Logic and the set theory
Lecture 4: Refutation trees

S. Choi
Department of Mathematical Science
KAIST, Daejeon, South Korea

Fall semester, 2012
About this lecture

- Refutation tree and valid argument
About this lecture

- Refutation tree and valid argument
- Refutation Tree Rules
About this lecture

- Refutation tree and valid argument
- Refutation Tree Rules
- Course homepages: http://mathsci.kaist.ac.kr/~schoi/logic.html and the moodle page http://moodle.kaist.ac.kr
About this lecture

- Refutation tree and valid argument
- Refutation Tree Rules
- Course homepages: http://mathsci.kaist.ac.kr/~schoi/logic.html and the moodle page http://moodle.kaist.ac.kr
- Grading and so on in the moodle. Ask questions in moodle.
Some helpful references

- Richard Jeffrey, Formal logic: its scope and limits, Mc Graw Hill
- Whitehead, Russel, Principia Mathematica (our library). (This could be a project idea.)
- See also "The Search-for-Counterexample Test for Validity."

S. Choi (KAIST)
Some helpful references

- Richard Jeffrey, Formal logic: its scope and limits, Mc Graw Hill
Some helpful references

- Richard Jeffrey, Formal logic: its scope and limits, Mc Graw Hill
- Whitehead, Russel, Principia Mathematica (our library). (This could be a project idea.)
Some helpful references

- Richard Jeffrey, Formal logic: its scope and limits, Mc Graw Hill
- Whitehead, Russel, Principia Mathematica (our library). (This could be a project idea.)
Some helpful references

- Whitehead, Russel, *Principia Mathematica* (our library). (This could be a project idea.)
- [http://plato.stanford.edu/contents.html](http://plato.stanford.edu/contents.html) has much resource.
Recall the valid argument
Recall the valid argument

To check, we need to show the premises $T, T, \ldots, T$ imply that the conclusion is $T$ always.
Recall the valid argument

To check, we need to show the premises $T, T, .., T$ imply that the conclusion is $T$ always.

Start by negating the conclusion. Then show that premises and the negated conclusion cannot be all true at the same time. Then this is a valid argument.
Recall the valid argument

To check, we need to show the premises $T, T, \ldots, T$ imply that the conclusion is $T$ always.

Start by negating the conclusion. Then show that premises and the negated conclusion cannot be all true at the same time. Then this is a valid argument.

If the premises and the negated conclusions are all true in some way, then the argument is invalid.
Recall the valid argument

To check, we need to show the premises $T, T, \ldots, T$ imply that the conclusion is $T$ always.

Start by negating the conclusion. Then show that premises and the negated conclusion cannot be all true at the same time. Then this is a valid argument.

If the premises and the negated conclusions are all true in some way, then the argument is invalid.

Note that I did not supply a proof that this works always.
Refutation tree example

We break the statements down to atomic items and see if there can be all true instances or not.
Refutation tree example

- We break the statements down to atomic items and see if there can be all true instances or not.
- The aim is to obtain paths of atomic statements.
We break the statements down to atomic items and see if there can be all true instances or not.
The aim is to obtain paths of atomic statements.

\[ P \land Q \vdash P. \]
We break the statements down to atomic items and see if there can be all true instances or not.
The aim is to obtain paths of atomic statements.

- \( P \land Q \vdash P \).
- \( P \land Q, \neg P \).
Refutation tree example

- We break the statements down to atomic items and see if there can be all true instances or not.
- The aim is to obtain paths of atomic statements.
- $P \land Q \vdash P$.
- $P \land Q, \neg P$.
- $\checkmark P \land Q, P, Q, \neg P$. 
Refutation tree example

- We break the statements down to atomic items and see if there can be all true instances or not.
- The aim is to obtain paths of atomic statements.
  - $P \land Q \vdash P$.
  - $P \land Q, \neg P$.
  - $\checkmark P \land Q, P, Q, \neg P$.
- The nonchecked atomic items cannot all be true.
Refutation tree example

- We break the statements down to atomic items and see if there can be all true instances or not.
- The aim is to obtain paths of atomic statements.
- $P \land Q \vdash P$.
- $P \land Q, \neg P$.
- $P \land Q, P, Q, \neg P$.
- The nonchecked atomic items cannot all be true.
- Valid
Refutation tree example

\[ P \lor Q, \]
\[ \neg P, \]
\[ \neg Q, \]
\[ (i) P \quad (ii) Q. \]

