# CS228 Logic for Computer Science 2022

Lecture 8: Completeness

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2022-01-21



# Topic 8.1

# Completeness



### Completeness

Now let us ask the daunting question!!!!!

Is resolution proof system complete?

In other words,

if  $\Sigma$  is unsatisfiable, are we guaranteed to derive  $\Sigma\vdash \bot$  via resolution?

We need a notion of not able to derive something.

# Clauses derivable with proofs of depth n

We define the set  $Res^n(\Sigma)$  of clauses that are derivable via resolution proofs of at most depth *n* from the set of clauses  $\Sigma$ .

# Definition 8.1

Let  $\Sigma$  be a set of clauses.

$$Res^{0}(\Sigma) \triangleq \Sigma$$
$$Res^{n+1}(\Sigma) \triangleq Res^{n}(\Sigma) \cup \{C | C \text{ is a resolvent of clauses } C_{1}, C_{2} \in Res^{n}(\Sigma)\}$$

### Example 8.1

$$Let \Sigma = \{(p \lor q), (\neg p \lor q), (\neg q \lor r), \neg r\}.$$
  

$$Res^{0}(\Sigma) = \Sigma$$
  

$$Res^{1}(\Sigma) = \Sigma \cup \{q, p \lor r, \neg p \lor r, \neg q\}$$
  

$$Res^{2}(\Sigma) = Res^{1}(\Sigma) \cup \{r, q \lor r, p, \neg p, \bot\}$$



# All derivable clauses

 $Res^n(\Sigma)$  may saturate at some time point.

Definition 8.2 Let  $\Sigma$  be a set of clauses. There may be some m such that

$$\mathit{Res}^{m+1}(\Sigma) = \mathit{Res}^m(\Sigma).$$

Let  $Res^*(\Sigma) \triangleq Res^m(\Sigma)$ .

If  $\Sigma$  is finite then *m* certainly exists.



# Completeness

### Theorem 8.1

If a finite set of clauses  $\Sigma$  is unsatisfiable,  $\bot \in Res^*(\Sigma)$ .

### Proof.

We prove the theorem using induction over number of variables in  $\Sigma.$  Wlog, We assume that there are no tautology clauses in  $\Sigma_{.(why?)}$ 

#### base case:

p is the only variable in  $\Sigma$ . Assume  $\Sigma$  is unsat. Therefore,  $\{p, \neg p\} \subseteq \Sigma$ .

We have the following derivation of  $\perp$ .

$$\frac{\Sigma \vdash p \qquad \Sigma \vdash \neg p}{\mid}$$

# Completeness (contd.)

# Proof(contd.)

#### induction step:

Assume: theorem holds for all the formulas containing variables  $p_1, ..., p_n$ . Consider an unsatisfiable set  $\Sigma$  of clauses containing variables  $p_1, ..., p_n, p$ . Let

►  $\Sigma_0 \triangleq$  the set of clauses from  $\Sigma$  that have *p*.

▶  $\Sigma_1 \triangleq$  be the set of clauses from  $\Sigma$  that have  $\neg p$ .

▶  $Σ_* \triangleq$  be the set of clauses from Σ that have neither *p* nor ¬p. Furthermore, let

• 
$$\Sigma'_0 \triangleq \{C - \{p\} | C \in \Sigma_0\}$$
  
•  $\Sigma'_1 \triangleq \{C - \{\neg p\} | C \in \Sigma_1\}$ 

$$\Sigma = \Sigma_0 \wedge \Sigma_1 \wedge \Sigma_*$$

#### Exercise 8.1

Show 
$$\Sigma'_0 \models \Sigma_0$$
 and  $\Sigma'_1 \models \Sigma_1$   
©0.000 CS228 Logic for Computer Science 2022

