Lecture 19 - More Natural Deduction

Vaishnavi Sundararajan

COL703/COL7203 - Logic for Computer Science

Recap: Natural deduction proof system

- Proof system that more closely mirrors human reasoning
- No axiom schema, all proof rules
- Each operator gets an introduction rule and/or an elimination rule
- Introduction rule: Operator appears in the conclusion
- Elimination rule: Operator appears in the (RHS of) premise(s), does not appear in the conclusion
- More amenable to automation; enjoys some nice properties

Recap: Proof rules for propositional fragment

Introduction rule	Elimination rule
$\frac{\Gamma \vdash \varphi_0 \qquad \Gamma \vdash \varphi_1}{} \land i$	$\frac{\Gamma \vdash \varphi_0 \land \varphi_1}{} \land e$
$\Gamma \vdash \varphi_0 \land \varphi_1$	$\Gamma \vdash \varphi_j$ $\land e_j$
$\Gamma \vdash \varphi_j$ $\forall i_i$	$\Gamma \vdash \varphi_0 \lor \varphi_1 \qquad \Gamma, \varphi_0 \vdash \psi \qquad \Gamma, \varphi_1 \vdash \psi$
$\Gamma \vdash \varphi_0 \lor \varphi_1$	Γ⊢ψ
$\Gamma, \varphi \vdash \psi$ $\supset i$	$ \begin{array}{ccc} \Gamma \vdash \varphi \supset \psi & \Gamma \vdash \varphi \\ \hline $
$\Gamma \vdash \varphi \supset \psi$	Γ⊢ψ
$\Gamma, \varphi \vdash \neg \psi \qquad \Gamma, \varphi \vdash \psi$	Γ ⊢ ¬¬φ
$\Gamma \vdash \neg \varphi$	$\Gamma \vdash \varphi$

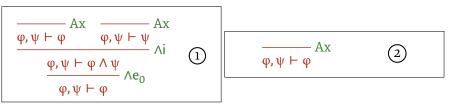
Recap: Proof rules for \exists and \forall

Introduction rule	Elimination rule
$\frac{\Gamma \vdash \varphi\{y/x\}}{\Gamma \vdash \forall x. \ [\varphi]} \ \forall i \ (y \ \text{fresh})$	$\frac{\Gamma \vdash \forall x. \ [\varphi]}{\Gamma \vdash \varphi\{t/x\}} \ \forall e$
$\frac{\Gamma \vdash \varphi\{t/x\}}{\Gamma \vdash \exists x. \ [\varphi]} \exists i$	$\frac{\Gamma \vdash \exists x. \ [\phi] \Gamma, \phi\{y/x\} \vdash \psi}{\Gamma \vdash \psi} \exists e \ (y \ fresh)$

where *t* is a term in the language, and $y \in \mathcal{V}$ is fresh if $y \notin \text{vars}(\Gamma \cup \{\varphi, \psi\})$.

We say that $\Gamma \vdash_{\mathscr{G}} \varphi$ if there is a proof of φ from assumptions Γ using Ax and the rules in both the above tables.

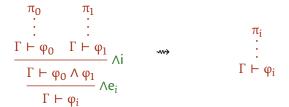
Unnecessary detours in proofs



- In \bigcirc 1, we first introduce an \land 1, and then immediately eliminate it.
- Could have replaced this entire proof by ②, without any such wasteful detours involving large expressions.
- Clearly both valid proofs of the same sequent.
- Prefer (2), since no large expression ($\phi \wedge \psi$ in this case) is introduced only to be immediately eliminated.
- What other useless detours are possible? Can we get rid of those also?

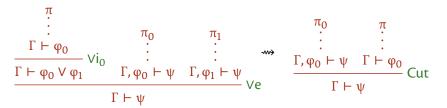
Removing unnecessary detours: A

Suppose $\Gamma \vdash \varphi_0$ via a proof π_0 and $\Gamma \vdash \varphi_1$ via π_1 .



