

# Propositional Resolution

Michael Genesereth

Autumn 2007

## Axiom Schema Instances

$$\varphi \Rightarrow (\psi \Rightarrow \varphi)$$

$$\begin{array}{lll}
 p \Rightarrow (p \Rightarrow p) & p \Rightarrow (p \wedge p \Rightarrow p) & p \Rightarrow (p \vee p \Rightarrow p) \\
 p \Rightarrow (q \Rightarrow p) & p \Rightarrow (p \wedge q \Rightarrow p) & p \Rightarrow (p \vee q \Rightarrow p) \\
 p \Rightarrow (r \Rightarrow p) & p \Rightarrow (p \wedge r \Rightarrow p) & p \Rightarrow (p \vee r \Rightarrow p) \\
 q \Rightarrow (p \Rightarrow q) & \dots & \dots \\
 q \Rightarrow (q \Rightarrow q) & & \\
 q \Rightarrow (r \Rightarrow q) & & \\
 r \Rightarrow (p \Rightarrow r) & & \\
 r \Rightarrow (q \Rightarrow r) & & \\
 r \Rightarrow (r \Rightarrow r) & & 
 \end{array}$$

Proofs may be short, but many alternatives to consider.

## Propositional Resolution

*Propositional resolution* is a rule of inference.

Using propositional resolution alone (without axiom schemata or other rules of inference), it is possible to build a theorem prover that is sound and complete for all of Propositional Logic.

The search space using propositional resolution is much smaller than for Modus Ponens and the Standard Axiom Schemata.

3

## Clausal Form

Propositional resolution works only on expressions in *clausal form*.

Fortunately, it is possible to convert any set of propositional calculus sentences into an equivalent set of sentences in clausal form.

4

## Clausal Form

A *literal* is either an atomic sentence or a negation of an atomic sentence.

$$p, \neg p$$

A *clausal sentence* is either a literal or a disjunction of literals.

$$p, \neg p, p \vee q$$

A *clause* is a set of literals.

$$\{p\}, \{\neg p\}, \{p, q\}$$

5

## Empty Sets

The empty clause  $\{\}$  is unsatisfiable.

Why? It is equivalent to an empty disjunction.

6

## Conversion to Clausal Form

Implications Out:

$$\varphi_1 \Rightarrow \varphi_2 \rightarrow \neg \varphi_1 \vee \varphi_2$$

$$\varphi_1 \Leftarrow \varphi_2 \rightarrow \varphi_1 \vee \neg \varphi_2$$

$$\varphi_1 \Leftrightarrow \varphi_2 \rightarrow (\neg \varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \neg \varphi_2)$$

Negations In:

$$\neg \neg \varphi \rightarrow \varphi$$

$$\neg(\varphi_1 \wedge \varphi_2) \rightarrow \neg \varphi_1 \vee \neg \varphi_2$$

$$\neg(\varphi_1 \vee \varphi_2) \rightarrow \neg \varphi_1 \wedge \neg \varphi_2$$

7

## Conversion to Clausal Form

Distribution

$$\varphi_1 \vee (\varphi_2 \wedge \varphi_3) \rightarrow (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3)$$

$$(\varphi_1 \wedge \varphi_2) \vee \varphi_3 \rightarrow (\varphi_1 \vee \varphi_3) \wedge (\varphi_2 \vee \varphi_3)$$

$$\varphi \vee (\varphi_1 \vee \dots \vee \varphi_n) \rightarrow (\varphi \vee \varphi_1 \vee \dots \vee \varphi_n)$$

$$(\varphi_1 \vee \dots \vee \varphi_n) \vee \varphi \rightarrow (\varphi_1 \vee \dots \vee \varphi_n \vee \varphi)$$

$$\varphi \wedge (\varphi_1 \wedge \dots \wedge \varphi_n) \rightarrow (\varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_n)$$

$$(\varphi_1 \wedge \dots \wedge \varphi_n) \wedge \varphi \rightarrow (\varphi_1 \wedge \dots \wedge \varphi_n \wedge \varphi)$$

