COMP360 Automated Theorem Proving - Lecture Notes

Joomy Korkut

Spring 2016

1 Conjunctive Normal Form

- A *literal* is a propositional symbol P or its negation $\neg P$.
- A *clause* is a disjunction of literals. e.g. $P \vee (\neg Q) \vee R$ is a clause. Note that the empty clause $\square$ is a clause, but it is always false since it has no true element.
- A *formula* is the a disjunction of clauses. $P \wedge (Q \vee (\neg R))$ is a formula, consisting of the clauses P and $Q \vee (\neg R)$. Note that the empty formula $\emptyset$ is a formula, but it is always true since it has no false element.
- For conversion of a proposition into CNF, you can use the following logical equivalencies, which we know to be true in classical logic:
  - Implication : $(P \Rightarrow Q) \iff (\neg P) \vee Q$
  - Double negation : $P \iff \neg(\neg P)$
  - De Morgan's law : $\neg(P \vee Q) \iff (\neg P) \wedge (\neg Q)$
  - De Morgan's law : $\neg(P \wedge Q) \iff (\neg P) \vee (\neg Q)$
  - Distributive law : $P \vee (Q \wedge R) \iff (P \vee Q) \wedge (P \vee R)$
  - Distributive law : $P \wedge (Q \vee R) \iff (P \wedge Q) \vee (P \wedge R)$

Example 1. *Conversion the following proposition to CNF:* $(A \wedge B) \vee (C \wedge D) \vee E$.

Solution:

$$\begin{aligned}
&(A \wedge B) \vee (C \wedge D) \vee E \\
&= (A \vee (C \wedge D)) \wedge (B \vee (C \wedge D)) \vee E \quad \text{(by distributive law 1 $\Rightarrow$ flipped)} \\
&= E \vee (A \vee (C \wedge D)) \wedge (B \vee (C \wedge D)) \quad \text{(flipping)} \\
&= (E \vee (A \vee (C \wedge D))) \wedge (E \vee (B \vee (C \wedge D))) \quad \text{(by distributive law 1 $\Rightarrow$)} \\
&= (E \vee ((A \vee C) \wedge (A \vee D)) \wedge (E \vee (B \vee (C \wedge D))) \quad \text{(by distributive law 1 $\Rightarrow$)} \\
&= ((E \vee A \vee C) \wedge (E \vee A \vee D)) \wedge (E \vee (B \vee (C \wedge D))) \quad \text{(by distributive law 1 $\Rightarrow$)} \\
&= ((E \vee A \vee C) \wedge (E \vee A \vee D)) \wedge (E \vee ((B \vee C) \wedge (B \vee D))) \quad \text{(by distributive law 1 $\Rightarrow$)} \\
&= ((E \vee A \vee C) \wedge (E \vee A \vee D)) \wedge (E \vee B \vee C) \wedge (E \vee B \vee D))) \quad \text{(by distributive law 1 $\Rightarrow$)}
\end{aligned}$$

The resulting proposition is a conjunction of clauses, hence it is in CNF. We can also represent this as a set of sets, where the inner sets represent clauses (disjunction) and the outer set represents the formula (conjunction). So the result we reached above would be represented as $\{\{E, A, C\}, \{E, A, D\}, \{E, B, C\}, \{E, B, D\}\}$. This is the representation we are going to use in resolution.

Example 2. *Convert the following proposition to CNF:* $\neg(((A \Rightarrow B) \wedge A) \Rightarrow B)$.

Solution:

$$\begin{aligned}
&\neg(((A \Rightarrow B) \wedge A) \Rightarrow B) \\
&= \neg((((\neg A) \vee B) \wedge A) \Rightarrow B) \quad \text{(by implication)} \\
&= \neg(\neg(((\neg A) \vee B) \wedge A) \vee B) \quad \text{(by implication)} \\
&= (((\neg A) \vee B) \wedge A) \wedge (\neg B) \quad \text{(by De Morgan's 1 } \Rightarrow)
\end{aligned}$$

Once again, we reached a conjunction of clauses. Let's write this in set representation: $\{\{\neg A, B\}, \{A\}, \{\neg B\}\}$.

1.1 CNF Conversion Algorithm

The input to our algorithm is a proposition. The output is either a proposition in CNF, or a formula in our set notation.

