Algebraic and Discrete Methods in Biology

Propositional Resolution

Nikolaj Popov and Tudor Jebelean

Research Institute for Symbolic Computation, Linz

popov@risc.uni-linz.ac.at

Propositional Resolution

Propositional resolution is a rule of inference. Applying iteratively the resolution rule in a suitable way allows us to decide whether a propositional formula is satisfiable.

It works only on expressions in clausal form. Before the rule can be applied, the premises and conclusions must be converted to this form. Fortunately, there is a simple procedure for making this conversion.

Propositional Resolution

Propositional resolution is a rule of inference. Applying iteratively the resolution rule in a suitable way allows us to decide whether a propositional formula is satisfiable.

It works only on expressions in clausal form. Before the rule can be applied, the premises and conclusions must be converted to this form. Fortunately, there is a simple procedure for making this conversion.

Propositional Resolution

Definition

A *literal* is an atom or the negation of an atom.
In the propositional calculus the atoms are the logical variables.

Definition

A clause expression is either a literal or a disjunction of literals.

If P and Q are logical variables, then the following are clause expressions:

P

$\neg P$

$\neg P \vee Q$

Propositional Resolution

Definition
A *literal* is an atom or the negation of an atom.
In the propositional calculus the atoms are the logical variables.

Definition
A clause expression is either a literal or a disjunction of literals.

If P and Q are logical variables, then the following are clause expressions:

P

$\neg P$

$\neg P \vee Q$

Propositional Resolution

Definition
A *literal* is an atom or the negation of an atom.
In the propositional calculus the atoms are the logical variables.

Definition
A clause expression is either a literal or a disjunction of literals.

If P and Q are logical variables, then the following are clause expressions:
P
$\neg P$
$\neg P \vee Q$

Propositional Resolution

Definition
A clause is the set of literals in a clause expression.

For example, the following sets are the clauses corresponding to the clause expressions above:
$\{P\}$
$\{\neg P\}$
$\{\neg P, Q\}$

Note that the empty set $\{\}$ is also a clause. It is equivalent to an empty disjunction and, therefore, is unsatisfiable.

Propositional Resolution

Definition
A clause is the set of literals in a clause expression.

For example, the following sets are the clauses corresponding to the clause expressions above:
$\{P\}$
$\{\neg P\}$
$\{\neg P, Q\}$

Note that the empty set $\{\}$ is also a clause. It is equivalent to an empty disjunction and, therefore, is unsatisfiable.

Propositional Resolution

Definition
A clause is the set of literals in a clause expression.

For example, the following sets are the clauses corresponding to the clause expressions above:
$\{P\}$
$\{\neg P\}$
$\{\neg P, Q\}$

Note that the empty set $\{\}$ is also a clause. It is equivalent to an empty disjunction and, therefore, is unsatisfiable.

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Step 1 **Use the laws:**

$$\varphi \Leftrightarrow \psi \;=\; (\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$$
$$\varphi \Rightarrow \psi \;=\; \neg\varphi \vee \psi$$

Step 2 **Use the laws:**

$$\neg(\neg\psi) \;=\; \psi$$
$$\neg(\varphi \vee \psi) \;=\; \neg\varphi \wedge \neg\psi$$
$$\neg(\varphi \wedge \psi) \;=\; \neg\varphi \vee \neg\psi$$

Step 3 **Use the law:**

$$(\varphi \wedge \psi) \vee \chi \;=\; (\varphi \vee \chi) \wedge (\psi \vee \chi)$$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Step 1 **Use the laws:**

$$\varphi \Leftrightarrow \psi \;=\; (\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$$
$$\varphi \Rightarrow \psi \;=\; \neg\varphi \vee \psi$$

Step 2 **Use the laws:**

$$\neg(\neg\psi) \;=\; \psi$$
$$\neg(\varphi \vee \psi) \;=\; \neg\varphi \wedge \neg\psi$$
$$\neg(\varphi \wedge \psi) \;=\; \neg\varphi \vee \neg\psi$$