Operators Out

$$\varphi_1 \vee \dots \vee \varphi_n \rightarrow \{\varphi_1, \dots, \varphi_n\}$$

$$\varphi_1 \wedge \dots \wedge \varphi_n \rightarrow \varphi_1, \dots, \varphi_n$$

8

## Example

$g \wedge (r \Rightarrow f)$   
I  $g \wedge (\neg r \vee f)$   
N  $g \wedge (\neg r \vee f)$   
D  $g \wedge (\neg r \vee f)$   
O  $\{g\}$   
 $\{\neg r, f\}$

9

## Example

10

## Resolution Principle

General:

$$\frac{\begin{array}{l} \{\varphi_1, \dots, \chi, \dots, \varphi_m\} \\ \{\psi_1, \dots, \neg\chi, \dots, \psi_n\} \end{array}}{\{\varphi_1, \dots, \varphi_m, \psi_1, \dots, \psi_n\}}$$

Example:

$$\frac{\begin{array}{l} \{p, q\} \\ \{\neg p, r\} \end{array}}{\{q, r\}}$$

11

## Issues

Collapse

$$\frac{\begin{array}{l} \{\neg p, q\} \\ \{p, q\} \end{array}}{\{q\}}$$

Singletons

$$\frac{\begin{array}{l} \{\neg p, q\} \\ \{p\} \end{array}}{\{q\}} \qquad \frac{\begin{array}{l} \{p\} \\ \{\neg p\} \end{array}}{\{\}}$$

12

## Issues

### Multiple Conclusions

$$\begin{array}{l} \{p, q\} \\ \{\neg p, \neg q\} \\ \hline \{p, \neg p\} \\ \{q, \neg q\} \end{array}$$

### Single Application Only

$$\begin{array}{l} \{p, q\} \\ \{\neg p, \neg q\} \\ \hline \{\} \end{array}$$

Wrong!!

13

## Special Cases

### Modus Ponens

$$\begin{array}{l} p \Rightarrow q \\ p \\ \hline q \end{array}$$

$$\begin{array}{l} \{\neg p, q\} \\ \{p\} \\ \hline \{q\} \end{array}$$

### Modus Tolens

$$\begin{array}{l} p \Rightarrow q \\ \neg q \\ \hline \neg p \end{array}$$

$$\begin{array}{l} \{\neg p, q\} \\ \{\neg q\} \\ \hline \{\neg p\} \end{array}$$

### Chaining

$$\begin{array}{l} p \Rightarrow q \\ q \Rightarrow r \\ \hline p \Rightarrow r \end{array}$$

$$\begin{array}{l} \{\neg p, q\} \\ \{\neg q, r\} \\ \hline \{\neg p, r\} \end{array}$$

14

## Incompleteness?

Propositional Resolution is not *generatively* complete.

We cannot generate  $p \Rightarrow (q \Rightarrow p)$  using propositional resolution. There are no premises. Consequently, there are no conclusions.

15

## Answer

This apparent problem disappears if we take the clausal form of the premises (if any) together with the negated goal and try to derive the empty clause.

General Method: To determine whether a set  $\Delta$  of sentences logically entails a sentence  $\varphi$ , rewrite  $\Delta \cup \{\neg\varphi\}$  in clausal form and try to derive the empty clause using the resolution rule of inference.

16

## Example

$$\neg(p \Rightarrow (q \Rightarrow p))$$

$$\text{I } \neg(\neg p \vee \neg q \vee p)$$

$$\text{N } \neg\neg p \wedge \neg\neg q \wedge \neg p$$

$$p \wedge q \wedge \neg p$$

$$\text{D } p \wedge q \wedge \neg p$$

$$\text{O } \{p\}$$

$$\{q\}$$

$$\{\neg p\}$$

17

## Example

If Mary loves Pat, then Mary loves Quincy. If it is Monday, Mary loves Pat or Quincy. Prove that, if it is Monday, then Mary loves Quincy.

1.  $\{\neg p, q\}$  Premise
2.  $\{\neg m, p, q\}$  Premise
3.  $\{m\}$  Negated Goal
4.  $\{\neg q\}$  Negated Goal
5.  $\{p, q\}$  3,2
6.  $\{q\}$  5,1
7.  $\{\}$  6,4

18

## Example

Heads you win. Tails I lose. Show that you always win.