# Example: projections

Example 8.2 Consider  $\Sigma = \{p_1 \lor p, p_2, \neg p_1 \lor \neg p_2 \lor p, \neg p_2 \lor \neg p\}$ 

$$\begin{split} \Sigma_0 &= \{ p_1 \lor p, \neg p_1 \lor \neg p_2 \lor p \} \\ \Sigma_1 &= \{ \neg p_2 \lor \neg p \} \\ \Sigma_* &= \{ p_2 \} \end{split}$$

$$\begin{split} \boldsymbol{\Sigma}_0' &= \{\boldsymbol{p}_1, \neg \boldsymbol{p}_1 \lor \neg \boldsymbol{p}_2\} \\ \boldsymbol{\Sigma}_1' &= \{\neg \boldsymbol{p}_2\} \end{split}$$

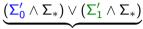
Let us get familiar with an important formula:  $(\Sigma'_0 \wedge \Sigma_*) \lor (\Sigma'_1 \wedge \Sigma_*) = \{p_1, \neg p_1 \lor \neg p_2, p_2\} \lor \{\neg p_2, p_2\}$ 



# Completeness (contd.)

Proof(contd.)

Now consider formula



p is not in the formula

claim: If  $(\Sigma'_0 \wedge \Sigma_*) \vee (\Sigma'_1 \wedge \Sigma_*)$  is sat then  $\Sigma$  is sat.

- Assume for some  $m, m \models (\Sigma'_0 \land \Sigma_*) \lor (\Sigma'_1 \land \Sigma_*).$
- Therefore,  $m \models \Sigma_{*}$ .(why?)
- ► Case 1:  $m \models (\Sigma'_1 \land \Sigma_*)$ . Since all the clauses of  $\Sigma_0$  have p,  $m[p \mapsto 1] \models \Sigma_{0(why?)}$ . Since  $\Sigma'_1$  and  $\Sigma_*$  have no p,  $m[p \mapsto 1] \models \Sigma'_1$  and  $m[p \mapsto 1] \models \Sigma_*$ . Since  $\Sigma'_1 \models \Sigma_1$ ,  $m[p \mapsto 1] \models \Sigma_1$ .

• Case 2:  $m \models (\Sigma'_0 \land \Sigma_*)$ . Symmetrically,  $m[p \mapsto 0] \models \Sigma_0 \land \Sigma_1 \land \Sigma_*$ .

• Therefore,  $\Sigma_0 \wedge \Sigma_1 \wedge \Sigma_*$  is sat.

Exercise 8.2 Show  $\Sigma$  and  $(\Sigma'_0 \wedge \Sigma_*) \vee (\Sigma'_1 \wedge \Sigma_*)$  are equivalent but not equivalent.

# Completeness (contd.)

$$\begin{split} & \text{Proof(contd.)} \\ & \text{Since } \Sigma \text{ is unsat, } (\Sigma_0' \wedge \Sigma_*) \vee (\Sigma_1' \wedge \Sigma_*) \text{ is unsat.} \end{split}$$

Now we apply the induction hypothesis. Since  $(\Sigma'_0 \wedge \Sigma_*) \vee (\Sigma'_1 \wedge \Sigma_*)$  is unsat and has no  $p, \perp \in Res^*(\Sigma'_0 \wedge \Sigma_*)$  and  $\perp \in Res^*(\Sigma'_1 \wedge \Sigma_*)$ .

Choose a derivation of  $\bot$  from both. Now there are two cases.

Case 1:  $\bot$  was derived using only clauses from  $\Sigma_*$  in any of the two proofs. Therefore,  $\bot \in Res^*(\Sigma_*)$ . Therefore,  $\bot \in Res^*(\Sigma_0 \land \Sigma_1 \land \Sigma_*)$ .

Case 2: In both the derivations  $\Sigma'_0$  are  $\Sigma'_1$  are involved respectively.



# Example: choosing derivations

#### Example 8.3

Recall our example  $\Sigma_* = \{p_2\}, \Sigma'_0 = \{p_1, \neg p_1 \lor \neg p_2\}, \Sigma'_1 = \{\neg p_2\}.$ 

Proofs for our running example



The above proofs belong to the case 2.