Removing unnecessary detours: \lor

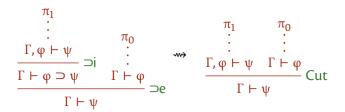
Suppose $\Gamma \vdash \varphi_0$ via π , Γ , $\varphi_0 \vdash \psi$ via π_0 , and Γ , $\varphi_1 \vdash \psi$ via π_1 .



Exercise: What about an application of $\forall i_i$ in the second or third premise? Is that a detour to be handled?

Unnecessary detours: ⊃

Suppose $\Gamma \vdash \varphi$ via a proof π_0 and $\Gamma, \varphi \vdash \psi$ via a proof π_1 .



Normal proofs

- We can eliminate the unnecessary detours for \land , \lor , and \supset .
- If we keep getting rid of these useless detours, eventually, we arrive at a **normal proof** with no detours.
- Every proof can be converted to a normal equivalent (How?)
- Is a smaller proof inherently better?
- How large can a proof of $\Gamma \vdash \varphi$ be?

Normal proofs

- We can eliminate the unnecessary detours for \land , \lor , and \supset .
- If we keep getting rid of these useless detours, eventually, we arrive at a **normal proof** with no detours.
- Every proof can be converted to a normal equivalent (How?)
- Is a smaller proof inherently better?
- How large can a proof of $\Gamma \vdash \varphi$ be?
- No ab initio bound, since we still need to instantiate each proof rule with expressions.
- Is there a bound on the size of any expression that can occur in any proof of $\Gamma \vdash \phi$?

Proof search: System without negation

- A normal proof will satisfy a subformula property
- Any expression occurring in any normal proof of $\Gamma \vdash \varphi$ is a subformula of φ , or of some expression in Γ .
- Need to consider subformulas of the conclusion only when the last rule is an introduction rule! Just subformulas of Γ for elimination rules.
- Consider the set S of subformulae of Γ and φ. S is perhaps large, depending on how big Γ is (but still finite)!
- No longer have to consider arbitrary expressions in any proof; gives us an algorithm for proof search!
- Algorithm is non-deterministic: Guess the last rule of a possible proof, and check if premises are derivable.

Proof search algorithm

- Want to determine if $\Gamma \vdash \varphi$. Let the last rule of a proof be **r**.
- Suppose φ is $\alpha \wedge \beta$, and we guess r to be \wedge i
- Then, check if $\Gamma \vdash \alpha$ and $\Gamma \vdash \beta$
- Both (recursive) calls need to succeed!
- What if φ is $\alpha \supset \beta$, and we guess r to be \supset i?
- Left hand side has to be enlarged!
- Recursive call to check if Γ , $\alpha \vdash \beta$

Proof search algorithm - continued

- Why all the song and dance about a subformula property?
- Suppose we guess \mathbf{r} to be $\Lambda \mathbf{e}_0$
- Then, we have to guess a ψ such that $\phi \wedge \psi \in S$, and the recursive call is to check if $\Gamma \vdash \phi \wedge \psi$
- Could be an enlarged LHS if r guessed to be ∨e
- If we "mark" formulas and contexts for which we have proofs, then
 only polynomially many recursive calls are made to check if Γ ⊢ φ
- One gets a PSPACE algorithm
- Theorem provers often use smart heuristics to improve this!

About negation

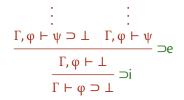
• Does this strategy lift to all of ⊢_𝒯?

About negation

- Does this strategy lift to all of ⊢_𝒯?
- What if I have to apply ¬e to get φ?
- Have to consider $\neg \neg \varphi$ one level up. Not a subformula!
- Perhaps still doable; add $\neg \neg \varphi$ to the set of "subformulae" of φ
- What about ¬i? Recall that we had to carefully think about which expression to derive in contradictory forms.
- What tells me which such expression is the correct one?
- Not much more than intuition, it would seem!
- Negation seems to complicate life, even in the propositional fragment

More about negation

- Write $\neg \varphi$ as $\varphi \supset \bot$
- Can capture ¬i as follows.



What about ¬e? No equivalent rule as such!

