Artificial Intelligence Programming
Resolution and Expert Systems

Chris Brooks

Department of Computer Science
University of San Francisco
Recall that Resolution is a rule that lets us combine clauses.

$A \lor B$ and $\neg A \lor C$ allows us to conclude $B \lor C$.

$A$ is either true or not true. If $A$ is true, then $C$ must be true.

If $A$ is false, then $B$ must be true.

This can be generalized to clauses of any length.

Sentences must first be in CNF.
12-1: Resolution example

Start with the following sentences:

- \( R_1 : \neg p_{1,1} \) - There is no pit in 1,1
- \( R_2 : B_{1,1} \iff (p_{1,2} \lor p_{2,1}) \)
- \( R_3 : B_{2,1} \iff (p_{1,1} \lor p_{2,2} \lor p_{3,1}) \) (a square is breezy iff there is a pit in a neighboring square)
- \( R_4 : \neg B_{1,1} \) - It’s not breezy in (1,1)
- \( R_5 : B_{2,1} \) - It is breezy in 2,1

Initially, we used logical inference to derive new facts.
12-2: New facts

6 $R_6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$
(biconditional elimination on $R_2$).

6 $R_7 : (P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1}$ - And-elimination on $R_6$.

6 $R_8 : (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \lor P_{2,1}))$ - contraposition on $R_7$.

6 $R_9 : \neg(P_{1,2} \lor P_{2,1})$ - Modus Ponens on $R_9$ and $R_4$

6 $R_{10} \neg P_{1,2} \land \neg P_{2,1}$ DeMorgan’s on R10.
Add the following facts:

\[ R_{11} : \neg B_{1,2} \]
\[ R_{12} : B_{1,2} \iff (P_{1,1} \lor P_{2,2} \lor P_{1,3}) \]

We can re-apply the same process as on the previous slide to derive:

\[ R_{13} : \neg P_{2,2} \]
\[ R_{14} : \neg P_{1,3} \]

Now let’s derive more new facts.

\[ R_{15} : P_{1,1} \lor P_{2,2} \lor P_{3,1} - \text{biconditional elimination on } R_3, \text{ followed by Modus Ponens with } R_5. \]
Now we can apply resolution.

$R_{16} : P_{1,1} \lor P_{3,1}$ Resolution with $R_{13}$ and $R_{15}$

$R_{17} : P_{3,1}$ Resolution with $R_{16}$ and $R_{1}$

Now we can conclude that there is a pit in 3,1.
Converting our KB to CNF gives us with the following sentences:

- $R_1 : \neg P_{2,1} \lor B_{1,1}$
- $R_2 : \neg B_{1,1} \lor P_{1,2} \lor P_{2,1}$
- $R_3 : \neg P_{1,2} \lor B_{1,1}$
- $R_4 : \neg B_{1,1}$

We want to prove $\neg P_{1,2}$, so we insert the opposite into the KB.

$R_5 : P_{1,2}$
12-6: Refutation by resolution

- Resolving $R_4$ and $R_3$ gives us $R_6 : \neg P_{1,2}$
- Resolving $R_6$ with $R_5$ gives us FALSE - we have a contradiction.
- Assuming our original facts were all true, the contradiction must derive from the sentence we inserted.
- Therefore, that sentence is false, and its opposite $\neg P_{1,2}$ is true.

When to use refutation, and when to use forward chaining?
- Forward chaining is useful for deriving all possible new facts.
- Refutation is useful for proving the truth of a particular new fact.
Expert systems are programs designed to model and reason about human knowledge.

Typically focused around a domain
- Disease diagnosis
- Space flight
- Logistics
- Software troubleshooting

Programmed declaratively
- Programmers add *facts* and *rules*
- System automates the inference process.

Can either do forward chaining (production systems) or backward chaining (theorem provers)
Jess is a port of a well-known expert system shell called CLIPS.
Ported from C to Java
Has its own knowledge representation language.
Can also interface with Java libraries
Jess is a forward chaining system
- You write facts and rules, Jess derives their consequences.