Truth Tree to Fitch Converter

John (Jack) Cusick

Outline

- **Objective & Example**
- Relevant Software
- Detective Work
- Software Solution
- Demo
- Future Additions

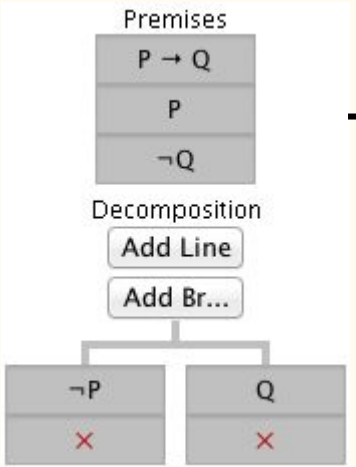
Objective

- Objectives:
  - Convert a truth tree to a formal Fitch proof.
  - Understand the file formatting of Fitch.
- Constraints:
  - Truth tree must be correct and represent a valid argument (all branches close).
  - No FOL truth trees.

Example Modus Ponens

$$\frac{P \rightarrow Q \quad P}{Q}$$

Truth Tree



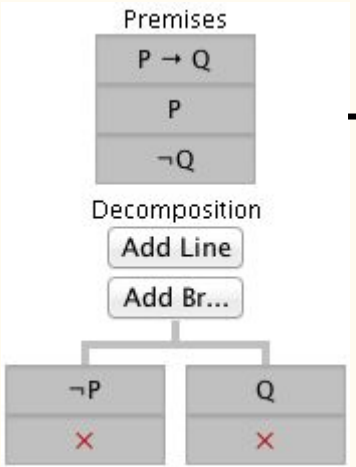
?

Fitch

Systematic Conversion

$$\frac{P \rightarrow Q \quad P}{Q}$$

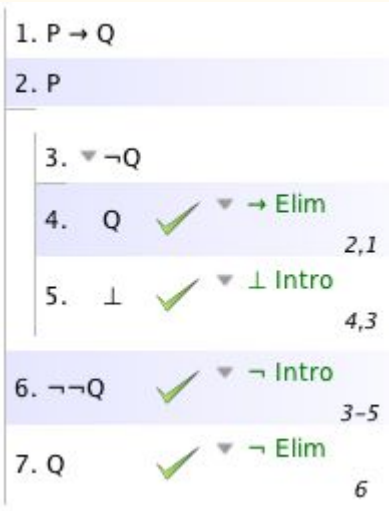
Truth Tree



1. Proof by Contradiction

Assume the negation of the conclusion.

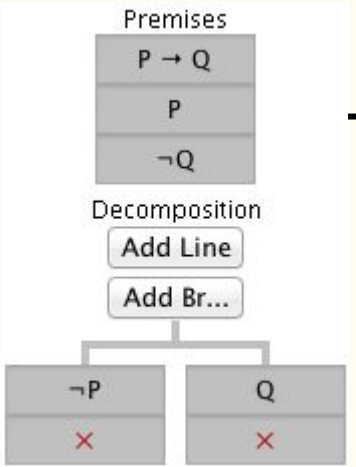
Fitch



Systematic Conversion

$$\frac{P \rightarrow Q \quad P}{Q}$$

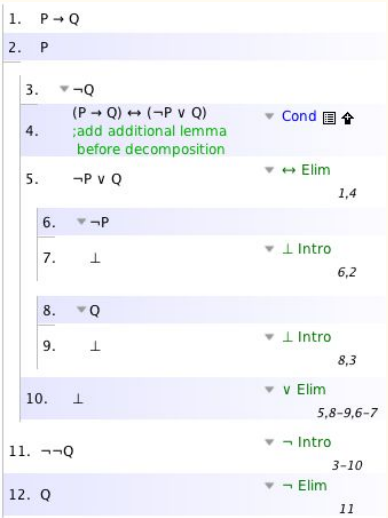
Truth Tree



2. Proof by Cases

Represent the branches as subproofs.