1. Go through the proposition and recursively apply to implication rule in the $\Rightarrow$ direction.
2. Go through the resulting proposition and recursively apply the double negation rule and De Morgan's laws in the $\Rightarrow$ direction.
3. Go through the resulting proposition and look for $\vee$ connectives. When you see $P \vee Q$, recursively apply step 3 to P and Q and get P' and Q'. Apply the distributive law in the $\Rightarrow$ direction to $P' \vee Q'$. Recursively apply step to all immediate subformulae of other connectives.
4. As a result, you will get a proposition in CNF.

   It can be just a clause, which means it does not have any $\wedge$ in it, in that case, represent the clause in a set S and your formula's set representation will be $\{S\}$.

   Otherwise it will have $\wedge$ as the outermost (principal) connective. For $P \wedge Q$, recursively apply step 4 to P and Q and get two sets S_P, S_Q. Your result will be $S_P \cup S_Q$.

2 Resolution

2.1 Resolution rule

Let C_1, C_2 be clauses (sets of disjunctions of literals) such that $P \in C_1$ and $(\neg P) \in C_2$. The resolution rule states that we can deduce $(C_1 \cup C_2) \setminus \{P, \neg P\}$ from this. What we deduce from the rule is called a *resolvent*. In a more formal notation:

$$\frac{C_1 \qquad C_2}{(C_1 \cup C_2) \setminus \{P, \neg P\}} \ \text{where } P \in C_1, (\neg P) \in C_2$$

For example, if we have the clauses $\{P, \neg Q\}$ and $\{Q, R\}$, the resolution rule allows us to deduce $\{P, R\}$. In propositional notation, this means we are deducing $P \vee R$ from $P \vee (\neg Q)$ and $Q \vee R$.

☞ Note that the resolution rule is not destructive. Even though the resolution rule creates new clauses, it does not require you to remove the old ones. You can resolve other clauses with the ones you used before.

2.2 Resolution refutation

Resolution refutation is a contradiction based proof method that only has one rule. The other rules are hidden in the formatting rules when we are converting the input to CNF. If we want to show $\Gamma \vdash \Delta$, we negate the elements of Δ, convert Γs and negated Δs to CNF and then apply the resolution rule until we reach the clause $\square$, or until we cannot apply the rule anymore.

Example 3. *Prove the following propositional sequent with resolution:* $\vdash ((A \Rightarrow B) \wedge A) \Rightarrow B$

Solution: Since the right side of $\vdash$ only has one element, we negate it and convert it to CNF. The negated version is $\neg(((A \Rightarrow B) \wedge A) \Rightarrow B)$. We already did the CNF conversion of this as an example before, so we get the set representation $\{\{\neg A, B\}, \{A\}, \{\neg B\}\}$. Let's apply the resolution rule now.

$$\begin{aligned}
&\{\{\neg A, B\}, \{A\}, \{\neg B\}\} \\
&\{\{B\}, \{\neg B\}\} \quad \text{(by resolving } \{\neg A, B\} \text{ and } \{A\}) \\
&\{\square\}
\end{aligned}$$

Let's do another example. e.g. $(C \vee (A \wedge B)) \vdash ((C \vee A) \wedge (C \vee B))$. Let's convert the left hand side of $\vdash$ to CNF. By applying the distributive law, the left side becomes $(C \vee A) \wedge (C \vee B)$. Now let's negate the right hand side of $\vdash$ and convert it to CNF.

$$\begin{aligned}
&\neg(((C \vee A) \wedge (C \vee B))) \\
&= (\neg(C \vee A)) \vee (\neg(C \vee B)) \quad \text{(by De Morgan's 1 } \Rightarrow) \\
&= ((\neg C) \wedge (\neg A)) \vee (\neg(C \vee B)) \quad \text{(by De Morgan's 1 } \Rightarrow) \\
&= ((\neg C) \wedge (\neg A)) \vee ((\neg C) \wedge (\neg B)) \quad \text{(by De Morgan's 1 } \Rightarrow) \\
&= (\neg C) \wedge ((\neg A) \vee (\neg B)) \quad \text{(by distributive law 1 } \Leftarrow)
\end{aligned}$$