The nonchecked atomic items cannot all be true. Thus valid.
Refutation tree example

- $P \lor Q$
- $\neg P$
- $\vdash Q$

$P \lor Q$
$\neg P$
$\neg Q$

(i) $P$
(ii) $Q$

The nonchecked atomic items cannot all be true. Thus valid.
Refutation tree example

- $P \lor Q,$
- $\neg P,$
- $\vdash Q.$

- $\checkmark P \lor Q,$
- $\neg P,$
- $\neg Q,$
- (i) $P$ or (ii) $Q.$

- $\checkmark P \lor Q,$
- $\neg P,$
- $\neg Q,$
- (i) $P$ (X) (ii) $Q.$ (X)
- The nonchecked atomic items cannot all be true.
- Thus valid.
Refutation Tree Rules

- **Negation \( \neg \)**: If any open path contains both a formula and its negation, place X. (This path is now closed)
Refutation Tree Rules

- **Negation** $\neg$: If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation** $\neg\neg$: In any open path, check any unchecked $\neg\neg \phi$ and write $\phi$ at the bottom of every path containing it.
Refutation Tree Rules

- **Negation** $\neg$: If any open path contains both a formula and its negation, place $X$. (This path is now closed)
- **Negated negation** $\neg\neg$: In any open path, check any unchecked $\neg\neg\phi$ and write $\phi$ at the bottom of every path containing it.
- **Conjunction** $\land$: In any open path, check any unchecked $\phi \land \psi$ and write $\phi$ and $\psi$ at the bottom of every path containing it. (same path)

A path is finished (or closed) if $X$ appears. See 3.27 and 3.28.
Refutation Tree Rules

- **Negation** $\neg$: If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation** $\neg\neg$: In any open path, check any unchecked $\neg\neg\phi$ and write $\phi$ at the bottom of every path containing it.
- **Conjunction** $\land$: In any open path, check any unchecked $\phi \land \psi$ and write $\phi$ and $\psi$ at the bottom of every path containing it. (same path)
- **Disjunction** $\lor$: If an open path contain unchecked $\phi \lor \psi$, then check it and the split the bottom of every path containing it into two with (i) one $\phi$ added and (ii) the other $\psi$ added.

A path is finished (or closed) if X appears.

See 3.27 and 3.28.
Refutation Tree Rules

- **Negation ¬**: If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation ¬¬**: In any open path, check any unchecked ¬¬φ and write φ at the bottom of every path containing it.
- **Conjunction ∧**: In any open path, check any unchecked φ ∧ ψ and write φ and ψ at the bottom of every path containing it. (same path)
- **Disjunction ∨**: If an open path contain unchecked φ ∨ ψ, then check it and the split the bottom of every path containing it into two with (i) one φ added and (ii) the other ψ added.
- **Conditional →**: Unchecked φ → ψ. Check it and branch every path containing it into two (i) ¬φ (ii) ψ.
Refutation Tree Rules

- **Negation** \( \neg \): If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation** \( \neg \neg \): In any open path, check any unchecked \( \neg \neg \phi \) and write \( \phi \) at the bottom of every path containing it.
- **Conjunction** \( \land \): In any open path, check any unchecked \( \phi \land \psi \) and write \( \phi \) and \( \psi \) at the bottom of every path containing it. (same path)
- **Disjunction** \( \lor \): If an open path contains an unchecked \( \phi \lor \psi \), then check it and split the bottom of every path containing it into two with (i) one \( \phi \) added and (ii) the other \( \psi \) added.
- **Conditional** \( \rightarrow \). Unchecked \( \phi \rightarrow \psi \). Check it and branch every path containing it into two (i) \( \neg \phi \) (ii) \( \psi \).
- **Biconditional** \( \leftrightarrow \). Unchecked \( \phi \leftrightarrow \psi \). Check it and branch every path containing it into two (i) \( \neg \phi, \neg \psi \) and (ii) \( \phi, \psi \).
Refutation Tree Rules