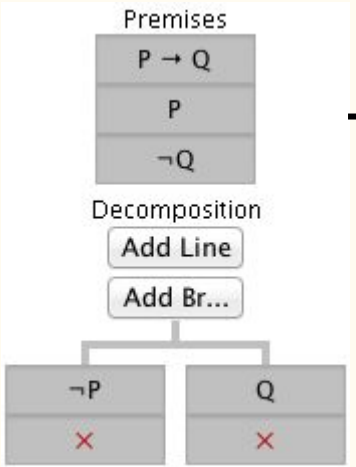
Fitch



Systematic Conversion

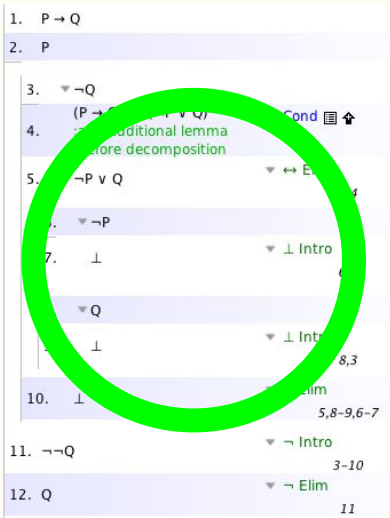
$$\frac{P \rightarrow Q \quad P}{Q}$$

Truth Tree



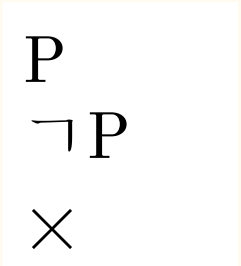
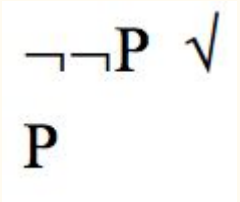
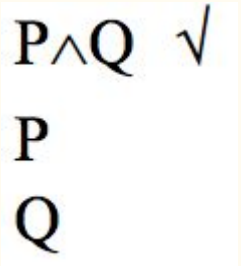
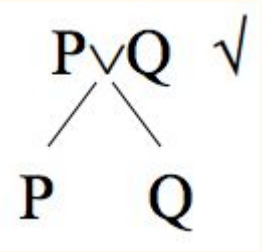
Proof by:
1. Contradiction
2. Cases

Fitch'



Fitch

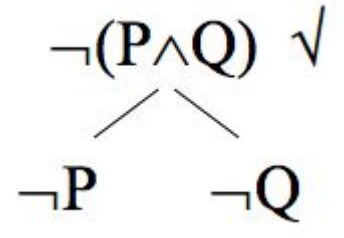
Decomposition Rules for Truth Trees

Fitch Equivalent:	$\perp$ Intro	$\neg$ Elim	x2 $\wedge$ Elim	x2 Subproofs $\vee$ Elim
Tree Decomposition:				

Fundamental Truth Tree Lemmas

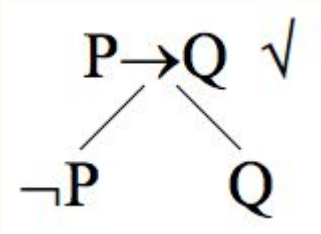
DeMorgan's 1

$\neg(P \wedge Q) \leftrightarrow (\neg P \vee \neg Q)$



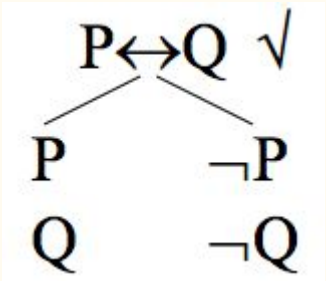
Conditional

$(P \rightarrow Q) \leftrightarrow (\neg P \vee Q)$



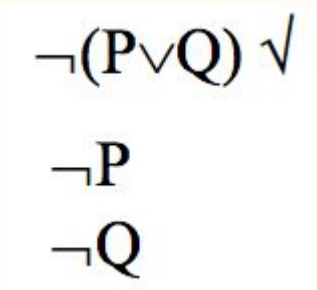
Biconditional

$(P \leftrightarrow Q) \leftrightarrow ((P \wedge Q) \vee (\neg P \wedge \neg Q))$