Step 3 **Use the law:**

$$(\varphi \wedge \psi) \vee \chi \;=\; (\varphi \vee \chi) \wedge (\psi \vee \chi)$$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Step 1 **Use the laws:**

$$\varphi \Leftrightarrow \psi \; = \; (\varphi \Rightarrow \psi) \land (\psi \Rightarrow \varphi)$$
$$\varphi \Rightarrow \psi \; = \; \neg\varphi \lor \psi$$

Step 2 **Use the laws:**

$$\neg(\neg\psi) \; = \; \psi$$
$$\neg(\varphi \lor \psi) \; = \; \neg\varphi \land \neg\psi$$
$$\neg(\varphi \land \psi) \; = \; \neg\varphi \lor \neg\psi$$

Step 3 **Use the law:**

$$(\varphi \land \psi) \lor \chi \; = \; (\varphi \lor \chi) \land (\psi \lor \chi)$$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Transform the formula from *CNF* into *Clausal Form*

***Step 4* Use the law:**

$\varphi_1 \ \wedge \ \ldots \ \wedge \ \varphi_n \ = \ \{\varphi_1, \ \ldots \ , \varphi_n\}$

***Step 5* Use the law:**

$\{L_1 \vee \cdots \vee L_n \ , \ \ldots \ , \ L_m \vee \cdots \vee L_k\} \ = \ \{\{L_1, \ldots, L_n\} \ , \ \ldots \ \ , \ \{L_m, \ldots, L_k\}\}$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Transform the formula from *CNF* into *Clausal Form*

Step 4 **Use the law:**

$\varphi_1 \wedge \ldots \wedge \varphi_n = \{\varphi_1, \ldots, \varphi_n\}$

Step 5 **Use the law:**

$\{L_1 \vee \cdots \vee L_n, \ldots, L_m \vee \cdots \vee L_k\} = \{\{L_1, \ldots, L_n\}, \ldots, \{L_m, \ldots, L_k\}\}$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Transform the formula from *CNF* into *Clausal Form*

Step 4 **Use the law:**

$$\varphi_1 \wedge \ldots \wedge \varphi_n = \{\varphi_1, \ldots, \varphi_n\}$$

Step 5 **Use the law:**

$$\{L_1 \vee \cdots \vee L_n , \ldots , L_m \vee \cdots \vee L_k\} = \{\{L_1, \ldots, L_n\} , \ldots , \{L_m, \ldots, L_k\}\}$$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Example

Obtain *clausal form* for the formula $\varphi_1 \ \hat{=} \ (P \vee \neg Q) \Rightarrow R$

$\{\{\neg P, R\}, \{Q, R\}\}$

Example

Obtain *clausal form* for the formula $\varphi_2 \ \hat{=} \ (P \wedge (Q \Rightarrow R)) \Rightarrow S$

$\{\{\neg P, S\}, \{Q, S\}, \{\neg R, S\}\}$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Example

Obtain *clausal form* for the formula $\varphi_1 \doteq (P \vee \neg Q) \Rightarrow R$

$\{\{\neg P, R\}, \{Q, R\}\}$

Example

Obtain *clausal form* for the formula $\varphi_2 \doteq (P \wedge (Q \Rightarrow R)) \Rightarrow S$

$\{\{\neg P, S\}, \{Q, S\}, \{\neg R, S\}\}$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Example

Obtain *clausal form* for the formula $\varphi_1 \doteq (P \vee \neg Q) \Rightarrow R$

$\{\{\neg P, R\}, \{Q, R\}\}$

Example

Obtain *clausal form* for the formula $\varphi_2 \doteq (P \wedge (Q \Rightarrow R)) \Rightarrow S$