- **Negation** ¬: If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation** ¬¬: In any open path, check any unchecked ¬¬φ and write φ at the bottom of every path containing it.
- **Conjunction** ∧: In any open path, check any unchecked φ ∧ ψ and write φ and ψ at the bottom of every path containing it. (same path)
- **Disjunction** ∨: If an open path contain unchecked φ ∨ ψ, then check it and the split the bottom of every path containing it into two with (i) one φ added and (ii) the other ψ added.
- **Conditional** →. Unchecked φ → ψ. Check it and branch every path containing it into two (i) ¬φ (ii) ψ.
- **Biconditional** ↔. Unchecked φ ↔ ψ. Check it and branch every path containing it into two (i) ¬φ, ¬ψ and (ii) φ, ψ.
- A path is finished (or closed) if X appears.
Refutation Tree Rules

- **Negation** $\neg$: If any open path contains both a formula and its negation, place X. (This path is now closed)
- **Negated negation** $\neg\neg$: In any open path, check any unchecked $\neg\neg\phi$ and write $\phi$ at the bottom of every path containing it.
- **Conjunction** $\land$: In any open path, check any unchecked $\phi \land \psi$ and write $\phi$ and $\psi$ at the bottom of every path containing it. (same path)
- **Disjunction** $\lor$: If an open path contain unchecked $\phi \lor \psi$, then check it and the split the bottom of every path containing it into two with (i) one $\phi$ added and (ii) the other $\psi$ added.
- **Conditional** $\rightarrow$. Unchecked $\phi \rightarrow \psi$. Check it and branch every path containing it into two (i) $\neg\phi$ (ii) $\psi$.
- **Biconditional** $\leftrightarrow$. Unchecked $\phi \leftrightarrow \psi$. Check it and branch every path containing it into two (i) $\neg\phi, \neg\psi$ and (ii) $\phi, \psi$.
- A path is finished (or closed) if X appears.
- See 3.27 and 3.28.
Refutation Tree Rules

- Negated conjunction $\neg \wedge$: Unchecked $\neg (\phi \land \psi)$. Check it and split the bottom of every open path containing it into two (i) add $\neg \phi$ (ii) add $\neg \psi$. 
Refutation Tree Rules

- Negated conjunction $\neg \wedge$: Unchecked $\neg (\phi \land \psi)$. Check it and split the bottom of every open path containing it into two (i) add $\neg \phi$ (ii) add $\neg \psi$.

- Negated disjunction $\neg \lor$: unchecked $\neg (\phi \lor \psi)$ and write $\neg \phi$ and $\neg \psi$ at the bottom of every (open) path containing it.
Refutation Tree Rules

- **Negated conjunction** $\neg\land$: Unchecked $\neg(\phi \land \psi)$. Check it and split the bottom of every open path containing it into two (i) add $\neg\phi$ (ii) add $\neg\psi$.

- **Negated disjunction** $\neg\lor$: unchecked $\neg(\phi \lor \psi)$ and write $\neg\phi$ and $\neg\psi$ at the bottom of every (open) path containing it.

- **Negated conditional** $\neg\rightarrow$: In any open path, check any unchecked $\neg(\phi \rightarrow \psi)$ and write $\phi$ and $\neg\psi$ at the bottom of every path containing it. (same path)
Negated conjunction \( \neg \wedge \): Unchecked \( \neg (\phi \wedge \psi) \). Check it and split the bottom of every open path containing it into two (i) add \( \neg \phi \) (ii) add \( \neg \psi \).

Negated disjunction \( \neg \lor \): unchecked \( \neg (\phi \lor \psi) \) and write \( \neg \phi \) and \( \neg \psi \) at the bottom of every (open) path containing it.

Negated conditional \( \neg \rightarrow \): In any open path, check any unchecked \( \neg (\phi \rightarrow \psi) \) and write \( \phi \) and \( \neg \psi \) at the bottom of every path containing it. (same path)

Negated biconditional \( \neg \leftrightarrow \): In any open path, check any unchecked \( \neg (\phi \leftrightarrow \psi) \) and branch the bottom of every path containing it into two write \( \phi \) and \( \neg \psi \) at one (i) and write \( \neg \phi \) and \( \psi \) (ii)
Example

1. \( B \rightarrow \neg A \)
2. \( \neg B \rightarrow C \).
3. \( \neg (A \rightarrow C) \).