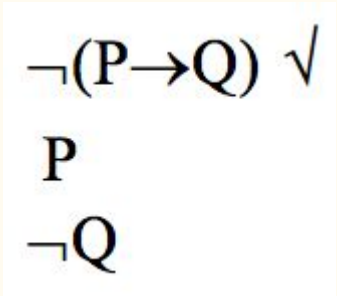
DeMorgan's 2

$\neg(P \vee Q) \leftrightarrow (\neg P \wedge \neg Q)$



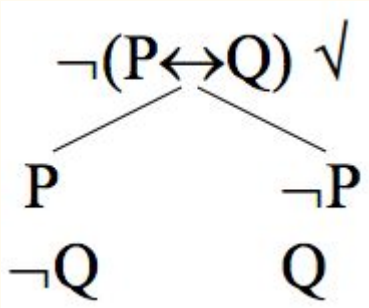
Negated Conditional

$\neg(P \rightarrow Q) \leftrightarrow (P \wedge \neg Q)$



Negated Biconditional

$\neg(P \leftrightarrow Q) \leftrightarrow ((P \wedge \neg Q) \vee (\neg P \wedge Q))$



Outline

- Objective & Example
- **Relevant Software**
- Detective Work
- Software Solution
- Demo
- Future Additions

Truth Trees

- Using Aaron Perl's Truth Tree Software[1].

- Files well organized in xml.
- Makes parsing easy.

Thanks Aaron!

Example File: Modus Ponens

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Tree>
    <BranchLine content="P → Q" index="0">
        <Decomposition branchIndex="0"/>
        <Decomposition lineIndex="3"/>
        <Decomposition lineIndex="5"/>
    </BranchLine>
    <BranchLine content="P" index="1"/>
    <BranchLine content="¬Q" index="2"/>
    <Branch index="0">
        <Branch index="1">
            <BranchLine content="¬P" index="3"/>
            <Terminator index="4">
                <Decomposition lineIndex="3"/>
                <Decomposition lineIndex="1"/>
            </Terminator>
        </Branch>
        <Branch index="1">
            <BranchLine content="Q" index="5"/>
            <Terminator index="6">
                <Decomposition lineIndex="5"/>
                <Decomposition lineIndex="2"/>
            </Terminator>
        </Branch>
    </Branch>
</Tree>
```

Formal Fitch Proofs

- Using Language Proof and Logic's [2] implementation of Fitch.

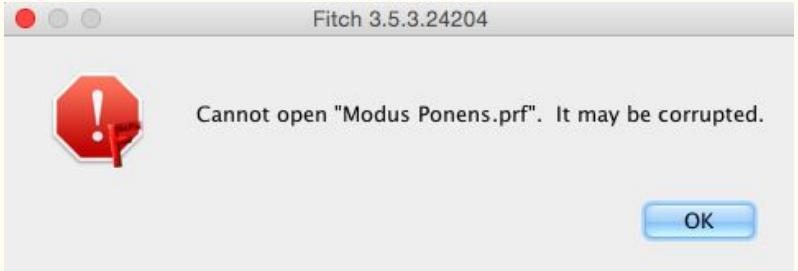
- Extremely cryptic files.
- Contains timestamps and hashes that prevent friendly interaction.

Example File: Modus Ponens

```
1  3.5.3.24204
2  macs:Mac OS X10.10.5
3  FchF
4  C1460473164554D1460473257376D1460473926984C1460475410004D1460475422199
5  newFormat
6  =openproof.zen.Openproof{p=openproof.fitch.FitchProofDriver{p=openproof.
   proofdriver.DRProof{s(openproof.proofdriver.DRStepInfo=openproof.
   proofdriver.DRStepInfo{r&1;})r=openproof.proofdriver.
   DRProofRule{u=uProof;s=step;}o=openproof.zen.proofdriver.
   OPDStatusObject{c=1;s="";l="";d@k="";t=false;}u=openproof.proofdriver.
   DRSupport{t()}b()f(openproof.proofdriver.DRSimpleStep=openproof.
   proofdriver.DRSimpleStep{s(openproof.proofdriver.DRStepInfo=openproof.
   proofdriver.DRStepInfo{r=openproof.foldriver.FOLDriver{t="P & Q";}})
   r=openproof.stepdriver.SRPremiseRule{u=uPremise;s=step;}o=openproof.zen.
   proofdriver.OPDStatusObject{c=1;s="";l="";d@k="";t=false;}u=openproof.
   proofdriver.DRSupport{t()}b()},openproof.proofdriver.
   DRSimpleStep=openproof.proofdriver.DRSimpleStep{s(openproof.proofdriver.
   DRStepInfo=openproof.proofdriver.DRStepInfo{r=openproof.foldriver.
   FOLDriver{t=P;}})r=openproof.fold.OPConjunctionElimRule{u="u\u2227
   Elim";s=fol;}o=openproof.fold.
   FOLRuleStatus{c=1;s="";l="";d@k="";t=false;f=1;}u=openproof.proofdriver.
   DRSupport{t(openproof.proofdriver.DRSupportPack=openproof.proofdriver.
   DRSupportPack{si&13;ss=0;sb=false;})}b()},openproof.proofdriver.
   DRSimpleStep=openproof.proofdriver.DRSimpleStep{s(openproof.proofdriver.
   DRStepInfo=openproof.proofdriver.DRStepInfo{r=openproof.foldriver.
   FOLDriver{t=Q;}})r=openproof.fold.OPConjunctionElimRule{u="u\u2227
   Elim";s=fol;}o=openproof.fold.
   FOLRuleStatus{c=1;s="";l="";d@k="";t=false;f=1;}u=openproof.proofdriver.
   DRSupport{t(openproof.proofdriver.DRSupportPack=openproof.proofdriver.
   DRSupportPack{si&13;ss=0;sb=false;})}b()})}g=openproof.proofdriver.
   DRGoalList{g()}a=true;}}c=163697;
7  s=234558;
```

Detective Work

- Changing anything:



- Decompiled
- Attached source to Fitch.jar in Eclipse
- Found logger, redirected to console
- Tried opening a corrupt proof and...

Eureka!

ERROR Fitch - File is corrupted?
Modus Ponens-corrupt.prf
Caused by: openproof.zen.exception.CheckSumException:
Fitch - **checksum does not match (350656 != 349185)**

- **Fitch log outputs the expected “password”!**

The Checksum Mystery Solved

- Fitch uses a type of checksum
- After much more testing...