$\{\{\neg P, S\}, \{Q, S\}, \{\neg R, S\}\}$

Transformation into Normal Form ($\rightsquigarrow$ *CNF* $\rightsquigarrow$ *Clausal Form*)

Example

Obtain *clausal form* for the formula $\varphi_1 \ \hat{=}\ (P \vee \neg Q) \Rightarrow R$

$\{\{\neg P, R\}, \{Q, R\}\}$

Example

Obtain *clausal form* for the formula $\varphi_2 \ \hat{=}\ (P \wedge (Q \Rightarrow R)) \Rightarrow S$

$\{\{\neg P, S\}, \{Q, S\}, \{\neg R, S\}\}$

Propositional Resolution

Resolution

The idea: Suppose we know that P is true or Q is true, and suppose we also know that P is false or R is true.

$\{P, Q\}$
$\{\neg P, R\}$

One clause contains P, and the other contains $\neg P$.
If P is false, then by the first clause Q must be true.
If P is true, then, by the second clause, R must be true.

Since P must be either true or false, then it must be the case that Q is true or R is true.
In other words, we can cancel the P literals.

$$\frac{\{P, Q\} \quad \{\neg P, R\}}{\{Q, R\}}$$

Propositional Resolution

Resolution

The idea: Suppose we know that P is true or Q is true, and suppose we also know that P is false or R is true.

$\{P, Q\}$
$\{\neg P, R\}$

One clause contains P, and the other contains $\neg P$.
If P is false, then by the first clause Q must be true.
If P is true, then, by the second clause, R must be true.

Since P must be either true or false, then it must be the case that Q is true or R is true.
In other words, we can cancel the P literals.

$$\frac{\begin{array}{l}\{P, Q\}\\ \{\neg P, R\}\end{array}}{\{Q, R\}}$$

Propositional Resolution

Resolution

The idea: Suppose we know that P is true or Q is true, and suppose we also know that P is false or R is true.

$\{P, Q\}$
$\{\neg P, R\}$

One clause contains P, and the other contains $\neg P$.
If P is false, then by the first clause Q must be true.
If P is true, then, by the second clause, R must be true.

Since P must be either true or false, then it must be the case that Q is true or R is true.
In other words, we can cancel the P literals.

$$\frac{\{P, Q\} \quad \{\neg P, R\}}{\{Q, R\}}$$

Propositional Resolution

Resolution

The idea: Suppose we know that P is true or Q is true, and suppose we also know that P is false or R is true.

$\{P, Q\}$
$\{\neg P, R\}$

One clause contains P, and the other contains $\neg P$.
If P is false, then by the first clause Q must be true.
If P is true, then, by the second clause, R must be true.

Since P must be either true or false, then it must be the case that Q is true or R is true.
In other words, we can cancel the P literals.

$$\frac{\{P, Q\} \quad \{\neg P, R\}}{\{Q, R\}}$$

Propositional Resolution

More generally, given a clause containing a literal Q and another clause containing the literal $\neg Q$, we can infer the clause consisting of all the literals of both clauses without the complementary pair.
This rule of inference is called propositional resolution.

$$\frac{\begin{array}{l}\{P_1, \ \ldots \ , P_n, Q\} \\ \{\neg Q, R_1, \ \ldots \ , R_m\}\end{array}}{\{P_1, \ldots, P_n, , R_1, \ldots, R_m\}}$$

If either of the clauses is a singleton set, we see that the number of literals in the result is less than the number of literals in the other clause.

From the clause $\{\neg P, Q\}$ and the singleton clause $\{P\}$, we can derive the singleton clause $\{Q\}$.

$$\frac{\begin{array}{l}\{\neg P, Q\} \\ \{P\}\end{array}}{\{Q\}}$$

Note the similarity between this deduction and that of Modus Ponens.