More about negation

- Write $\neg \varphi$ as $\varphi \supset \bot$
- Can capture $\neg i$ as follows.

$$\begin{array}{ccc} \vdots & \vdots \\ \frac{\Gamma, \varphi \vdash \psi \supset \bot & \Gamma, \varphi \vdash \psi}{\frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \varphi \supset \bot}} \supset e \end{array}$$

 What about ¬e? No equivalent rule as such! Can write the following rule to capture the effect of ¬e

$$\frac{\Gamma, \neg \varphi \vdash \bot}{\Gamma \vdash \varphi} \neg \mathsf{new}$$

- Moves an expression from left to right, and removes a negation
- Can still normalize and get **some** notion of a "subformula" property

Can we handle \neg better?

- ¬e a consequence of the law of excluded middle (LEM)
- LEM: $\varphi \lor \neg \varphi$ is valid for any expression φ
- What if we threw away LEM?
- Reject classical logic; move to intuitionistic logic
- Introduced by Brouwer in the first decade of the 20th century
- Basic idea: every proof needs to be constructive
- Informally: "An expression could be True, False, or unknown"
- Not allowed to get a proof of φ V ¬φ without proving φ or ¬φ

Intuitionistic logic: Propositional fragment

- Ax, and the rules for \land , \lor , and \supset as earlier; remove rules for \neg
- Use the ⊥ operator, and the following (elimination) rule

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi} \bot e$$

¬i can be captured using ⊥ and ⊃i as follows

$$\frac{\Gamma, \varphi \vdash \psi \supset \bot \quad \Gamma, \varphi \vdash \psi}{\frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \varphi \supset \bot}} \supset e$$

• Subformula property: \perp is a subformula of any φ ; still a finite set!

And one more thing...

- What about normalization though?
- Do earlier rewrites suffice? Do we need to handle detours due to ⊥?

And one more thing...

- What about normalization though?
- Do earlier rewrites suffice? Do we need to handle detours due to ⊥?
- What about a proof of the following shape?

$$\begin{array}{c} \vdots \\ \frac{\Gamma \vdash \bot}{\Gamma \vdash \alpha \land \beta} \bot e \\ \hline \Gamma \vdash \alpha & \land e_0 \end{array}$$

- Could have got α directly from \perp ; unnecessarily introduced $\alpha \wedge \beta$
- **New normalization rule**: No rule follows an application of ⊥e
- Any normal proof enjoys the subformula property involving ⊥
- Clean proof search (that also handles negation-without-LEM)

What about FO now?

Are there unnecessary detours for \forall and \exists as well? Suppose $\Gamma \vdash \varphi(y)$ for some fresh $y \notin \text{vars}(\Gamma)$ via a proof π .

$$\frac{\Gamma \vdash \varphi(y)}{\Gamma \vdash \forall x. \ [\varphi(x)]} \forall i$$

$$\frac{\Gamma \vdash \varphi(t)}{\Gamma \vdash \varphi(t)} \forall e$$

$$\frac{\pi'}{\Gamma}$$

$$\vdots$$

$$\Gamma \vdash \varphi(t)$$

Here, π' is the proof π where every occurrence of y has been replaced by t. Γ is unaffected since y is fresh.

What about FO now?

Here, π_2' is the proof π_2 where every occurrence of z has been replaced by t. The proof is unaffected since $z \notin vars(\Gamma \cup \{\psi\})$, so replacing it by t (which might or might not appear in Γ or ψ) makes no difference to the overall structure of the proof.

What about FO now?

- Subformula property has to be modified
- Every $\varphi(t)$ a subformula of $\exists x$. $[\varphi(x)]$ (introduction rule)
- Every $\varphi(y)$ a subformula of $\forall x$. $[\varphi(x)]$
- Can remove detours; but the set of subformulae is now infinite!
- Unfortunately, no getting around this in the general case
- Proof search is not decidable
- But depending on the application, one might be able to restrict the shapes of these rules to get decidability
- A security application, for example, might only existentially quantify terms that a principal can generate not arbitrary ones.