```
DRSupportPack{si&13;ss=0;sb=false;})}b()})}g=openproof.proofdriver.
DRGoalList{g()}a=true;}]c=163697;
```

- c is the checksum: the sum of all characters' ASCII values on lines 1 - 6

```
4   C1460473164554D1460473257376D1460473926984C1460475410004D1460475422199
```

- You don't even need the long hash, you just get a "no timestamps" warning

S.R.O.U.B. and Sometimes F.

Line 6 stores the actual proof, and can be decomposed into the following:

- S: Step info
- R: Rule
- O: Object (class info)
- U: Support
- B: “b()”

```
openproof.proofdriver.DRSimpleStep=openproof.proofdriver.DRSimpleStep{
    s(
        openproof.proofdriver.DRStepInfo=openproof.proofdriver.DRStepInfo{
            r=openproof.foldriver.FOLDriver{t="~P & ~Q";}
        }
    )
    r=openproof.stepdriver.SRPremiseRule{u=uPremise;s=step;}
    o=openproof.zen.proofdriver.OPDStatusObject{c=1;s="";l="";d@k="";t=false;}
    u=openproof.proofdriver.DRSupport{t()}
    b()
},
```

When opening a new subproof…

- F: List of additional proof lines (S.R.O.U.B.)

Fitch File Format

Line 1: Fitch Version

Line 2: Operating System and Version

Line 3: “FchF”

Line 4: Time Stamps (can be blank)

Line 5: Fitch File Format (“newFormat”)

Line 6: Proof Data

- Class Calls -- S.R.O.U.B. and F.
- Stored Checksum c = …

Line 7: s = ...

Example File: Modus Ponens

