9.2.3

Reasoning with Propositional Logic

Chapter 7.4–7.8

Some material adopted from notes by Andreas Geyer-Schulz and Chuck Dyer

Overview

- There are many ways to approach reasoning with propositional logic
- We’ll look at one, resolution refutation, that can be extended to first order logic
- Later, we will look other approaches that are special to propositional logic

Reasoning / Inference

- **Logical inference** creates new sentences that logically follow from a set of sentences (KB)
- It can also detect if a KB is inconsistent, i.e., has sentences that entail a **contradiction**
- An inference rule is **sound** if every sentence it produces from a KB logically follows from the KB
  - i.e., inference rule creates no contradictions
- An inference rule is **complete** if it can produce every expression that logically follows from (is entailed by) the KB
  - Note analogy to complete search algorithms

Sound rules of inference

Examples of sound rules of inference

Each can be shown to be sound using a truth table

RULE	PREMISE	CONCLUSION
Modus Ponens	$A, A \rightarrow B$	B
And Introduction	A, B	$A \wedge B$
And Elimination	$A \wedge B$	A
Double Negation	$\neg\neg A$	A
Unit Resolution	$A \vee B, \neg B$	A
Resolution	$\mathbf{A \vee B, \neg B \vee C}$	$\mathbf{A \vee C}$

Resolution

- **Resolution** is a valid inference rule producing a new clause implied by two clauses containing *complementary literals*

  Literal: atomic symbol or its negation, i.e., P, ~P

- Amazingly, this is the **only** interference rule needed to build a sound & complete theorem prover
  - Based on proof by contradiction, usually called resolution refutation
- The resolution rule was discovered by Alan Robinson (CS, U. of Syracuse) in the mid 1960s

Resolution

- A KB is a set of sentences all of which are true, i.e., a conjunction of sentences
- To use resolution, put KB into *conjunctive normal form* (CNF)
  - Each sentence is a disjunction of one or more literals (positive or negative atoms)
- Every KB can be put into CNF, by rewriting its sentences using standard tautologies, e.g.:
  - $P \rightarrow Q \equiv \sim P \vee Q$
  - $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R) \equiv (P \vee Q)$, $(P \vee R)$

Resolution Example

Tautologies

$(A \rightarrow B) \leftrightarrow (\sim A \vee B)$

$(A \vee (B \wedge C)) \leftrightarrow (A \vee B) \wedge (A \vee C)$

- KB: $[P \rightarrow Q , Q \rightarrow R \wedge S]$
- KB: $[P \rightarrow Q , Q \rightarrow R, Q \rightarrow S]$
- KB in CNF: $[\sim P \vee Q , \sim Q \vee R , \sim Q \vee S]$
- Resolve KB[0] and KB[1] producing:

  $\sim P \vee R$ *(i.e., $P \rightarrow R$)*
- Resolve KB[0] and KB[2] producing:

