INF3170 / INF4171
Notes on Resolution

Andreas Nakkerud

Autumn 2015

1 Introduction

This is a short description of the Resolution calculus for propositional logic, and for first order logic. We will only outline the proofs, as only the general ideas and concepts are part of the course syllabus.

2 Preliminaries

2.1 Conjunctive Normal Form

Every propositional logic formula is equivalent to formulas on both Conjunctive Normal Form (CNF) and Disjunctive Normal Form (DNF). We will only need the Conjunctive Normal Form.

Definition 1 (Literals and Clauses)**.** Let ϕ be an atomic formula (propositional or first order). ϕ and $\neg\phi$ are **literals**. A **clause** is a disjunction of literals. If C_i is a clause of n_i literals, and L_i^j are the literals of C_i, we write $C_i = L_i^1 \vee \ldots \vee L_i^{n_i}$ or $C_i = \{L_i^1, \ldots, L_i^{n_i}\}$.

The complement $\overline{L}$ of a literal L is defined so that if ϕ is atomic, $\overline{\phi} = \neg\phi$ and $\overline{\neg\phi} = \phi$.

A literal containing no variables is called a **ground literal**. A clause of ground literals is called a **ground clause**.

From Definition 1 we see that a set of literals is read as a disjunction. It is often useful to write sets in place of disjunctions.

Note that if a formula only contains ground literals, it is essentially a propositional formula, since we can create one propositional variable for each ground literal.

Definition 2 (Conjunctive Normal Form)**.** A formula ϕ is on **Conjunctive Normal Form (CNF)** if it is a conjunction of clauses. If C_i are clauses, we usually write ϕ as $\{C_1, \ldots, C_m\}$.

Set notation is justified, since both $\wedge$ and $\vee$ are idempotent. In Definition 2, a set of sets of literals is read as a conjunction of disjunctions. This is common when it is given that the formulas we work with are on CNF, but the notation may be ambiguous and should be used with care.

Note also that a first order formula can be on CNF if it is quantifier free (open).

Example 3. The formula

$$\phi = (Px \vee Rxy \vee Pa) \wedge (\neg Pa \vee Ryz \vee \neg Rxx) \wedge (\neg Pb \vee Pc)$$

is on CNF, and we can write it as

$$\{\{Px, Rxy, Pa\}, \{\neg Pa, Ryz, \neg Rxx\}, \{\neg Pb, Pc\}\}.$$

The following result can easily be proven by induction on (quantifier free) formulas.

Theorem 4. *All (quantifier free) formulas are equivalent to a formula on CNF.*

2.2 Prenex Normal Form

Prenex Normal Form is a normal form for first order formulas. It limits the placement of quantifiers.

Definition 5 (Prenex Normal Form)**.** A first order formula is on **Prenex Normal Form (PNF)** if it consists of a string of quantifiers followed by a quantifier free (open) formula. The string of quantifiers is referred to as the **prefix**, and the quantifier free part as the **matrix**.

In order to bring a formula onto PNF, we first eliminate all occurrences of $\rightarrow$ using the identity

$$\phi \rightarrow \psi \quad \iff \quad \neg\phi \vee \psi.$$

We then use the following identities to bring all quantifiers to the beginning of the formula.

$$\begin{array}{rcl@{\qquad}rcl}
\neg\forall x\phi & \iff & \exists x\neg\phi & \neg\exists x\phi & \iff & \forall x\neg\phi \\
\forall x\phi \wedge \psi & \iff & \forall x(\phi \wedge \psi) & \exists x\phi \wedge \psi & \iff & \exists x(\phi \wedge \psi) \\
\forall x\phi \vee \psi & \iff & \forall x(\phi \vee \psi) & \exists x\phi \vee \psi & \iff & \exists x(\phi \vee \psi)
\end{array}$$

Note that we must make sure not to bind free variables when applying the above identities. We avoid this by renaming all bound variables so that they all have names not used elsewhere in the formula. Note also that $\forall x\phi \wedge \psi = \psi \wedge \forall x\phi$.