```
1  3.5.3.24204
2  macs:Mac OS X10.10.5
3  FchF
4  C1460473164554D1460473257376D1460473926984C1460475410004D1460475422199
5  newFormat
6  =openproof.zen.Openproof{p=openproof.fitch.FitchProofDriver{p=openproof.
   proofdriver.DRProof{s(openproof.proofdriver.DRStepInfo=openproof.
   proofdriver.DRStepInfo{r&1;})r=openproof.proofdriver.
   DRProofRule{u=uProof;s=step;}o=openproof.zen.proofdriver.
   OPDStatusObject{c=1;s="";l="";d@k="";t=false;}u=openproof.proofdriver.
   DRSupport{t()}b()f(openproof.proofdriver.DRSimpleStep=openproof.
   proofdriver.DRSimpleStep{s(openproof.proofdriver.DRStepInfo=openproof.
   proofdriver.DRStepInfo{r=openproof.foldriver.FOLDriver{t="P & Q";}})
   r=openproof.stepdriver.SRPremiseRule{u=uPremise;s=step;}o=openproof.zen.
   proofdriver.OPDStatusObject{c=1;s="";l="";d@k="";t=false;}u=openproof.
   proofdriver.DRSupport{t()}b()},openproof.proofdriver.
   DRSimpleStep=openproof.proofdriver.DRSimpleStep{s(openproof.proofdriver.
   DRStepInfo=openproof.proofdriver.DRStepInfo{r=openproof.foldriver.
   FOLDriver{t=P;}})r=openproof.fold.OPConjunctionElimRule{u="u\u2227
   Elim";s=fol;}o=openproof.fold.
   FOLRuleStatus{c=1;s="";l="";d@k="";t=false;f=1;}u=openproof.proofdriver.
   DRSupport{t(openproof.proofdriver.DRSupportPack=openproof.proofdriver.
   DRSupportPack{si&13;ss=0;sb=false;})}b()},openproof.proofdriver.