So for the right hand side, we have $\{\{\neg C\}, \{\neg A, \neg B\}\}$. We already had $\{\{C, A\}, \{C, B\}\}$ for the left hand side. Now we run the resolution rule on the union of these two:

$$\begin{aligned}
&\{\{C, A\}, \{C, B\}, \{\neg C\}, \{\neg A, \neg B\}\} \\
&\{\{C, \neg B\}, \{C, B\}, \{\neg C\}\} \quad \text{(by resolving } \{C, A\} \text{ and } \{\neg A, \neg B\}) \\
&\{\{C\}, \{\neg C\}\} \quad \text{(by resolving } \{C, \neg B\} \text{ and } \{C, B\}) \\
&\{\square\}
\end{aligned}$$

Observe that if you follow a different order of resolution, you will have to use one of the clauses more than one time, and that is allowed. You can see the reused clause below in bold.

$$\{\{C, A\}, \{C, B\}, \{\neg C\}, \{\neg A, \neg B\}\}$$
$$\{\{A\}, \{C, B\}, \{\neg \mathbf{C}\}, \{\neg A, \neg B\}\} \quad \text{(by resolving } \{C, A\} \text{ and } \{\neg C\}\text{)}$$
$$\{\{A\}, \{B\}, \{\neg A, \neg B\}\} \quad \text{(by resolving } \{C, B\} \text{ and } \{\neg C\}\text{)}$$
$$\{\{B\}, \{\neg B\}\} \quad \text{(by resolving } \{A\} \text{ and } \{\neg A, \neg B\}\text{)}$$
$$\{\Box\}$$

2.3 Resolution Algorithm

The input to our algorithm is a formula in our set notation. O is the set of old/used clauses. N is the set of new/unused clauses.

1. Pick $C_1 \in N$, resolve with all $C_2 \in O \cup N$. Let k be the resolvent.
2. If $k \notin O \cup N$, add k to N.
3. Remove C_1 from N an put it into O.
4. Return the proof if you have $\Box$.
5. If N is not empty, go to step 1. If N is empty, conclude that the formula cannot be proved.

3 First-order logic

3.1 Prenex normal form

Prenex normal form is the form of a formula where we have all the universal and existential quantifiers *in the beginning* of the formula. So if we have $\forall x(\exists y R(y)) \vee (\exists z(S(z)) \Rightarrow T(x))$, we want to get $\forall x \exists y \forall z(R(y) \vee (S(z) \Rightarrow T(x)))$ as a result.

Here are the conversion rules:

- Conjunction and disjunction
  - $(\forall x P) \wedge Q$ becomes $\forall x(P \wedge Q)$.
  - $(\forall x P) \vee Q$ becomes $\forall x(P \vee Q)$.
  - $(\exists x P) \wedge Q$ becomes $\exists x(P \wedge Q)$.
  - $(\exists x P) \vee Q$ becomes $\exists x(P \vee Q)$.
- Negation
  - $\neg(\exists x P)$ becomes $\forall x(\neg P)$.
  - $\neg(\forall x P)$ becomes $\exists x(\neg P)$.

- Implication
  - $(\forall x P) \Rightarrow Q$ becomes $\exists x(P \Rightarrow Q)$.
  - $(\exists x P) \Rightarrow Q$ becomes $\forall x(P \Rightarrow Q)$.
  - $P \Rightarrow (\forall x Q)$ becomes $\forall x(P \Rightarrow Q)$.
  - $P \Rightarrow (\exists x Q)$ becomes $\exists x(P \Rightarrow Q)$.

> ☞ An important point to remember is that you can accidentally capture a free variable with your quantifier, which will give you an incorrect formula. Here is an example of this: $(\forall x R(x, y)) \Rightarrow R(x, y)$. If we apply the implication rule directly, this will become $\forall x(R(x, y) \Rightarrow R(x, y))$. **This is an error.** You have to rename the quantifier variable and apply the rule only after renaming. We first rename the bounded variable, so the formula becomes $(\forall z R(z, y)) \Rightarrow R(x, y)$. Now we can apply the rule, so we have $\forall z(R(z, y) \Rightarrow R(x, y))$. This is equivalent to what we had in the beginning.