Conclusion \( A \rightarrow C \).
Example

- 1. \( B \rightarrow \neg A \)
- 2. \( \neg B \rightarrow C \)
- Conclusion \( A \rightarrow C \)

- 1. \( B \rightarrow \neg A \)
- 2. \( \neg B \rightarrow C \)
- 3. \( \neg (A \rightarrow C) \)

- 4. \( A \)
- 5. \( \neg C \)
Example

1. \( B \rightarrow \neg A \)
2. \( \neg B \rightarrow C \).
3. Conclusion \( A \rightarrow C \).

1. \( B \rightarrow \neg A \)
2. \( \neg B \rightarrow C \),
3. \( \neg (A \rightarrow C) \).

1. \( B \rightarrow \neg A \)
2. \( \neg B \rightarrow C \),
3. \( \neg (A \rightarrow C) \).
4. \( A \),
5. \( \neg C \).
1. $B \rightarrow \neg A$,
2. $\neg B \rightarrow C$,
3. $\neg (A \rightarrow C)$.
4. $A$,
5. $\neg C$
6. (i) $\neg B$ (ii) $\neg A$ (X) from 4.

Now complete. valid
1. $B \rightarrow \neg A$, 
2. $\neg B \rightarrow C$, 
3. $\neg (A \rightarrow C)$.

4. $A$, 
5. $\neg C$

6 (i) $\neg B$ (ii) $\neg A$ (X) from 4.

Now complete. valid
Open tree case

If open path arises without X, then invalid.

1. $A \rightarrow B$
2. $\neg A$
3. $\vdash B$.

(i) $\neg A$
(ii) $B$. (X).

(i) is still alive.

Invalid case: $\neg A, \neg B$ is the counter example.
Open tree case

If open path arises without $X$, then invalid.

1. $A \rightarrow B$
2. $\neg A$
3. $\vdash B$

1. $A \rightarrow B$
2. $\neg A$
3. $\neg B$. 

Invalid case: $\neg A$, $\neg B$ is the counterexample.
Open tree case

If open path arises without X, then invalid.

1. $A \rightarrow B$
2. $\neg A$
3. $\vdash B$

1. $A \rightarrow B$
2. $\neg A$
3. $\neg B$

✓

1. $A \rightarrow B$
2. $\neg A$
3. $\neg B$

(i) $\neg A$ (ii) $B$. (X).
(i) is still alive.
Invalid case: $\neg A, \neg B$ is the counter example.
A wff $\phi$ is a tautology if and only if $\neg\phi$ is truth-functionally inconsistent.
Tautology Rules

A wff $\phi$ is a tautology if and only if $\neg\phi$ is truth-functionally inconsistent.

$\phi$ is a tautology if and only if all path in the finished tree are closed.
Tautology Rules: An example

- \( \neg(A \lor B) \iff \neg A \land \neg B \).
- \( \neg(\neg(A \lor B) \iff \neg A \land \neg B) \).
- negation first.
Tautology Rules: An example

\( \neg(A \lor B) \leftrightarrow \neg A \land \neg B. \)
\( \neg(\neg(A \lor B) \leftrightarrow \neg A \land \neg B). \)
negation first.

✓ \( \neg(\neg(A \lor B) \leftrightarrow (\neg A \land \neg B)). \)

(i) \( \neg(\neg(A \lor B)) \), (ii) \( \neg(A \lor B) \)

(i) \( \neg(A \land \neg B) \), (ii) \( \neg(\neg A \land \neg B) \).

✓ \( \neg \leftrightarrow \) rule.
Tautology Rules: An example

- \( \neg (A \lor B) \iff \neg A \land \neg B. \)
- \( \neg (\neg (A \lor B) \iff \neg A \land \neg B). \)
- negation first.

- \( \checkmark \neg (\neg (A \lor B) \iff (\neg A \land \neg B)). \)
- (i) \( \neg (A \lor B), \) (ii) \( \neg (A \lor B) \)
- (i) \( \neg A \land \neg B), \) (ii) \( \neg (A \land \neg B) \)
- \( \neg \iff \) rule.

