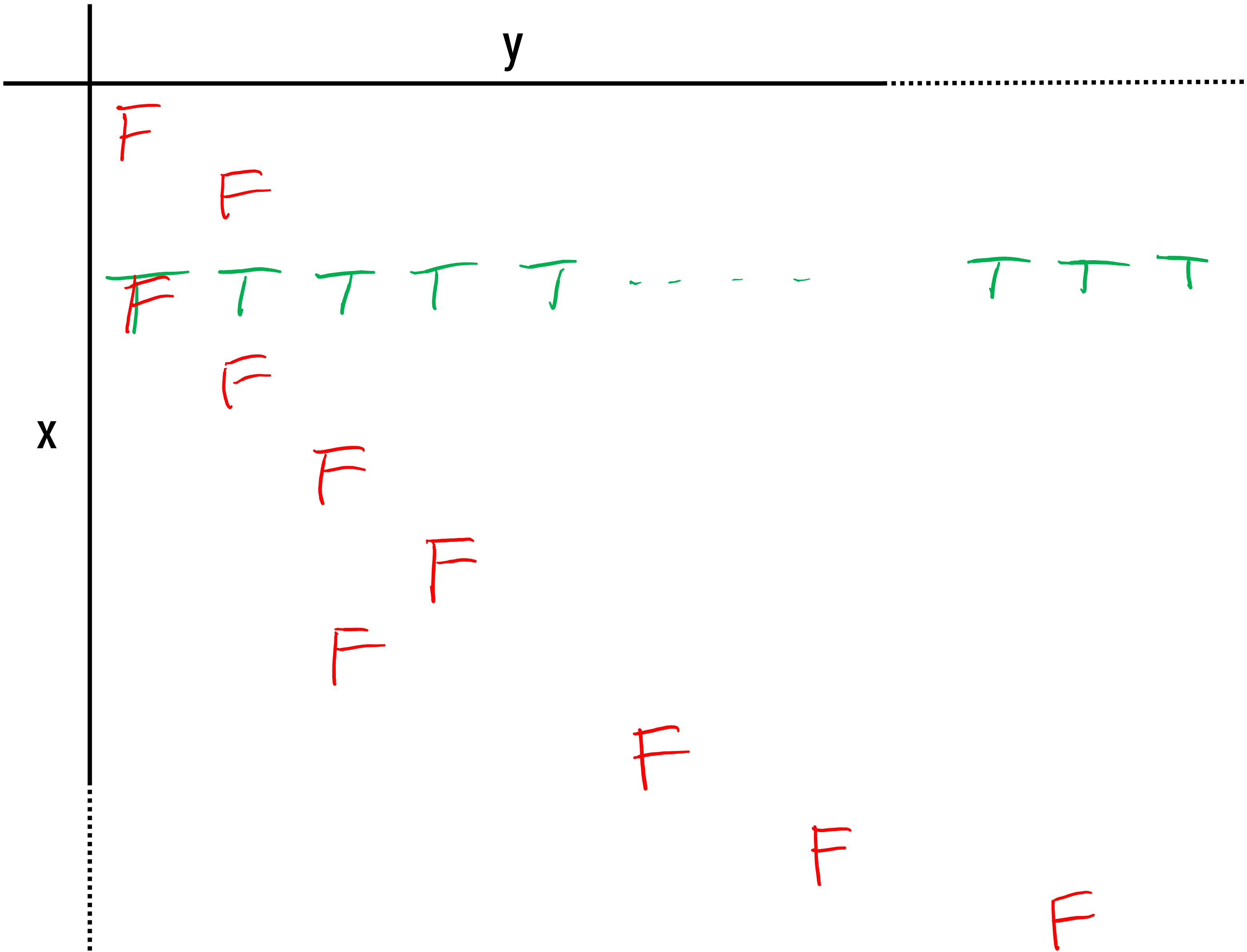
$$\exists x\,\exists y\,P(x,y)$$



$$\forall x\, \exists y\, P(x, y)$$



$\exists x\, \forall y\, P(x, y)$



quantification with two variables

expression	**when true**	**when false**
$\forall x \forall y P(x, y)$		
$\exists x \exists y P(x, y)$		
$\forall x \exists y P(x, y)$		
$\exists x \forall y P(x, y)$		