Propositional Resolution

More generally, given a clause containing a literal Q and another clause containing the literal $\neg Q$, we can infer the clause consisting of all the literals of both clauses without the complementary pair. This rule of inference is called propositional resolution.

$$\frac{\begin{array}{l}\{P_1,\ \ldots\ , P_n, Q\}\\ \{\neg Q, R_1,\ \ldots\ , R_m\}\end{array}}{\{P_1, \ldots, P_n, , R_1, \ldots, R_m\}}$$

If either of the clauses is a singleton set, we see that the number of literals in the result is less than the number of literals in the other clause.

From the clause $\{\neg P, Q\}$ and the singleton clause $\{P\}$, we can derive the singleton clause $\{Q\}$.

$$\frac{\begin{array}{l}\{\neg P, Q\}\\ \{P\}\end{array}}{\{Q\}}$$

Note the similarity between this deduction and that of Modus Ponens.

Propositional Resolution

More generally, given a clause containing a literal Q and another clause containing the literal $\neg Q$, we can infer the clause consisting of all the literals of both clauses without the complementary pair. This rule of inference is called propositional resolution.

$$\frac{\begin{array}{l}\{P_1,\ \ldots\ ,P_n,Q\}\\ \{\neg Q,R_1,\ \ldots\ ,R_m\}\end{array}}{\{P_1,\ldots,P_n,,R_1,\ldots,R_m\}}$$

If either of the clauses is a singleton set, we see that the number of literals in the result is less than the number of literals in the other clause.

From the clause $\{\neg P, Q\}$ and the singleton clause $\{P\}$, we can derive the singleton clause $\{Q\}$.

$$\frac{\begin{array}{l}\{\neg P,Q\}\\ \{P\}\end{array}}{\{Q\}}$$

Note the similarity between this deduction and that of Modus Ponens.

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\{P\} \quad \{\neg P\}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\{\neg P, Q\} \quad \{P, \neg Q\}}{\{\}}$$ **Wrong!**

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\begin{array}{l}\{P\}\\ \{\neg P\}\end{array}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\begin{array}{l}\{\neg P, Q\}\\ \{P, \neg Q\}\end{array}}{\{\}}$$ **Wrong!**

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\begin{array}{l}\{P\}\\ \{\neg P\}\end{array}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\begin{array}{l}\{\neg P, Q\}\\ \{P, \neg Q\}\end{array}}{\{\}} \quad \textbf{Wrong!}$$

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\begin{array}{l}\{P\}\\ \{\neg P\}\end{array}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\begin{array}{l}\{\neg P, Q\}\\ \{P, \neg Q\}\end{array}}{\{\}}$$ **Wrong!**

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\{P\} \quad \{\neg P\}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\{\neg P, Q\} \quad \{P, \neg Q\}}{\{\neg P, P\}}$$ **Correct!**

Propositional Resolution

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all.

$$\frac{\begin{array}{l}\{P\}\\ \{\neg P\}\end{array}}{\{\}}$$

The derivation of the empty clause means that the database contains a contradiction.

Note that in resolving two clauses, only one pair of literals may be resolved at a time, even though there are multiple resolvable pairs.

For example, the following is not a legal application of propositional resolution.

$$\frac{\begin{array}{l}\{\neg P, Q\}\\ \{P, \neg Q\}\end{array}}{\{Q, \neg Q\}}$$ **Also correct!**

Propositional Resolution

To determine whether a formula ψ is a logical consequence of a set of formulae $\{\varphi_1, \ldots \varphi_n\}$

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$

we add the formula $\neg\psi$ to the set $\{\varphi_1, \ldots \varphi_n\}$,

$\{\varphi_1, \ldots \varphi_n, \neg\psi\}$

transform in clausal form and try to derive the empty clause.

Propositional Resolution

To determine whether a formula ψ is a logical consequence of a set of formulae $\{\varphi_1, \ldots \varphi_n\}$

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$

we add the formula $\neg\psi$ to the set $\{\varphi_1, \ldots \varphi_n\}$,

$\{\varphi_1, \ldots \varphi_n, \neg\psi\}$

transform in clausal form and try to derive the empty clause.