If we wanted to implement a converter to prenex normal form, we would have to make sure that our implementation does not accidentally capture free variables. An easy solution would be to generate fresh variables and rename all bounded variables with fresh variable names. In that case, we would eliminate the possibility of accidental capture. Another solution would be to check if the free variables and bounded variables have the same name, and rename that bounded variable if there is one.

Example 4. *Convert the following formula to prenex normal form:* $\forall x R(x) \Rightarrow \exists x R(x)$.

Solution:

$$\begin{aligned}
&\forall x R(x) \Rightarrow \exists x R(x) \\
&= \exists x(R(x) \Rightarrow \exists x R(x)) \quad \text{(by implication for } \forall) \\
&= \exists x(R(x) \Rightarrow \exists y R(y)) \quad \text{(renaming the bounded variable)} \\
&= \exists x \exists y(R(x) \Rightarrow R(y)) \quad \text{(by implication for } \exists)
\end{aligned}$$

Example 5. *Convert the following formula to prenex normal form:* $\neg((\exists x P(x)) \vee (\exists x Q(x)))$

Solution:

$$\begin{aligned}
&\neg((\exists x P(x)) \vee (\exists x Q(x))) \\
&= \neg(\exists x(P(x) \vee (\exists x Q(x)))) \quad \text{(by disjunction for } \exists) \\
&= \neg(\exists x(P(x) \vee (\exists y Q(y)))) \quad \text{(renaming the bounded variable)} \\
&= \neg(\exists x \exists y(P(x) \vee Q(y))) \quad \text{(by disjunction for } \exists) \\
&= \forall x \neg(\exists y(P(x) \vee Q(y))) \quad \text{(by negation for } \exists) \\
&= \forall x \forall y(\neg(P(x) \vee Q(y))) \quad \text{(by negation for } \exists)
\end{aligned}$$

3.2 Skolemization

Skolemization is the process of removing the existential quantifiers ($\exists$s) from our formulae. It proceeds as the following: $\forall x_1 \forall x_2 ... \forall x_n \exists y R(x, y)$ becomes $\forall x_1 \forall x_2 ... \forall x_n R(x, f(x_1, x_2, ..., x_n))$. The intuition behind it is that if you had different values for the universal quantifiers $x_1, x_2, ..., x_n$, the y you would pick would change. Hence the value y bounded by the existential quantifier in fact depends on $x_1, x_2, ..., x_n$. We can represent this with a Skolem function, that picks the y for given input $x_1, x_2, ..., x_n$. If there is no universal quantifier that the existential quantifier depends on, then the Skolem function will have the arity 0, which makes it a Skolem constant. e.g. $\exists y \forall x R(x, y)$ becomes, $\forall x R(x, c)$, where c is the Skolem constant.

> ☞ Every time you skolemize an existential quantifier, you have to use a fresh Skolem constant/function.

Let's do an example: $\forall x \exists y \forall z R(x, y, z)$ becomes $\forall x \forall z R(x, f(x), z)$. Observe that the Skolem function only includes the universal quantifier variables than come *before* our existential quantifier. $\forall x$ comes before $\exists y$, so it is included in $f(x)$, but $\forall z$ comes after $\exists y$, so it should not be included in $f(x)$, because y does not depend on z.

3.3 First-order resolution

Before we discuss the resolution rule for first-order logic, let's see what we have in the big picture. In propositional logic, we could directly convert our proposition to CNF and then resolve. However, the quantifiers are an obstacle when we try to do the same to first-order formulae. Therefore, we convert our formula to prenex normal form and thereby obtain a formula where all the quantifiers are in the beginning. We will now skolemize our formula and get rid of the existential quantifiers. Then we can make the universal quantifiers implicit, in other words, we will remove the quantifiers by making them metavariables. Now we have a quantifier-free formula! There is no obstacle to convert it to CNF and get a formula in our set notation.

However, when we want to prove it, our resolution rule from propositional logic is not enough to prove first-order logic formula. Let's update our rule:

Let C_1, C_2 be clauses (sets of disjunctions of literals) such that $P \in C_1$ and $Q \in C_2$. The resolution rule states that, if the substitution list θ is a most general unifier for P and $\neg Q$, then we can deduce $(C_1 \cup C_2) \setminus \{P, Q\}$ from this. In a more formal notation:

$$\frac{C_1 \qquad C_2}{((C_1 \cup C_2) \setminus \{P, Q\})\theta} \text{ where } P \in C_1, Q \in C_2, \theta$$

Example 6. *Prove the following sequent with resolution:* $\exists x(P(x) \vee Q(x)) \vdash (\exists x P(x)) \vee (\exists x Q(x))$.