Theorem 6. *Every first order formula is equivalent to a formula on PNF. Furthermore, every formula on PNF is equivalent to a formula on PNF with its matrix on CNF.*

2.3 Skolem Normal Form

After bringing a formula to Prenex Normal Form, we can eliminate all existential quantifiers to bring the formula to Skolem Normal Form. The Skolem Normal Form preserves satisfiability.

Example 7. Consider the formula $\forall x\exists y\forall z\exists w R(x, y, z, w)$. When satisfying this formula, our choice for w may depend on the choice of all the other variables. We can therefore replace w by some function $f_w(x, y, z)$. We then make a similar transformation for y.

$$\begin{gathered}\forall x\exists y\forall z\exists w R(x, y, z, w)\\ \Downarrow\\ \forall x\exists y\forall z R(x, y, z, f(x, y, z))\\ \Downarrow\\ \forall x\forall z R(x, f_y(x), z, f_w(x, f_y(x), z))\end{gathered}$$

We can interpret f_w and f_y so that they return the required values for the above formula to be true, but not every interpretation needs to have this property. Thus, Skolem Normal Forms need not preserve validity.

We will not go into details about the Skolem Normal Form. You can read more about this normal form in van Dalen's Logic and Structure.

2.4 Unification

Unification in the process of making two terms equal under a common substitution.

Definition 8 (Unification)**.** Let t and s be terms, and σ a substitution. We call σ a **unifier** for t and s, and say that σ unifies t and s if $t\sigma$ and $s\sigma$ are the same terms.

If σ is such that every unifier τ for t and s can be written as $\tau = \sigma\tau'$, then σ is a Most General Unifier (MGU) for t and s.

Example 9. Let $t = f(x, g(y))$ and $s = f(f(z, z), w)$, then

$$\sigma = \{f(z, z)/x, g(y)/w\}$$

is an MGU for t and s.

We will use unifiers to make atomic formulas equal. Therefore, we need to simultaneously unify multiple pairs of terms. The following algorithm achieves this.

2.4.1 Unification algorithm

This algorithm is due to Martelli and Montanari.[1]

Let G be a finite set of potential (desired) equations

$$G = \{s_1 \doteq t_1, \dots, s_n \doteq t_n\},$$

where s_i and t_j are terms.

We transform this set into an equivalent set of equations

$$\{x_1 \doteq u_1, \dots, x_m \doteq u_m\},$$

which can be read as the substitution $\{u_1/x_1, \dots, u_m/x_m\}$. The variables x_i must be distinct, and may not occur in the terms u_j.

The set G is modified step by step according to a set of rules. At any step, the unification may fail. In that case, there is no substitution that simultaneously unifies each pair in the original G. The modification rules are as follows:

delete Pairs of equal terms puts no further restrictions on the unification.

$$G \cup \{t \doteq t\} \quad \Rightarrow \quad G$$

decompose Composite terms are equal if their parts are equal.

$$G \cup \{f(s_0, \dots, s_k) \doteq f(t_0, \dots, t_k)\} \quad \Rightarrow \quad G \cup \{s_0 \doteq t_0, \dots, s_k \doteq t_k\}$$

conflict Composite terms constructed with different function symbols cannot be unified. If this is required, the unification fails.

$$G \cup \{f(s_0, \dots, s_k) \doteq g(t_0, \dots, t_k)\} \quad \Rightarrow \quad \bot \qquad \text{if } f \neq g$$

[1] https://en.wikipedia.org/wiki/Unification_(computer_science)

swap Left side of potential equations must be variables.

$$G \cup \{t \doteq x\} \quad \Rightarrow \quad G \cup \{x \doteq t\} \qquad t \text{ is not a variable}$$

eliminate Account a potential equations by substituting in all other equations.