Propositional Resolution

To determine whether a formula ψ is a logical consequence of a set of formulae $\{\varphi_1, \dots \varphi_n\}$

that is: $\varphi_1 \wedge \dots \wedge \varphi_n \models \psi$

we add the formula $\neg\psi$ to the set $\{\varphi_1, \dots \varphi_n\}$,

$\{\varphi_1, \dots \varphi_n, \neg\psi\}$

transform in clausal form and try to derive the empty clause.

Propositional Resolution

To determine whether a formula ψ is a logical consequence of a set of formulae $\{\varphi_1, \ldots \varphi_n\}$

that is: $\varphi_1 \ \wedge \ \ldots \ \wedge \ \varphi_n \ \models \psi$

we add the formula $\neg\psi$ to the set $\{\varphi_1, \ldots \varphi_n\}$,

$\{\varphi_1, \ldots \varphi_n, \neg\psi\}$

transform in clausal form and try to derive the empty clause.

Propositional Resolution

If the empty clause is derived, the set of formulae $\{\varphi_1, \ldots \varphi_n, \neg\psi\}$ is unsatisfiable (or contradictory), and hence ψ is a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$.

If, on the other hand, the empty clause cannot be derived, and the resolution rule cannot be applied to derive any more new clauses, ψ is not a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \not\models \psi$.

Propositional Resolution

If the empty clause is derived, the set of formulae $\{\varphi_1, \ldots \varphi_n, \neg\psi\}$ is unsatisfiable (or contradictory), and hence ψ is a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$.

If, on the other hand, the empty clause cannot be derived, and the resolution rule cannot be applied to derive any more new clauses, ψ is not a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \not\models \psi$.

Propositional Resolution

If the empty clause is derived, the set of formulae $\{\varphi_1, \ldots \varphi_n, \neg\psi\}$ is unsatisfiable (or contradictory), and hence ψ is a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$.

If, on the other hand, the empty clause cannot be derived, and the resolution rule cannot be applied to derive any more new clauses, ψ is not a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \not\models \psi$.

Propositional Resolution

If the empty clause is derived, the set of formulae $\{\varphi_1, \ldots \varphi_n, \neg\psi\}$ is unsatisfiable (or contradictory), and hence ψ is a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \models \psi$.

If, on the other hand, the empty clause cannot be derived, and the resolution rule cannot be applied to derive any more new clauses, ψ is not a logical consequence of $\{\varphi_1, \ldots \varphi_n\}$,

that is: $\varphi_1 \wedge \ldots \wedge \varphi_n \not\models \psi$.

Propositional Resolution

Example

Show that $(P \Rightarrow Q) \; \land \; (Q \Rightarrow R) \; \models \; (P \Rightarrow R)$.

Example

Show that $\models \; (A \land (A \Rightarrow B)) \Rightarrow B$.

Example

Show that $C \; \models \; C \; \land \; ((A \land (A \Rightarrow B)) \Rightarrow B)$.

Propositional Resolution

Example

Show that $(P \Rightarrow Q) \ \wedge \ (Q \Rightarrow R) \ \models \ (P \Rightarrow R)$.

Example

Show that $\models \ (A \wedge (A \Rightarrow B)) \Rightarrow B$.

Example

Show that $C \ \models \ C \ \wedge \ ((A \wedge (A \Rightarrow B)) \Rightarrow B)$.

Propositional Resolution

Example

Show that $(P \Rightarrow Q) \ \wedge \ (Q \Rightarrow R) \ \models \ (P \Rightarrow R)$.

Example

Show that $\models \ (A \wedge (A \Rightarrow B)) \Rightarrow B$.

Example

Show that $C \ \models \ C \ \wedge \ ((A \wedge (A \Rightarrow B)) \Rightarrow B)$.