1.  $\{\neg h, y\}$  Premise
2.  $\{\neg t, \neg m\}$  Premise
3.  $\{h, t\}$  Premise
4.  $\{\neg h, \neg t\}$  Premise
5.  $\{m, y\}$  Premise
6.  $\{\neg m, \neg y\}$  Premise
7.  $\{\neg y\}$  Negated Goal
8.  $\{t, y\}$  3,1
9.  $\{\neg m, y\}$  8,2
10.  $\{y\}$  9,5
11.  $\{\}$  10,7

19

## Soundness and Completeness

A sentence is *provable* from a set of sentences by propositional resolution if and only if there is a derivation of the empty clause from the clausal form of  $\Delta \cup \{\neg \varphi\}$ .

Theorem: Propositional Resolution is sound and complete, i.e.  $\Delta \models \varphi$  if and only if  $\Delta \vdash \varphi$ .

20

## Two Finger Method

```

function tfm ( $\Delta$ )           ;  $\Delta$  is a linked list of clauses
{ var fast  $\leftarrow$   $\Delta$ ;
  var slow  $\leftarrow$   $\Delta$ ;
  do { if slow = [] then return failure;
     $\Delta \leftarrow \text{concat}(\Delta, \text{resolvents}(\text{first}(\text{fast}), \text{first}(\text{slow})))$ ;
    if {}  $\in \Delta$  then return  $\Delta$ ;
    if fast=slow then { fast  $\leftarrow$   $\Delta$ ; slow  $\leftarrow$  next(slow) }
    else fast  $\leftarrow$  next(fast) } }
  
```

```

function resolvents( $\Phi_1, \Phi_2$ )
{  $\Phi_1 - \{\varphi\} \cup \Phi_2 - \{\neg\varphi\} \mid \varphi \in \Phi_1$  and  $\neg\varphi \in \Phi_2$  }  $\cup$ 
{  $\Phi_1 - \{\neg\varphi\} \cup \Phi_2 - \{\varphi\} \mid \neg\varphi \in \Phi_1$  and  $\varphi \in \Phi_2$  }
  
```

21

## Two Finger Method in Operation

- |                            |                 |
|----------------------------|-----------------|
| 1. $\{p, q\}$ Premise      | 11. $\{r\}$ 2,6 |
| 2. $\{\neg p, r\}$ Premise | 12. $\{p\}$ 4,6 |
| 3. $\{\neg q, r\}$ Premise | 13. $\{q\}$ 1,7 |
| 4. $\{\neg r\}$ Premise    | 14. $\{r\}$ 6,7 |
| 5. $\{q, r\}$ 1,2          | 15. $\{p\}$ 1,8 |
| 6. $\{p, r\}$ 1,3          | 16. $\{r\}$ 5,8 |
| 7. $\{\neg p\}$ 2,4        | 17. $\{\}$ 4,9  |
| 8. $\{\neg q\}$ 3,4        |                 |
| 9. $\{r\}$ 3,5             |                 |
| 10. $\{q\}$ 4,5            |                 |

22

## TFM With Identical Clause Elimination

1.  $\{p, q\}$  Premise
2.  $\{\neg p, r\}$  Premise
3.  $\{\neg q, r\}$  Premise
4.  $\{\neg r\}$  Premise
5.  $\{q, r\}$  1,2
6.  $\{p, r\}$  1,3
7.  $\{\neg p\}$  2,4
8.  $\{\neg q\}$  3,4
9.  $\{r\}$  3,5
10.  $\{q\}$  4,5
11.  $\{p\}$  4,6
12.  $\{\}$  4,9

23

## TFM With ICE, Complement Detection

1.  $\{p, q\}$  Premise
2.  $\{\neg p, r\}$  Premise
3.  $\{\neg q, r\}$  Premise
4.  $\{\neg r\}$  Premise
5.  $\{q, r\}$  1,2
6.  $\{p, r\}$  1,3
7.  $\{\neg p\}$  2,4
8.  $\{\neg q\}$  3,4
9.  $\{r\}$  3,5
10.  $\{\}$  4,9

24

## Termination

Theorem: There is a resolution derivation of a conclusion from a set of premises if and only if there is a derivation using the two finger method.