- So far we've considered:
  - How to understand and *express* things using propositional and predicate logic
  - How to *compute* using Boolean (propositional) logic
  - How to show that different ways of expressing or computing them are *equivalent* to each other
- Logic also has methods that let us *infer* implied properties from ones that we know
  - Equivalence is only a small part of this

applications of logical inference

- Software Engineering
  - Express desired properties of program as set of logical constraints
  - Use inference rules to show that program implies that those constraints are satisfied
- Artificial Intelligence
  - Automated reasoning
- Algorithm design and analysis
  - e.g., Correctness, Loop invariants.
- Logic Programming, e.g. Prolog
  - Express desired outcome as set of constraints
  - Automatically apply logic inference to derive solution

foundations of rational thought…

- Start with hypotheses and facts
- Use rules of inference to extend set of facts
- Result is proved when it is included in the set

an inference rule: *Modus Ponens*

- If p and p $\rightarrow$ q are both true then q must be true

- Write this rule as $$\frac{p,\ p \rightarrow q}{\therefore\ q}$$

- Given:
  - If it is Monday then you have a 311 class today.
  - It is Monday.

- Therefore, by modus ponens:
  - You have a 311 class today.

Show that r follows from p, $p \rightarrow q$, and $q \rightarrow r$

1.	p	given
2.	$p \rightarrow q$	given
3.	$q \rightarrow r$	given
4.	q	modus ponens from 1 and 2
5.	r	modus ponens from 3 and 4

proofs can use equivalences too

Show that $\neg p$ follows from $p \rightarrow q$ and $\neg q$

1.	$p \rightarrow q$	given
2.	$\neg q$	given
3.	$\neg q \rightarrow \neg p$	contrapositive of 1
4.	$\neg p$	modus ponens from 2 and 3

- Each inference rule is written as:

$$\frac{A,\ B}{\therefore\ C, D}$$

...which means that if both A and B are true then you can infer C and you can infer D.

  – For rule to be correct $(A \wedge B) \to C$ and $(A \wedge B) \to D$ must be a tautologies

- Sometimes rules don't need anything to start with. These rules are called axioms:
  – e.g. *Excluded Middle Axiom*

$$\frac{\qquad}{\therefore\ p \vee \neg p}$$

simple propositional inference rules

Excluded middle plus two inference rules per binary connective, one to eliminate it and one to introduce it:

$$\frac{p \wedge q}{\therefore p, q} \qquad\qquad \frac{p, q}{\therefore p \wedge q}$$

$\neg q \rightarrow p$

$$\frac{p \vee q \,,\, \neg p}{\therefore q} \qquad\qquad \frac{p}{\therefore p \vee q,\ q \vee p}$$

$$\frac{p,\ p \rightarrow q}{\therefore q} \qquad\qquad \frac{p \Rightarrow q}{\therefore p \rightarrow q}$$

Direct Proof Rule

Not like other rules

3. $a \rightarrow b$
4. $b \wedge c$

5. p

$\vdots$

10. q
11. $p \rightarrow q$

important: applications of inference rules

- You can use equivalences to make substitutions of any sub-formula.

4. $(p \to q) \to r$

5. $(\neg p \lor q) \to r$ ↳ L.o.I.

- Inference rules only can be applied to whole formulas (not correct otherwise)

0. p

1. $p \to q$

e.g.

~~1.~~	~~$p \to q$~~	~~given~~
~~2.~~	~~$(p \lor r) \to q$~~	~~intro $\lor$ from 1.~~

1. p
2. $p \lor r$ intro $\lor$

Does not follow! e.g. p=**F**, q=**F**, r=**T**

$p \to q$

$(p \lor \neg p) \to q$

q

direct proof of an implication

$p \to q$ $q \to r$

- $p \Rightarrow q$ denotes a proof of q given p as an assumption

- The direct proof rule:

  1 $p \to q$ Given

  If you have such a proof then you can conclude that $p \to q$ is true

2 $q \to r$ Given

Example:

proof subroutine

1. 3 p p assumption   Assumption
2. $p \vee q$ intro for $\vee$ from 1   MP
3. $p \to (p \vee q)$ direct proof rule

4 q MP

5. r

6 $p \to r$