Solution: Let's look at the left hand side. It is already in prenex normal form. Let's skolemize it. It should become $P(c) \vee Q(c)$, where c is a Skolem constant. With the set notation, we now have $\{\{P(c), Q(c)\}\}$.

Let's look at the right hand side. We are supposed to negate it and then convert it to prenex normal form. The negated version is $\neg((\exists x P(x)) \vee (\exists x Q(x)))$. We converted this to prenex normal form as an exercise before. So now we have $\forall x \forall y(\neg(P(x) \vee Q(y)))$. We make the universal quantifiers implicit (meta-variables), so what we have is $(\neg(P(X) \vee Q(Y)))$. We now need to convert this to CNF. By the application of De Morgan's law, we can easily get $(\neg P(X)) \wedge (\neg Q(Y))$. Now we have a CNF formula which corresponds to

$\{\{\neg P(X)\},\{\neg Q(Y)\}\}$ in set notation. Let's apply the resolution rule now.

$$\begin{aligned}&\{\{P(c),Q(c)\},\{\neg P(X)\},\{\neg Q(Y)\}\}\\&\{\{Q(c)\}\{\neg Q(Y)\}\}\quad\text{(by the substitution [c/X])}\\&\{\square\}\quad\text{(by the substitution [c/Y])}\end{aligned}$$

Example 7. *Prove the Drinker paradox with resolution:* $\vdash \exists y(D(y)\Rightarrow(\forall x D(x)))$.

Solution: Let's convert the negation of the right hand side to prenex normal form.

$$\begin{aligned}&\neg(\exists y(D(y)\Rightarrow(\forall x D(x))))\\&\neg(\exists y\forall x(D(y)\Rightarrow D(x)))\quad\text{(by implication for }\forall)\\&\forall y(\neg(\forall x(D(y)\Rightarrow D(x))))\quad\text{(by negation for }\exists)\\&\forall y\exists x(\neg(D(y)\Rightarrow D(x))))\quad\text{(by negation for }\forall)\end{aligned}$$

If we replace y with a meta-variable and x with a Skolem function, we will obtain $\neg(D(Y)\Rightarrow D(f(Y)))$. Let's convert this to CNF.

$$\begin{aligned}&\neg(D(Y)\Rightarrow D(f(Y)))\\&\neg((\neg D(Y))\vee D(f(Y)))\quad\text{(by implication)}\\&D(Y)\wedge(\neg D(f(Y)))\quad\text{(by De Morgan's 1 }\Rightarrow)\end{aligned}$$

In set representation, we now have $\{\{D(Y)\},\{\neg D(f(Y))\}\}$. Let's try to resolve this. Note that $D(Y)$ and $D(f(Y))$ are not unifiable. **However, we are allowed to rename any free variable in any clause we want. (given that you rename all meta-variables with the same name in that clause)** This will allow us to prove the formula.

$$\begin{aligned}&\{\{D(Y)\},\{\neg D(f(Y))\}\}\\&\{\{D(Z)\},\{\neg D(f(Y))\}\}\quad\text{(by changing }Y\text{ to }Z\text{ in the first clause)}\\&\{\square\}\quad\text{(by the substitution [f(Y)/Z])}\end{aligned}$$

Example 8. *Prove the following sequent using resolution. (Asymmetric relations are also antisymmetric.) (Hint:* $<$ *and* $=$ *are just relations here. You can replace* $x<y$ *with* $R(x,y)$ *and it would be the same thing. However,* $<$ *and* $=$ *are different relations.)*

$$\forall x\forall y((x<y)\Rightarrow\neg(y<x))\vdash\forall x\forall y\big(((x<y)\wedge(y<x))\Rightarrow x=y\big)$$

Solution: Let's look at the left hand side first. It's already in prenex normal form. $\forall x\forall y((x<y)\Rightarrow\neg(y<x))$ becomes $((X<Y)\Rightarrow\neg(Y<X))$ by removing the universal quantifiers and replacing their variables with meta-variables. Now observe that

$$\begin{aligned}&(X<Y)\Rightarrow\neg(Y<X)\\&=(\neg(X<Y))\vee(\neg(Y<X))\quad\text{(by implication)}\end{aligned}$$