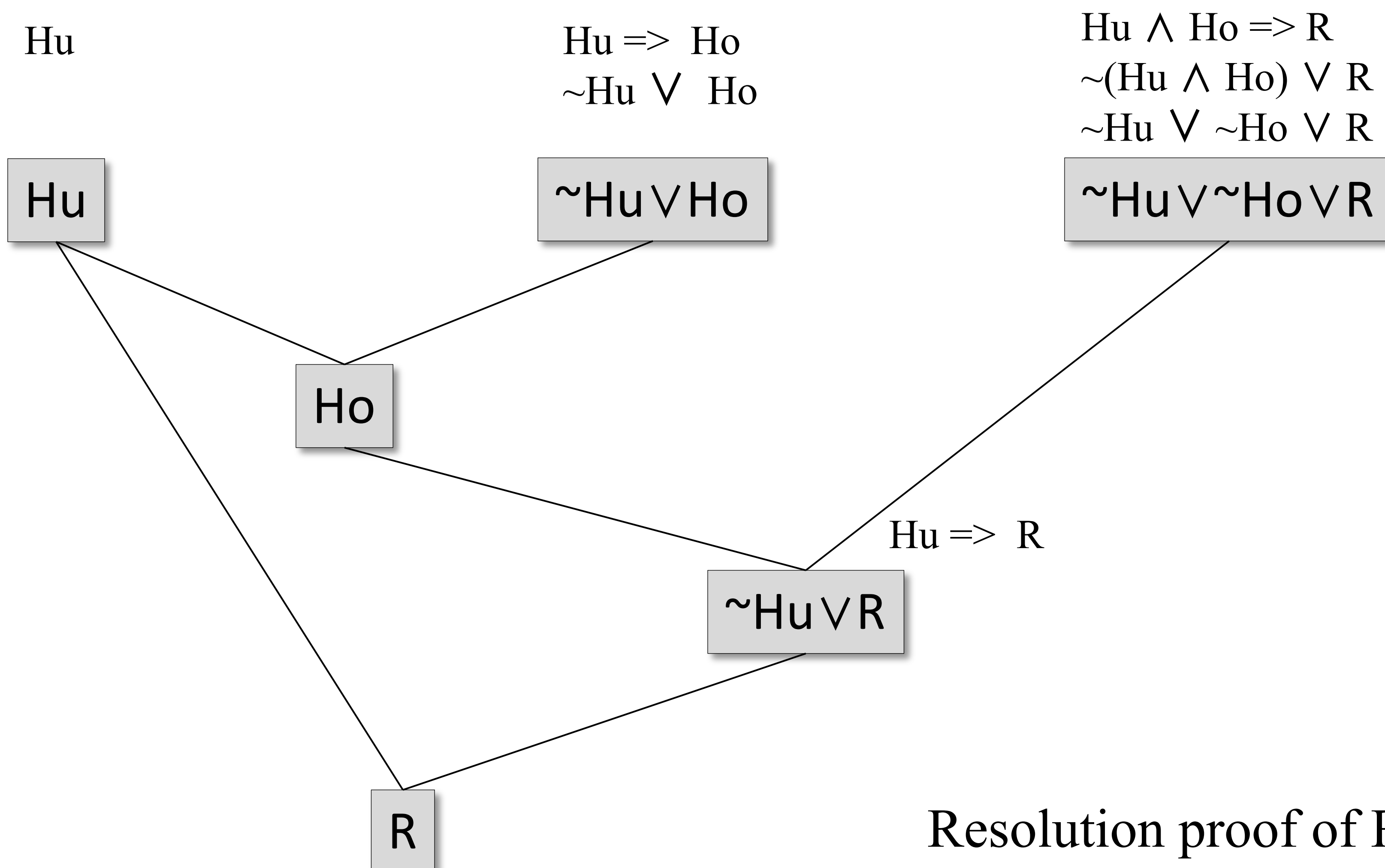
  $\sim P \vee S$ *(i.e., $P \rightarrow S$)*
- New KB: $[\sim P \vee Q , \sim Q \vee R, \sim Q \vee S, \sim P \vee R, \sim P \vee S]$

Proving it's raining with rules

- A **proof** is a sequence of sentences, where each is a premise (i.e., a given) or is derived from earlier sentences in the proof by an inference rule
- Last sentence is the **theorem** (also called goal or query) that we want to prove
- The *weather problem* using traditional reasoning

1 Hu	premise	"It's humid"
2 Hu$\rightarrow$Ho	premise	"If it's humid, it's hot"
3 Ho	modus ponens(1,2)	"It's hot"
4 (Ho$\wedge$Hu)$\rightarrow$R	premise	"If it's hot & humid, it's raining"
5 Ho$\wedge$Hu	and introduction(1,3)	"It's hot and humid"
6 R	modus ponens(4,5)	"It's raining"

Proving it's raining with resolution



Resolution proof of R

A simple proof procedure

This procedure generates new sentences in a KB

1. Convert all sentences in the KB to CNF[1]
2. Find all pairs of sentences with complementary literals[2] that have not yet been resolved
3. If there are no pairs stop else resolve each pair, adding the result to the KB and go to 2

- Is it sound?, complete? always terminate?

1: a KB in conjunctive normal form is a set of disjunctive sentences

2: a literal is a variable or its negation

Propositional Resolution

- It is sound!
- It's not *generatively complete* in that it can't derive all clauses that follow from the KB
  - The issues are not serious limitations, though
  - Example: if the KB includes P and includes Q we won't derive P ^ Q
- It will always terminate
- But generating all clauses that follow can take a long time and many may be useless

Refutation proofs

- Common use case: we have a question/goal (e.g, **P**) and want to know if it's true
- A refutation proof is a common approach:
  - We start with a KB with all true facts
  - Add negation of what we want to prove to KB (e.g., **~P**)
  - Try to find a contradiction
  - If proof ever produces one, it must be due to adding **~P**, so goal is proven
- Procedure easy to focus & control, so is tends to be more efficient