   DRSimpleStep=openproof.proofdriver.DRSimpleStep{s(openproof.proofdriver.
   DRStepInfo=openproof.proofdriver.DRStepInfo{r=openproof.foldriver.
   FOLDriver{t=Q;}})r=openproof.fold.OPConjunctionElimRule{u="u\u2227
   Elim";s=fol;}o=openproof.fold.
   FOLRuleStatus{c=1;s="";l="";d@k="";t=false;f=1;}u=openproof.proofdriver.
   DRSupport{t(openproof.proofdriver.DRSupportPack=openproof.proofdriver.
   DRSupportPack{si&13;ss=0;sb=false;})}b()})}g=openproof.proofdriver.
   DRGoalList{g()}a=true;}}c=163697;
7  s=234558;
```

Outline

- Objective & Example
- Relevant Software
- Detective Work
- **Software Solution**
- Demo
- Future Additions

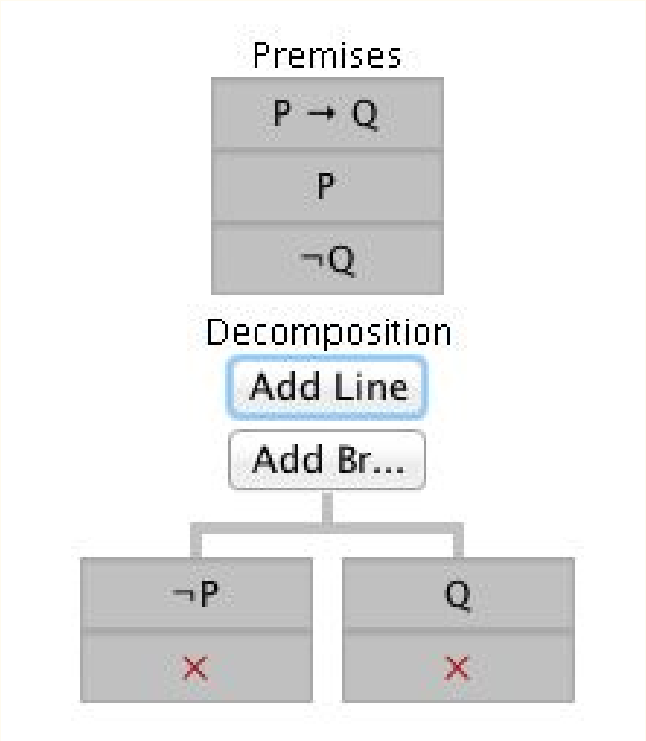
Software Solution

- Java
- Aaron Perl's Truth Tree Software
- Oracle GUI
- Git:

https://github.com/jmcusick/TruthTreeFormalizer

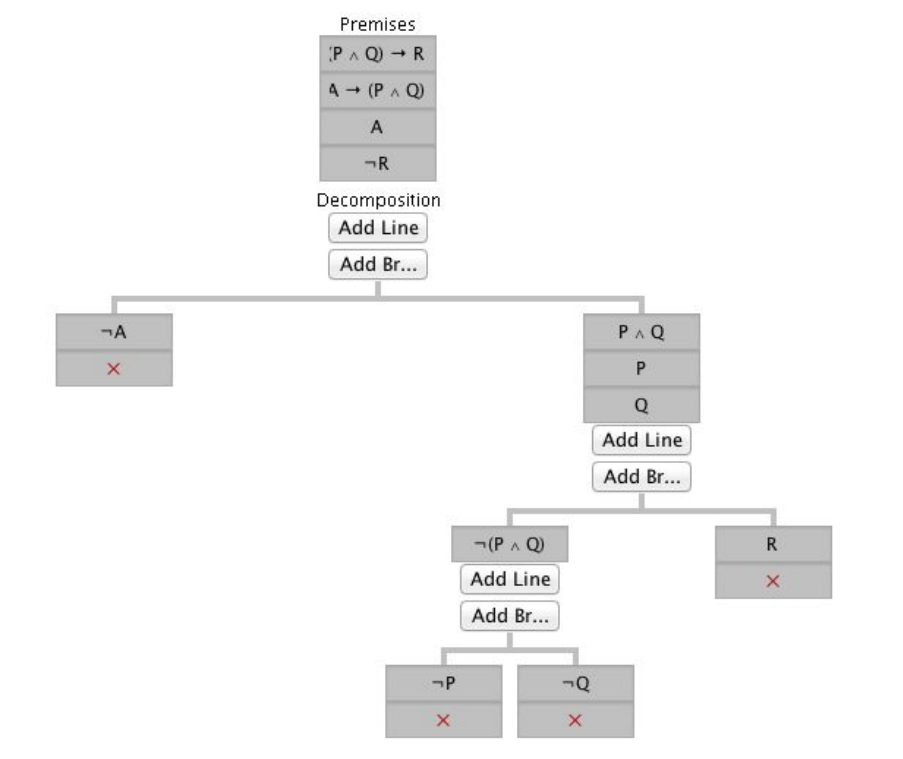
Demo

Modus Ponens



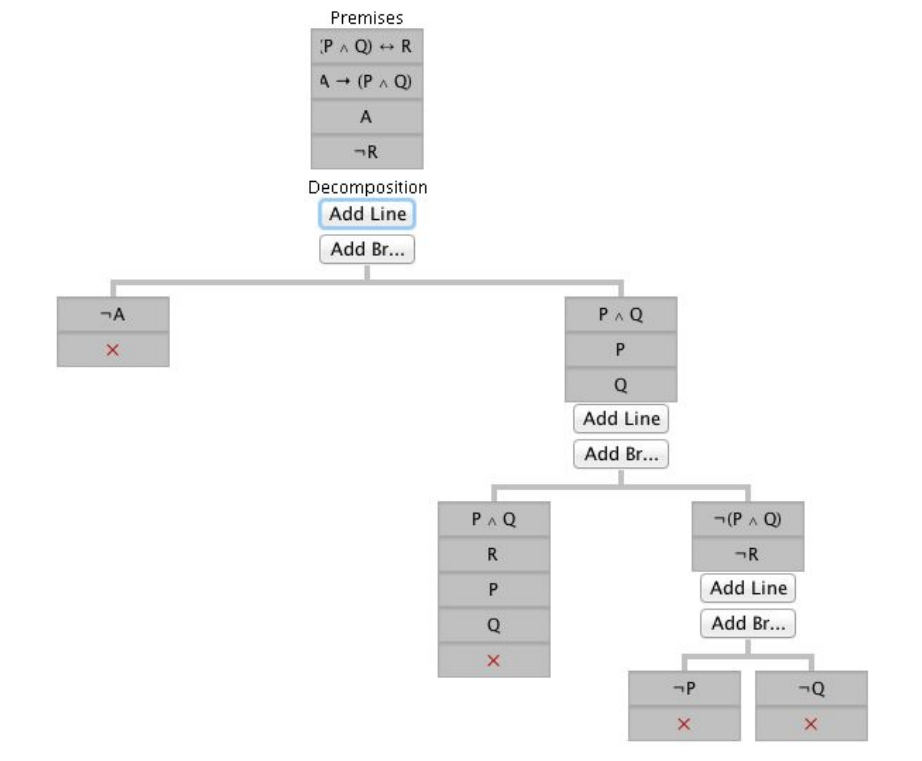
Demo

Example 2



Demo

Example 3



Future Additions

- FOL truth trees
- Invalid truth trees
- Fitch to truth tree
- Intermediate formal Fitch proof file format
- Fitch “beautifier”

References

[1] Aaron Perl's Truth Trees
- https://github.com/aaronperl/tftrees

[2] Language Proof and Logic -- Fitch
- https://ggweb.gradegrinder.net/lpl

[3] Bram van Heuveln's Slides
- http://www.cogsci.rpi.edu/~heuveb/

Questions?