We have a formula in CNF now. In set notation this would become: $\{\{\neg(X < Y), \neg(Y < X)\}\}$. Now let's look at the right hand side. We have a single formula there. Let's negate it.

$$\begin{aligned}
&\neg\Big(\forall x \forall y\big(((x < y) \wedge (y < x)) \Rightarrow (x = y)\big)\Big) \\
&= \neg\Big(\forall x \forall y\big(\neg((x < y) \wedge (y < x)) \vee (x = y)\big)\Big) \quad \text{(by implication)} \\
&= \exists x \neg\Big(\forall y\big(\neg((x < y) \wedge (y < x)) \vee (x = y)\big)\Big) \quad \text{(by negation for } \forall\text{)} \\
&= \exists x \exists y \neg\big(\neg((x < y) \wedge (y < x)) \vee (x = y)\big) \quad \text{(by negation for } \forall\text{)} \\
&= \exists x \exists y \neg\big(\neg((x < y) \wedge (y < x)) \vee (x = y)\big) \quad \text{(by negation for } \forall\text{)}
\end{aligned}$$

Now we have something in prenex normal form. If we replace the existential quantifiers with Skolem constants, we will get $\neg\big(\neg((c < d) \wedge (d < c)) \vee (c = d)\big)$. Let's convert this to CNF.

$$\begin{aligned}
&\neg\big(\neg((c < d) \wedge (d < c)) \vee (c = d)\big) \\
&= \big(((c < d) \wedge (d < c)) \wedge (\neg(c = d))\big) \quad \text{(by De Morgan's law 1} \Rightarrow\text{)}
\end{aligned}$$

Now we have a formula in CNF. In set notation: $\{\{c < d\}, \{d < c\}, \{\neg(c = d)\}\}$. So now for the resolution, let's get the union of the set notations we have for left and right hand sides and apply resolution to it.

$$\begin{aligned}
&\{\{\neg(X < Y), \neg(Y < X)\}, \{c < d\}, \{d < c\}, \{\neg(c = d)\}\} \\
&\{\{\neg(d < c)\}, \{d < c\}, \{\neg(c = d)\}\} \quad \text{(by resolving the first two clauses with [c/X, d/Y])} \\
&\{\square, \{\neg(c = d)\}\} \quad \text{(by resolving the first two clauses)}
\end{aligned}$$

Since we reached $\square$ (which is always false), we proved our sequent.

4 Superposition Calculus

Our objective is to extend resolution with the equality concept we introduced in previous classes.

4.1 New notation

We now consider $t \approx u$ a formula, and a literal. Its negation $t \not\approx u$ is also considered a literal. We assume $t \approx u$ and $u \approx t$ are the same atom.

We write $E[s]$ to mean an expression E with a particular occurrence of a term s. When we use the notation $E[s]$ and then write $E[t]$, the latter means the expression obtained from $E[s]$ by replacing the distinguished occurrence of s by the term t.

4.2 Naïve approach

We can convert the axioms into clauses, with the process we did in resolution (through CNF, prenex normal form etc.):

- Reflexivity: $\{t \approx t\}$
- Transitivity: $\{t \not\approx u, u \not\approx v, t \approx v\}$

- Congruence for all terms: $\{t_1 \not\approx u_1, ..., t_n \not\approx u_n, f(t_1, ..., t_n) \approx f(u_1, ..., u_n)\}$
- Congruence for all formulae: $\{t_1 \not\approx u_1, ..., t_n \not\approx u_n, \neg P(t_1, ..., t_n), P(u_1, ..., u_n)\}$

While this works in theory, this way is too verbose for practical use. There are also other problems caused by our axioms resolving each other. Therefore we will incorporate our equality notion to our resolution rule and get a new rule.

4.3 Simplification (reduction) ordering

During the inference process, applying our rule involves numerous rewrites, and this number increases quite fast. However if we suggest a restriction on how the rewrites will be done, this number can be reduced and we can make the inference more efficient. We will try to rewrite big terms to smaller terms and we will suggest an ordering on terms to achieve this.