$$G \cup \{x \doteq t\} \quad \Rightarrow \quad G[t/x] \cup \{x \doteq t\} \qquad \text{if } x \notin vars(t),\ x \in vars(G)$$

check A variable cannot be replaced by a term containing that variable.

$$G \cup \{x \doteq f(s_0, \dots, s_k)\} \quad \Rightarrow \quad \bot \qquad \text{if } x \in vars(f(s_0, \dots, s_k))$$

Theorem 10. *Iterating the above steps on a set G of potential equations yields an MGU for G, or fails if no unifier exists.*

3 Resolution for Propositional Logic

Resolution for Propositional Logic is a calculus for checking if a propositional formula is satisfiable. We are going to assume the formula is on CNF. Recall that if F is on CNF, we write

$$F = \{\{L_1^1, \dots, L_1^{n_1}\}, \dots, \{L_m^1, \dots, L_m^{n_m}\}\}.$$

A valuation v satisfies a clause $\{L_i \mid 1 \leq i \leq n\}$ if it satisfies at least one L_i. A valuation satisfies a formula $F = \{C_i \mid 1 \leq i \leq n\}$ if it simultaneously satisfies each C_i. A clause or set of clauses is satisfiable if some valuation satisfies it.

Note that an empty disjunction is considered contradictory, while an empty conjunction is considered valid.

Definition 11 (Ground resolvent)**.** Let C_1 and C_2 be ground clauses. C is a **ground resolvent** for C_1 and C_2 if for some ground literal L, $L \in C_1$ and $\overline{L} \in C_2$, and

$$C = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{\overline{L}\})$$

Lemma 12. *Let $C_1 = \{A_1, \dots, A_n, L\}$ and $C_2 = \{B_1, \dots, B_m, \overline{L}\}$ be ground clauses, where A_i, B_j and L are ground literals. The ground resolvent for C_1 and C_2 is $C = \{A_1, \dots A_n, B_1, \dots, B_m\}$. If a valuation v satisfies C_1 and C_2, then v also satisfies C.*

Proof. Exercise to the reader. □

Definition 13 (Ground resolution)**.** Let S be a set of ground clauses. The **ground resolution of** S, denoted $\mathcal{R}(S)$, is the set S together with all the ground resolvents of all pairs of clauses in S.

Lemma 14. *A valuation v satisfies S if and only if it satisfies $\mathcal{R}(S)$.*

Proof. The only if-direction follows immediately from Lemma **??**. For the other direction, observe that $S \subseteq \mathcal{R}(S)$. □

We can now iterate the ground resolution.

Definition 15 (n-th resolution)**.** The n-th resolution of S, denoted $\mathcal{R}^n(S)$, is defined as follows:

$$\begin{aligned}\mathcal{R}^0(S) &= S\\ \mathcal{R}^{n+1}(S) &= \mathcal{R}(\mathcal{R}^n(S)).\end{aligned}$$

The resolution process must terminate, since the number of distinct clauses that can be created from the literals in S is finite.

Theorem 16. *If for some n, $\emptyset \in \mathcal{R}^n(S)$, then S is unsatisfiable. Otherwise, S is satisfiable.*

Proof. Using induction on $\mathcal{R}^n$, we can show that v satisfies S if and only if it satisfies $\mathcal{R}^n(S)$. Thus, if $\emptyset \in \mathcal{R}^n(S)$ for some n, S is unsatisfiable, since no valuation satisfies $\emptyset$.

If there is no n such that $\emptyset \in \mathcal{R}^n(S)$, let k be the lowest number such that $\mathcal{R}^{k+1}(S) = \mathcal{R}^k(S)$. Let $P_1, \ldots, P_m$ be all the atomic formulas occurring in $\mathcal{R}^k(S)$. We now define recursively the set M so that $M_0 = \emptyset$, and for each $0 \leq i < m$, M_{j+1} is the set $M_j \cup \{P_{j+1}\}$ unless some clause in $\mathcal{R}^k(S)$ consists only of complements of literals in $M_j \cup \{P_{j+1}\}$, in which case, $M_{j+1} = M_j \cup \{\neg P_{j+1}\}$. We now define a valuation v, such that $v(P_i) = 1$ if and only if $P_i \in M_m$. (It follows that $v(L) = 1$ for all $L \in M_m$.)