The above proofs do not start from clauses that are from  $\Sigma$ . So we cannot use them immediately. We need a construction.



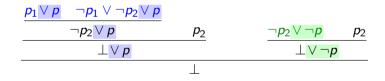
# Completeness (contd.)

# Proof(contd.)

Case 2: In both the derivations  $\Sigma'_0$  are  $\Sigma'_1$  are involved respectively.(contd.) Therefore,  $p \in Res^*(\Sigma_0 \land \Sigma_*)$  and  $\neg p \in Res^*(\Sigma_1 \land \Sigma_*)$ .(why?)[needs thinking; look at the example to understand.] Therefore,  $\bot \in Res^*(\Sigma_0 \land \Sigma_1 \land \Sigma_*)$ (why?).

### Example 8.4

Recall proofs.



### Exercise 8.3

Let F be an unsatisfiable CNF formula with n variables. Show that there is a resolution proof of  $\perp$  from F of size that is smaller than or equal to  $2^{n+1} - 1$ .

**Commentary:** By inserting p in  $\Sigma'_0$  clauses of the left proof we obtain clauses of  $\Sigma_0$ . Therefore, the proof transforms into a proof from  $\Sigma_0 \wedge \Sigma_*$ . Since there are no  $\neg p$  anywhere in  $\Sigma_0 \wedge \Sigma_*$ , we are guaranteed a leftover p. We need a symmetric argument for deriving  $\neg p$  from  $\Sigma_1 \wedge \Sigma_*$ .

© <b>()</b> \$0	CS228 Logic for Cor	nputer Science 2	2022	Instructor: Ashutosh G	upta	IITB, India	12

# Completeness so far

Theorem 8.2

Let  $\Sigma$  be a finite set of formulas and F be a formula. The following statements are equivalent.

► 
$$\Sigma \vdash F$$
  
►  $\emptyset \in Res^*(\Sigma')$ , where  $\Sigma'$  is CNF of  $\bigwedge \Sigma \land \neg F$   
►  $\Sigma \models F$   
Proof.  
Proof can be generated  
 $\emptyset \in Res^*(\Sigma')$  ←  $\Sigma \models F$ 

Exercise 8.4

How is the last theorem applicable here?

ast theorem

13

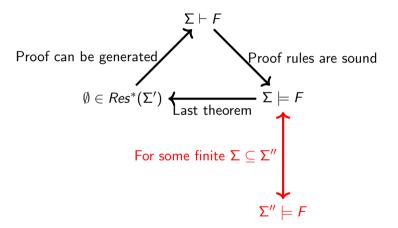
# Topic 8.2

# Finite to Infinite



How do we handle  $\Sigma'' \models F$  if  $\Sigma''$  is an infinite set?

There is an interesting argument.



We prove that if an infinite set implies a formula, then a finite subset also implies the formula.

# A theorem on strings

### Theorem 8.3

Consider an infinite set S of finite binary strings. There exists an infinite string w such that the following holds.

$$\forall n. |\{w' \in S | w_n \text{ is prefix of } w'\}| = \infty$$

where  $w_n$  is prefix of w of length n.

#### Proof.

We inductively construct w, and we will keep shrinking S. Initially  $w := \epsilon$ .

**Commentary:**  $\epsilon$  is the empty string.

#### base case:

w is prefix of all strings in S.



# A theorem on strings (contd.)

# Proof(contd.)

induction step:

Let us suppose we have w of length n and w is prefix of all strings in S.

- Let  $S_0 := \{u \in S | u \text{ has } 0 \text{ at } n + 1th \text{ position} \}.$
- Let  $S_1 := \{u \in S | u \text{ has } 1 \text{ at } n + 1th \text{ position} \}.$
- ▶ Let  $S_{\epsilon} := S \cap \{ w \}$ .