Theorem: Propositional resolution using the two-finger method always terminates.

Proof: There are only finitely many clauses that can be constructed from a finite set of logical constants.

25

## Decidability of Propositional Entailment

Propositional resolution is a decision procedure for Propositional Logic.

Logical entailment for Propositional Logic is decidable.

Sadly, the problem in general is NP-complete.

26

## Horn Clauses

A *Horn clause* is a clause containing at most one positive literal.

Example:  $\{r, \neg p, \neg q\}$

Example:  $\{\neg p, \neg q, \neg r\}$

Example:  $\langle p \rangle$

Non-Example:  $\{q, r, \neg p\}$

NB: Every Horn clause can be written as a “rule”.

$$\{\neg p, \neg q, r\} \rightarrow p \wedge q \Rightarrow r$$

27

## Complexity

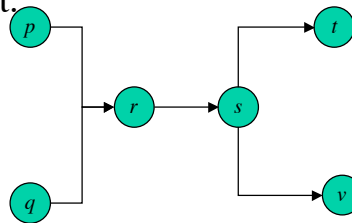
Good news: When a set of propositional sentences is Horn, satisfiability and, consequently, logical entailment can be decided in time linear in the size of the sentence set.

$$p \wedge q \Rightarrow r$$

$$r \Rightarrow s$$

$$s \Rightarrow t$$

$$s \Rightarrow v$$



Does  $\{p, q\} \models v$ ?

28

## Example for DP

- |     |                      |                      |
|-----|----------------------|----------------------|
| 1.  | $\{p, q\}$           | $p \vee q$           |
| 2.  | $\{p, \neg q\}$      | $p \vee \neg q$      |
| 3.  | $\{\neg p, q\}$      | $\neg p \vee q$      |
| 4.  | $\{\neg p, \neg q\}$ | $\neg p \vee \neg q$ |
| 5.  | $\{p\}$              | 1,2                  |
| 6.  | $\{q\}$              | 1,3                  |
| 7.  | $\{\neg q\}$         | 2,4                  |
| 8.  | $\{\neg p\}$         | 3,4                  |
| 9.  | $\{\}$               | 6,7                  |
| 10. | $\{\}$               | 5,8                  |

29

## Davis Putnam Procedure

```

function dp ( $\Delta$ )
  {for  $\varphi$  in vocabulary( $\Delta$ )
    do {var  $\Delta' \leftarrow \{\}$ ;
      for  $\Phi_1$  in  $\Delta$  for  $\Phi_2$  in  $\Delta$ 
        such that  $\varphi \in \Phi_1 \ \neg\varphi \in \Phi_2$ 
          do {var  $\Phi' \leftarrow \Phi_1 - \{\varphi\} \cup \Phi_2 - \{\neg\varphi\}$ ;
            if not tautology( $\Phi'$ ) then  $\Delta' \leftarrow \Delta' \cup \{\Phi'\}$ };
           $\Delta \leftarrow \Delta - \{\Phi \in \Delta \mid \varphi \in \Phi \text{ or } \neg\varphi \in \Phi\} \cup \Delta'$ ;
        return {if  $\{\} \in \Delta$  then unsatisfiable else satisfiable}}
  
```

```

function tautology( $\Phi$ )
  { $\varphi \in \Phi$  and  $\neg\varphi \in \Phi$ }
  
```

30

## Example Without DP

$\{p, q, r\}$	$\{q, r\}$	$\{p\}$	
$\{p, q, \neg r\}$	$\{q, \neg r\}$	$\{\neg p\}$	
$\{p, \neg q, r\}$	$\{\neg q, r\}$	$\{q\}$	
$\{p, \neg q, \neg r\}$	$\{\neg q, \neg r\}$	$\{\neg q\}$	
$\{\neg p, q, r\}$	$\{p, r\}$	$\{r\}$	
$\{\neg p, q, \neg r\}$	$\{p, \neg r\}$	$\{\neg r\}$	
$\{\neg p, \neg q, r\}$	$\{\neg p, r\}$		
$\{\neg p, \neg q, \neg r\}$	$\{\neg p, \neg r\}$	$\{\}$	
	$\{p, q\}$		
	$\{p, \neg q\}$		Cost =
	$\{\neg p, q\}$		378 resolutions
	$\{\neg p, \neg q\}$		