Claim: v satisfies S. If not, there is a least $0 < j \leq m$ such that some clause $C \in \mathcal{R}^k(S)$ contains only complements of literals in M_j. This is only possible if C contains only literals in $M_{j-1} \cup \{P_j\}$, and thus, $M_j = M_{j-1} \cup \{\neg P_j\}$. By j being least, we know that C contains P_j, but since $M_j = M_{j-1} \cup \{\neg P_j\}$, and by j being least, some clause $D \in \mathcal{R}^k(S)$ must contain only literals in M_j, including $\neg P_j$. Since $P_j \in C$ and $\neg P_j \in D$, we can create the resolvent of B of C and D. This resolvent will either be $\bot$, which it cannot be by assumption, or contain only complements of M_{j-1}, which contradicts j being least. □

We can now use the n-th resolvent to define a calculus for determining whether or not a CNF formula is a contradiction. Theorem 17 can then be restated as

Theorem 17. *Calculating the n-th resolvent to fix-point constitutes a sound and complete calculus for CNF unsatisfiability.*

4 Resolution for First Order Logic

Resolution for First Order Logic works much the same as for Propositional Logic. One of the major differences is that we can no longer assume the resolution operation has a fix-point.

We assume that the input formula is always on Skolem and Prenex Normal Form, with the matrix on Conjunctive Normal Form. We will make use of the fact that $\forall$ distributes over $\wedge$:

$$\forall x_1 \dots x_n (C_1 \wedge \dots \wedge C_m) \quad \Leftrightarrow \quad \forall x_1 \dots x_n C_1 \wedge \dots \wedge \forall x_1 \dots x_n C_m.$$

Since we only consider close formulas, and we only have universal quantification, we will no longer write out the quantifiers. We can therefore adopt the notation of the previous sections:

$$\phi = \{C_1, \dots, C_m\}.$$

Note that $\mathcal{M} \models \forall \vec{x} \phi$ if and only if $\mathcal{M} \models \forall \vec{x} C_j$ for each j.

We now extend the notion of ground resolvents.

Definition 18 (Resolvent)**.** Let C_1 and C_2 be clauses. C is a **resolvent** for C_1 and C_2 if

1. $L_1 \in C_1$ and $L_2 \in C_2$ are literals,
2. σ is an MGU of L_1 and $\overline{L}_2$, and
3. $C = [(C_1 \setminus \{L_1\}) \cup (C_2 \setminus \{L_2\})]\sigma$.

Lemma 19. *Let $\mathcal{M}$ be some model. If $\mathcal{M} \models C_1, C_2$, and C is a resolvent of C_1 and C_2, then $\mathcal{M} \models C$.*

Proof. Left as an exercise. Note that $\mathcal{M} \models \forall x \phi$ if and only if $\mathcal{M} \models \phi[t/x]$ for any term t, and that $\mathcal{M} \models \forall x \phi(x)$ if and only if $\mathcal{M} \models \forall y \phi(y)$ (where $y \notin FV(\phi(x))$). Remember that each variable is universally quantified. □

We now define the resolution $\mathcal{R}$ and n-th resolution $\mathcal{R}^n$ in exactly the same way as before. We prove a set of clauses S to be unsatisfiable if and only if $\mathcal{R}^n(S)$ contains $\bot$.

Since validity in First Order Logic is in general undecidable, we cannot expect to have the guarantee of termination we had for Propositional Logic. Indeed, there is no guarantee that $R^n(S)$ will ever reach a fix-point for a satisfiable S. As for Natural Deduction and LK, we have soundness and completeness, but not decidability.