Clearly,  $S = S_{\epsilon} \cup S_0 \cup S_1$ . Either  $S_0$  or  $S_1$  is infinite.(why?) If  $S_0$  is infinite, w := w0 and  $S := S_0$ . Otherwise, w := w1 and  $S := S_1$ . w of length n + 1 is prefix of all strings in the shrunk S.

Therefore, we can construct the required w.

### Exercise 8.5

- a. Is the above construction of w practical?
- b. Construct infinite w for set S containing words of form  $0^*1$

@**()**\$0

# Compactness

Theorem 8.4

A set  $\Sigma$  of formulas is satisfiable iff every finite subset of  $\Sigma$  is satisfiable.

Proof. Forward direction is trivial.(why?)

Reverse direction:

We order formulas of  $\Sigma$  in some order, *i.e.*,  $\Sigma = \{F_1, F_2, \dots, \}$ .

Let  $\{p_1, p_2, ...\}$  be ordered list of variables from Vars $(\Sigma)$  such that

- variables in  $Vars(F_1)$  followed by
- the variables in  $Vars(F_2) Vars(F_1)$ , and so on.

Due to the rhs, we have models  $m_n$  such that  $m_n \models \bigwedge_{i=1}^n F_i$ .

We need to construct a model *m* such that  $m \models \Sigma$ . Let us do it!

# Compactness (contd.) II

### Proof(contd.)

We assume  $m_n$ : Vars $(\bigwedge_{i=1}^n F_i) \to \mathcal{B}$ .

**Commentary:** Notation alert: we assumed our models assign values to all variables. Here we are defining a different object that maps only finitely many variables.

We may see  $m_n$  as finite binary strings, since variables are ordered  $p_1, p_2, ...$  and  $m_n$  is assigning values to some first k variables.

Let  $S = \{m_n \text{ as a string } | n > 0\}$ 

Due to the previous theorem, there is an infinite binary string m such that each prefix of m is prefix of infinitely many strings in S.



Example : some  $m_n$  may not be a prefix of m

#### Example 8.5

Consider  $\Sigma = \{ p \lor q, \neg p \land r, \dots \}$ 

Let  $m_1 = \{p \mapsto 1, q \mapsto 0\}$ 

Let 
$$m_2 = \{p \mapsto 0, q \mapsto 1, r \mapsto 1\}$$

Note that  $m_1 \not\models \neg p \land r$ . Therefore,  $m_1$  will not be prefix of any  $m_n$  and consequently not prefix of m.

#### Exercise 8.6

Give an example of  $\Sigma$ ,  $m_n s$ , and m following the construction of previous slide such that no  $m_n$  is prefix of m?

# Compactness (contd.) III

# Proof(contd.)

**claim:** if we interpret *m* as a model<sub>(how?)</sub>, then  $m \models \Sigma$ .

- Consider a formula  $F_n \in \Sigma$ .
- Let k be the number of variables appearing in  $\bigwedge_{i=1}^{n} F_i$ .

CS228 Logic for Computer Science 2022

- Let m' be the prefix of length k of m.
- ▶ There must be  $m_j \in S$ , such that m' is prefix of  $m_j$  and  $j > n_{(why?)}$
- Since  $m_j \models \bigwedge_{i=1}^j F_i$ ,  $m_j \models F_n$ .
- ▶ Therefore,  $m' \models F_n$ .
- ▶ Therefore,  $m \models F_n$ .

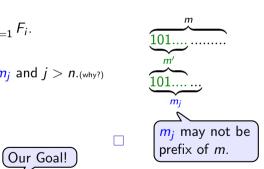
### Exercise 8.7

 $\Theta$ 

Using the above theorem prove that if  $\Sigma'' \models F$  then there is a finite  $\Sigma \subseteq \Sigma''$  such that  $\Sigma \models F$ .