31

## Example With DP

$\{p, q, r\}$	$\{q, r\}$
$\{p, q, \neg r\}$	$\{q, \neg r\}$
$\{p, \neg q, r\}$	$\{\neg q, r\}$
$\{p, \neg q, \neg r\}$	$\{\neg q, \neg r\}$
$\{\neg p, q, r\}$	
$\{\neg p, q, \neg r\}$	$\{r\}$
$\{\neg p, \neg q, r\}$	$\{\neg r\}$
$\{\neg p, \neg q, \neg r\}$	
	$\{\}$

Cost = 16 + 4 + 1 = 21 resolutions

32

## Motivation for DPLL

DP can cause a quadratic expansion every time it is applied. This can easily exhaust space on large problems.

DPLL attacks this problem by sequentially solving smaller problems.

Basic idea: Choose a literal. Assume true, simplify clause set, and try to show satisfiable. Repeat for the negation of the literal. Good because we do not cross multiply the clause set.

33

## Davis Putnam Logemann Loveland

```
function dpll ( $\Delta$ )  
  {var  $\varphi$ ;  
    if  $\Delta = \{\}$  then return yes;  
    if  $\{\} \in \Delta$  then return no;  
     $\varphi \leftarrow$  choose vocabulary( $\Delta$ );  
    if dpll(simplify( $\Delta$ ,  $\varphi$ )) return yes  
    else return dpll(simplify( $\Delta$ ,  $\neg\varphi$ ))}
```

34

## Simplification

```
function simplify ( $\Delta$ ,  $\varphi$ )
  {var  $\Delta'$ ;
  for  $\Phi \in \Delta$ 
  do {if  $\varphi \in \Phi$  then skip
      else if negation( $\varphi$ )  $\in \Phi$ 
        then  $\Delta' \leftarrow \Delta' \cup \{\Phi - \{\textit{negation}(\varphi)\}\}$ 
      else  $\Delta' \leftarrow \Delta' \cup \{\Phi\}}$ 
```

Example:

$\textit{simplify}(\{\{p,q\},\{\neg p,r\},\{\neg r,s\}\}, p) = \{\{r\},\{\neg r,s\}\}$

35

## Comparisons

<b>Problem</b>	<b>Tautology</b>	<b>DP</b>	<b>DPLL</b>
<i>Prime</i>	30.00	0.00	0.00
<i>Prime4</i>	0.02	0.06	0.04
<i>Prime9</i>	18.94	2.98	0.51
<i>Prime10</i>	11.40	3.03	0.96
<i>Prime11</i>	28.11	2.98	0.51
<i>Prime16</i>	> 1 hour	*	9.15
<i>Prime17</i>	> 1 hour	*	3.87
Mkadder32	>> 1 hour	6.50	7.34
Mkadder42	>> 1 hour	22.95	46.86
Mkadder52	>> 1 hour	44.83	170.98
Mkadder53	>> 1 hour	38.27	250.16
Mkadder63	>> 1 hour	*	1186.4
Mkadder73	>> 1 hour	*	3759.9

36

## Coin Logic

Syntax:

The logical constants: quarters, nickels, dimes

Negation: coin upside down

Disjunction: stack of coins

Conjunction: set of stacks

*Almost exactly clausal form.*

Propositional Resolution:

Add two stacks,

deleting at most one pair of complementary coins.

37

## Example

Quarter means it is Monday.

Nickel means Mary loves Pat.

Dime means Mary loves Quincy.

[¬Nickel,Dime]: If Mary loves Pat, Mary loves Quincy.

[¬Quarter,Nickel,Dime]: Monday Mary loves P or Q.

[¬Nickel, ¬Dime]: Mary loves only one of the two.

[¬Quarter,Dime]: If Monday, Mary loves Quincy.

[¬Quarter,-Nickel]: If Monday, Mary does not love Pat.

38