\( \checkmark \neg (\neg (A \lor B) \iff (\neg A \land \neg B)). \)

- (i) \( \neg (A \lor B), \) (ii) \( \neg (A \lor B) \)
- (i) \( \neg A \land \neg B), \) (ii) \( \neg (A \land \neg B) \)
- \( \neg \iff \) rule.
Tautology Rules: An example

1. $\neg (A \lor B) \leftrightarrow \neg A \land \neg B$.
2. $\neg (\neg (A \lor B) \leftrightarrow \neg A \land \neg B)$.

4. $\checkmark \neg (\neg (A \lor B) \leftrightarrow (\neg A \land \neg B))$.
5. (i) $\neg (\neg (A \lor B))$,
6. (i) $(\neg A \land \neg B)$,
7. $\neg \leftrightarrow$ rule.

8. $\checkmark \neg (A \lor B) \leftrightarrow (\neg A \land \neg B))$.
9. (ii) $\neg (A \lor B)$
10. (ii) $(\neg A \land \neg B)$.
11. $\neg \leftrightarrow$ rule.
✓ \neg (\neg (A \lor B) \leftrightarrow (\neg A \land \neg B)).

✓ (i) \neg (\neg (A \lor B)),

✓ (i) (\neg A \land \neg B),

(i) (A \lor B) \dashv \vdash rule.

(i) \neg A,

(i) \neg B (Conjunction rule).
Refutation Tree Rules

✓ \neg (\neg (A \lor B) \leftrightarrow (\neg A \land \neg B)).
✓ (i) \neg (\neg (A \lor B)),
✓ (i) (\neg A \land \neg B),
(i) (A \lor B) \neg \neg rule.
(i) \neg A,
(i) \neg B (Conjunction rule).

✓ \neg (\neg (A \lor B) \leftrightarrow (\neg A \land \neg B)).
✓ (i) \neg (\neg (A \lor B)),
✓ (i) (\neg A \land \neg B),
✓ (i) (A \lor B)
(i) \neg A,
(i) \neg B
(i)(i) A (X) (i)(ii) B (X) (Disjunction rule)
✓ \neg(\neg(A \lor B) \leftrightarrow (\neg A \land \neg B))

✓ (ii) \neg(A \lor B)

(ii) \neg(\neg A \land \neg B)

(ii) \neg A

(ii) \neg B \lor \text{rule}
Refutation Tree Rules

- ✓ $\neg \neg (\neg (A \lor B) \iff (\neg A \land \neg B))$.
- ✓ (ii) $\neg (A \lor B)$
- (ii) $\neg (\neg A \land \neg B)$.
- (ii) $\neg A$
- (ii) $\neg B \lor \text{rule}$

- ✓ $\neg (\neg (A \lor B) \iff (\neg A \land \neg B))$.
- ✓ (ii) $\neg (A \lor B)$
- ✓ (ii) $\neg (\neg A \land \neg B)$.
- (ii) $\neg A$
- (ii) $\neg B \lor \text{rule}$
- (ii)(i) $\neg
A \ (X)$ (ii)(ii) $\neg
B \ (X) \land \text{rule.}$
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
- The order of rules applied does not make any difference. It is more efficient to apply nonbranching rules first.
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
- The order of rules applied does not make any difference. It is more efficient to apply nonbranching rules first.
- The process eventually terminates. (not go on forever). Decidability.
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
- The order of rules applied does not make any difference. It is more efficient to apply nonbranching rules first.
- The process eventually terminates. (not go on forever). Decidability.
- Soundness of the test: If we obtain validity from the test, then we can trust it.
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
- The order of rules applied does not make any difference. It is more efficient to apply nonbranching rules first.
- The process eventually terminates. (not go on forever). Decidability.
- Soundness of the test: If we obtain validity from the test, then we can trust it.
- Completeness of the test: If we obtain invalidity from the test, then we can trust it: we even get counter-examples.
Some helpful remarks

- Do not apply rules to subformulas. (Confusing)
- The order of rules applied does not make any difference. It is more efficient to apply nonbranching rules first.
- The process eventually terminates. (not go on forever). Decidability.
- Soundness of the test: If we obtain validity from the test, then we can trust it.
- Completeness of the test: If we obtain invalidity from the test, then we can trust it: we even get counter-examples.
- We need proof: Omit proof in R. Jeffery, Formal logic page 34.