**Commentary:** m' may not be  $m_n$  as in the example 8.5. The theorem is about showing that even if  $m_n$  is not there, there is some other model that satisfies  $F_n$ . Furthermore,  $m_j$  may also be not a prefix of m. Surprised! Georg Cantor lost his mind thinking about  $\infty$ . Lookout for BBC documentary Dangerous Knowledge.

Instructor: Ashutosh Gupta



IITB. India

21

Implication is decidable for finite lhs.

### Theorem 8.5

If  $\Sigma$  is a finite set of formulas, then  $\Sigma \models F$  is decidable.

### Proof.

Due to truth tables.



# Two definitions: effectively enumerable and semi-decidable

### Definition 8.3

If we can enumerate a set using an algorithm, then it is called effectively enumerable.

### Example 8.6

- The set of all programs is effectively enumerable, since they are finite strings that can be parsed.
- > The set of all terminating programs is not effectively enumerable.

### Definition 8.4

A yes/no problem is semi-decidable, if we have an algorithm for only one side of the problem.

Commentary: The above definitions will be formally covered in complexity theory and automata theory.							
ତ୍ତାର୍ତ୍ତ	CS228 Logic for Computer Science 2022	Instructor: Ashutosh Gupta	IITB, India	23			

# Implication is semi-decidable

### Theorem 8.6

If  $\Sigma$  is effectively enumerable, then  $\Sigma \models F$  is at least semi-decidable.

### Proof.

Due to compactness if  $\Sigma \models F$ , there is a finite set  $\Sigma_0 \subseteq \Sigma$  such that  $\Sigma_0 \models F$ . Since  $\Sigma$  is effectively enumerable, let  $G_1, G_2, ...$  be the enumeration of  $\Sigma$ . Let  $S_n \triangleq \{G_1, \ldots, G_n\}$ . There must be a  $S_k \supseteq \Sigma_{0 \text{(why?)}}$ . Therefore,  $S_k \models F$ . We may enumerate  $S_n$  and check  $S_n \models F$ , which is decidable. Therefore, eventually we will say yes if  $\Sigma \models F$ .

**Commentary:** If  $\Sigma \models F$  does not hold, the above procedure will not terminate. Therefore, implication is only semi-decidable and not decidable. However, the proof is not complete. It does not show that there is no other algorithm that can not decide  $\Sigma \models F$ .

# Topic 8.3

# Problems



# Slim proofs

For an unsatisfiable CNF formula F, a resolution proof R is a sequence of clauses such that:

- Each clause in R is either from F or derived by resolution from the earlier clauses in R.
- The last clause in R is  $\perp$ .

Consider the following definitions

► For a clause *C* and literal 
$$\ell$$
, let  $C|_{\ell} \triangleq \begin{cases} \top & \ell \in C \\ C - \{\overline{\ell}\} & \text{otherwise.} \end{cases}$ 

- ► Let  $F|_{\ell} \triangleq \bigwedge_{C \in F} C|_{\ell}$ .
- Let width(R) and width(F) be the length of the longest clause in R and F, respectively.
- ▶ Let  $slimest(F) \triangleq min(\{width(R) | R \text{ is resolution proof of unsatisfiability of } F\}).$

### Exercise 8.8

Prove the following facts.

1. if  $F|_{\ell}$  has an unsatisfiability proof, then  $F \wedge \ell$  has an unsatisfiability proof.

2. if  $k \ge width(F)$ ,  $slimest(F|_{\ell}) \le k-1$ , and  $slimest(F|_{\ell}) \le k$  then  $slimest(F) \le k$ .

# Exercise: connect finite and infinite

### Exercise 8.9

Consider an infinite set S of finite binary strings. Prove/disprove: For each infinite binary string w the following holds.

 $\forall n. |\{w' \in S | w_n \text{ is prefix of } w'\}| > 0$  iff  $\forall n. |\{w' \in S | w_n \text{ is prefix of } w'\}| = \infty$ 

where  $w_n$  is prefix of w of length n.



# End of Lecture 8