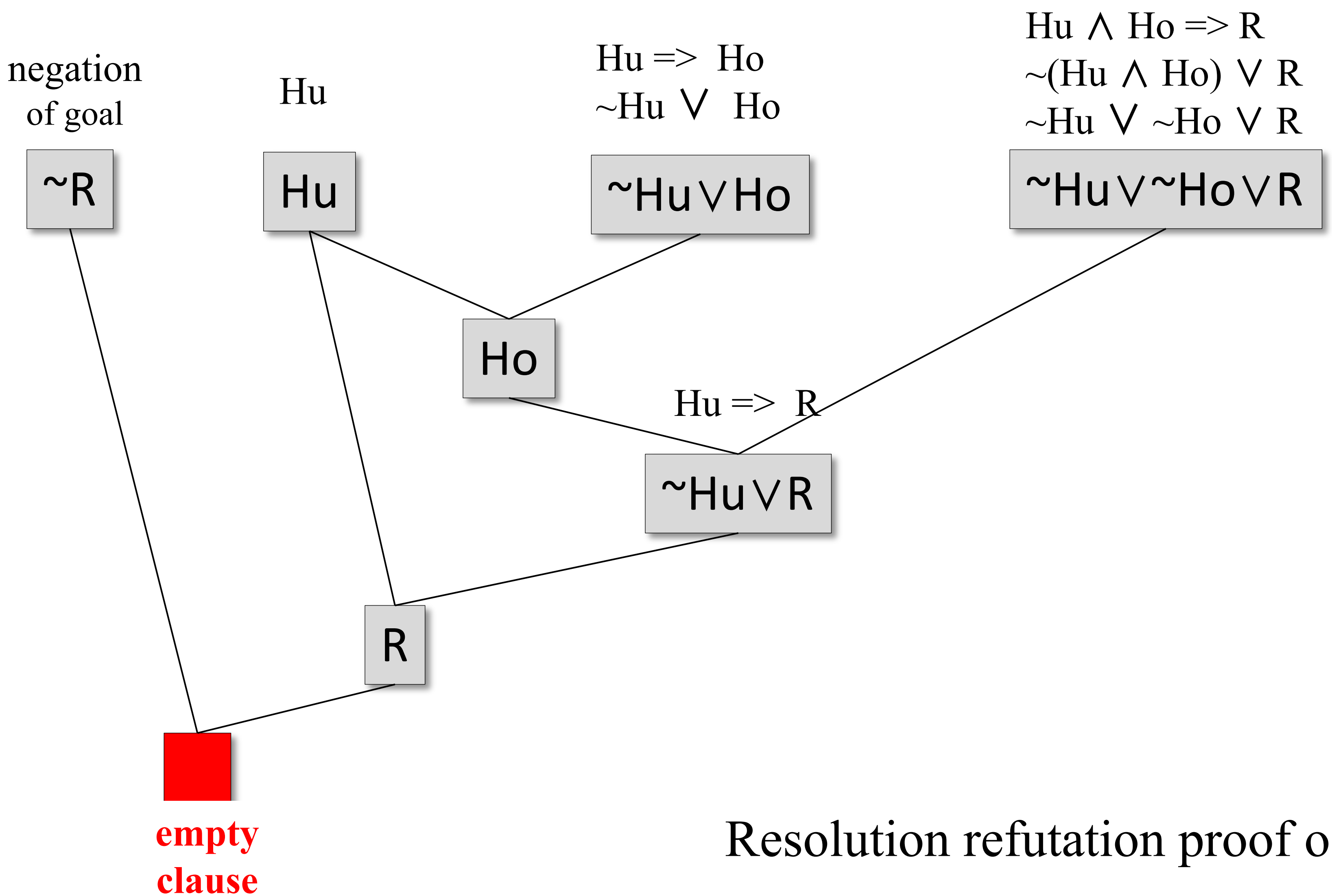
Resolution refutation

Procedure tries to prove a goal **P**

1. Add negation of goal to the KB, **~P**
2. Convert all sentences in KB to CNF
3. Find pairs of sentences with complementary literals that have not yet been resolved
4. If there are no pairs stop else resolve each pair, adding the result to the KB and go to 2

- If we get an empty clause (i.e., a contradiction) then **P** follows from the KB
  - e.g., resolving **X** with **~X** results in an empty clause
- If not, conclusion can't be proved from the KB

Proving it's raining with refutation resolution



Resolution refutation proof of R

Propositional Reasoning

- There are other reasoning tasks with propositions
- **Satisfiability** involves finding a set of values that will make a KB true
- There are many efficient and scalable proof procedures for sets of propositions
- Reducing a problem to a set of propositions and using an efficient proof technique is often a good way to solve a problem

Fin