We want to introduce a total ordering between ground terms, namely $\succ$. We want this ordering to satisfy certain properties:

1. Well-foundedness. (There is no infinite sequence of terms $t_1, t_2, ...$ such that $t_1 \succ t_2 \succ ....$)
2. Monotonic. (If $l \succ r$, then $s[l] \succ s[r]$.)
3. Stable under substitutions. (If $l \succ r$, then $l\theta \succ r\theta$.)
4. Subterm property. (If r is a strict subterm of l, then $l \succ r$.)

4.4 Ground superposition

In order to deal with equality directly, we will add new rules to our system, but we will keep using the resolution rule. To understand the intuition of these rules, let's first consider the system where the only predicate is equality.

4.4.1 Positive superposition rule

$$\frac{\{t \approx t'\} \cup C_1 \qquad \{s[t] \approx s'\} \cup C_2}{\{s[t'] \approx s'\} \cup C_1 \cup C_2} \text{ where } t \succ t',\ s[t] \succ s'$$

4.4.2 Negative superposition rule

$$\frac{\{t \approx t'\} \cup C_1 \qquad \{s[t] \not\approx s'\} \cup C_2}{\{s[t'] \not\approx s'\} \cup C_1 \cup C_2} \text{ where } t \succ t' \text{ and } s[t] \succ s'$$

4.4.3 Equality reflexivity rule

$$\frac{\{s \not\approx s\} \cup C}{C}$$

Since we know reflexivity always holds for equality, $s \not\approx s$ is not possible.

4.4.4 Equality factoring rule

$$\frac{\{t_1 \approx t_2, t_1 \approx t_3\} \cup C}{\{t_1 \approx t_2, t_2 \not\approx t_3\} \cup C} \text{ where } t_1 \succ t_2 \text{ and } t_2 \succeq t_3$$

This rule seems less intuitive than the others, so a verbal explanation would be helpful. We know that $t_1 \approx t_2$ or $t_1 \approx t_3$ holds. We want to show that $t_1 \approx t_2$ or $t_2 \not\approx t_3$ holds.

First case: If $t_1 \approx t_2$, then we already know what we were hoping to show.

Second case: If $t_1 \approx t_3$, then suppose $t_1 \not\approx t_2$ for now, because if they were $\approx$ then we would finish our proof in the first case. Since $t_1 \approx t_3$, we can replace t_1 with t_3 in $t_1 \not\approx t_2$. Therefore $t_2 \not\approx t_3$.

Example 9. *Prove the following sequent using ground superposition.*

$$\vdash \forall x \forall y (x \approx y \Rightarrow y \approx x)$$

Proof. Let's negate the right hand side and convert it to prenex normal form, skolemize it and convert it to CNF.

$$\neg(\forall x \forall y (x \approx y \Rightarrow y \approx x))$$
$$\exists x \exists y \neg (x \approx y \Rightarrow y \approx x))$$
$$\neg(a \approx b \Rightarrow b \approx a)$$
$$\neg(a \not\approx b \vee b \approx a)$$
$$a \approx b \wedge b \not\approx a$$

Now we have the formula $\{\{a \approx b\}, \{b \not\approx a\}\}$ to apply the rules on. Since $b \approx a$ and $a \approx b$ are the same atom, we are looking at $\{\{b \approx a\}, \{b \not\approx a\}\}$. Using the resolution rule we get $\square$. □

Example 10. *Prove the following sequent using ground superposition.*

$$\vdash \forall x \forall y \forall z ((x \approx y \wedge y \approx z) \Rightarrow x \approx z)$$

Proof. Let's negate the right hand side and convert it to prenex normal form, skolemize it and convert it to CNF.

$$\neg(\forall x \forall y \forall z ((x \approx y \wedge y \approx z) \Rightarrow x \approx z)$$
$$\exists x \exists y \exists z \neg ((x \approx y \wedge y \approx z) \Rightarrow x \approx z)$$
$$\neg((a \approx b \wedge b \approx c) \Rightarrow a \approx c)$$
$$\neg(\neg(a \approx b \wedge b \approx c) \vee a \approx c)$$
$$((a \approx b \wedge b \approx c) \wedge a \not\approx c)$$

Let's move this to set notation and complete the proof. Suppose the order is $c \succ b \succ a$.

$\{\{b \approx a\}, \{c \approx b\}, \{c \not\approx a\}\}$

$\{\{b \approx a\}, \{b \not\approx a\}\}$ (by negative sp of the 2nd and 3rd clauses, since $c \succ a$ and $b \succ a$)

$\{\square\}$ (by resolution)

$\square$

Of course, what we can do with ground superposition is limited, so we will later look at non-ground superposition.

Example 11. *Prove the following sequent using ground superposition.*

$$\vdash \forall x \forall y \forall z((x \approx f(y) \land y \approx g(z)) \Rightarrow x \approx f(g(z))$$

Proof. After conversion to prenex normal form, CNF and skolemization, in set notation, this would be: $\{\{a \approx f(b)\}, \{b \approx g(c)\}, \{a \not\approx f(g(c))\}\}$. Suppose the order is $f(c) \succ f(b) \succ f(a) \succ c \succ b \succ a$. Then we complete the proof with:

$\{\{f(b) \approx a\}, \{g(c) \approx b\}, \{f(g(c)) \not\approx a\}\}$

$\{\{f(b) \approx a\}, \{f(b) \not\approx a\}\}$ (by negative sp of the 2nd and 3rd clauses, since $g(c) \succ b$ and $f(g(c)) \succ a$)

$\{\square\}$ (by resolution)

$\square$

4.5 Non-ground superposition

Now $\approx$ is not the only predicate we have, therefore once again we need substitutions. Don't forget that we still have our resolution rule for first-order logic.

4.5.1 Positive superposition rule

$$\frac{\{t \approx t'\} \cup C_1 \qquad \{s[u] \approx s'\} \cup C_2}{(\{s[t'] \approx s'\} \cup C_1 \cup C_2)\theta} \text{ where } t \succ t',\ s[u] \succ s',\ \theta = mgu(t,u),\ u \text{ is not a variable}$$

4.5.2 Negative superposition rule

$$\frac{\{t \approx t'\} \cup C_1 \qquad \{s[u] \not\approx s'\} \cup C_2}{(\{s[t'] \not\approx s'\} \cup C_1 \cup C_2)\theta} \text{ where } t \succ t' \text{ and } s[u] \succ s',\ \theta = mgu(t,u),\ u \text{ is not a variable}$$

4.5.3 Equality reflexivity rule

$$\frac{\{s \not\approx s'\} \cup C}{C\theta} \text{ where } \theta = mgu(s,s')$$

4.5.4 Equality factoring rule

$$\frac{\{t_1 \approx t_2, t_1' \approx t_3\} \cup C}{\{t_1 \approx t_2, t_2 \not\approx t_3\} \cup C} \text{ where } t_1 \succ t_2 \text{ and } t_2 \succeq t_3,\ \theta = mgu(t_1, t_1')$$

4.5.5 Rule for superposition with other predicates

So far we only have rules for equalities to deal with each other, and the resolution rule to deal with all predicates (including $\approx$). However, our resolution rule does not make use of the ordering we defined. We can add a new rule to make use of this ordering when we are dealing with an equality and a predicate different than $\approx$. This rule is not necessary for our system, but it gives our proof a direction in terms of what to replace in the predicates, therefore our automated provers can complete the proof faster. In fact, the famous prover Vampire uses this rule. (Kovacs, Voronkov)

$$\frac{\{t \approx t'\} \cup C_1 \qquad L[u] \cup C_2}{(L[t'] \cup C_1 \cup C_2)\theta} \text{ where } t \succ t',\ \theta = mgu(t,u),\ u \text{ is not a variable}$$

Example 12. *Prove the following sequent using non-ground superposition. (It basically says, knowing the addition operation, prove that $1+2=3$.)*

$$\forall x\Big(add(o,x) \approx x\Big), \forall x,y,z\Big(add(x,y) \approx z \Rightarrow add(s(x),y) \approx s(z)\Big) \vdash add(s(o),s(s(o))) \approx s(s(s(o)))$$

Example 13. *Prove the following sequent using non-ground superposition.*

$$\forall x\Big(o \leq x\Big), \forall y,z\Big(y \leq z \Rightarrow s(y) \leq s(z)\Big), a \approx s(o), b \approx s(s(o)) \vdash a \leq b$$