Computational Proof Theory: Transforming Proofs using Automated Reasoning

David M. Cerna

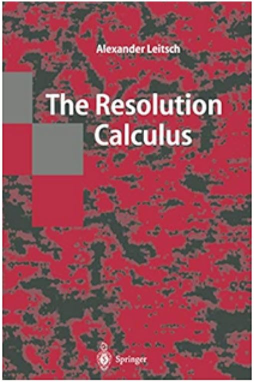
EuroProofNet Summer School on AI for Reasoning and Processing of Mathematics

June 25^{th} 2024

Goals

- Introduce automated reasoning and the resolution calculus
- Completeness of the resolution calculus
- the Sequent calculus
- Cut-elimination
- Eliminating cuts using resolution

Automated Theorem Proving



- Material based on "The Resolution Calculus" by Alexander Leitsch.

Automated Theorem Proving

- In the most basic sense, automated provers provide a proof that a statement follows from a particular theory T.
- For classical propositional logic, such a prover can decide if the statement follows from T.
- For classical first-order logic (FOL), a prover can only be guaranteed to find a proof if the statement follows from T.
- One can imagine an automated prover which exhaustively applies the rules of a particular complete calculus for T until a proof is constructed.
- For FOL there seem to be too many degrees of freedom, i.e. quantifier instantiations .

The Resolution Calculus

- Of the variety of approaches to automated reasoning:
  - the tableau calculus,
  - the connection method,
- we will focus on the the resolution calculus.
- It is the most commonly used method for theorem proving.
- In its most basic form it consist of a single rule:

$$\frac{\Delta \vee C \qquad \Delta' \vee \neg D}{\Delta\sigma \ \vee \ \Delta'\sigma}\ \text{Res}$$

- where $C\sigma \equiv D\sigma$
- $C\sigma, \neg C\sigma \notin \Delta\sigma$, and $D\sigma, \neg D\sigma \notin \Delta'\sigma$
- This is (almost) enough for a complete proof system for FOL.

Basic principle of Resolution

- $\Delta \vee C$ and $\Delta' \vee \neg D$ are disjunctions of literals.
- we will refer to them as clauses.
- C and D are literals which may be equated by an appropriate substitution of the free variables.
- σ is a unifier of the two literals.

$$f(x, y)\{x \mapsto y\} = f(y, x)\{x \mapsto y\}$$

- Essentially, if a set of clauses is unsatisfiable resolution can be used to provide a proof of unsatisfiability.
- Note: if a formula is valid, then its negation is unsatisfiable.
- Any FOL formula may be translated to a set of clauses.

Clausal Form

- Note: translating a FOL formula to a set of clauses can be done in a satisfiability preserving way.
  - Enough for our goal.
- We assume FOL formulas are constructed using the logic connectives $\{\exists, \forall, \wedge, \vee, \neg\}$.
- First step to translation to Negation normal form $nnf(F)$:
  - If $F = \neg Qx\ \varphi(x)$ for $Q \in \{\exists, \forall\}$ then $nnf(F) = \bar{Q}x\ nnf(\neg\varphi(x))$
  - If $F = \neg\varphi \square \psi$ for $\square \in \{\wedge, \vee\}$ then $nnf(F) = nnf(\neg\varphi(x))\ \bar{\square}\ nnf(\neg\psi(x))$
  - If $F = \neg P$ for an atom P then $nnf(F) = \neg P$.
- The goal is to push negation to the literals.

Clausal Form

- After translation to nnf, quantifiers may be prenexified.
  - Move quantifier to the outer most scope (without switching order)
- Next we can skolemize the the $\exists$ quantifiers.
- Confusing? Which quantifier to skolemize depends on the context. (Sometimes called Herbrandization)
- Resolution is a refutation calculus, $\exists$ quantifiers denote arbitrary terms (**Strong quantifiers**).
- **Skolemization**?
- Semantically valid syntax extension based on the quantifier structure (not unique!)

$$\forall x \exists y p(x, y) \vee \forall w \exists r q(w, r)$$

$$\forall x \exists y \forall w \exists r (p(x, y) \vee q(w, r)) \qquad \forall w \exists r \forall x \exists y (p(x, y) \vee q(w, r))$$

$$\forall x \forall w (p(x, a) \vee q(w, f(x, w))) \qquad \forall w \forall x (p(x, f(w, x)) \vee q(w, a))$$

Clausal Form

- Skolemization can be done without prenexing. (**More efficient!**)
- These transformations result in a formula of the form

$$\forall x_1 \cdots \forall x_n F$$

  where F is quantifier-free.
- At this point the quantifiers can be removed as variables in different clauses can be treated independently.
- Now we can cover the whole process to CNF.

Translation to clausal form

- Consider F =

$$\forall x \exists y (P(x,y) \wedge \forall u \forall v (P(u,v) \rightarrow R(u))) \rightarrow \forall z R(z)$$

- Remove implications

$$\neg(\forall x \exists y (P(x,y) \wedge \forall u \forall v (\neg P(u,v) \vee R(u)))) \vee \forall z R(z)$$

- Negate the formula

$$\forall x \exists y (P(x,y) \wedge \forall u \forall v (\neg P(u,v) \vee R(u))) \wedge \neg \forall z R(z)$$

- Convert to nnf

$$\forall x \exists y (P(x,y) \wedge \forall u \forall v (\neg P(u,v) \vee R(u))) \wedge \exists z \neg R(z)$$

Translation to clausal form

- Prenex (Not entirely necessary)

$$\exists z \forall x \exists y \forall u \forall v (P(x,y) \wedge (\neg P(u,v) \vee R(u)) \wedge \neg R(z))$$

- Skolemize

$$\forall x \forall u \forall v (P(x, f(x)) \wedge (\neg P(u,v) \vee R(u)) \wedge \neg R(a))$$

- As clause set

$$\{P(x, f(x)) \ , \ \neg P(u,v) \vee R(u) \ , \ \neg R(a)\}$$

- Notice it is unsatisfiable.

Towards Completeness of Resolution

- We have simplified the syntactic structure of FOL formula.
- However, to show that a formula is unsatisfiable, we still need to show that no interpretation satisfies it.
- There are uncountably infinite interpretations.
- We restrict ourselves to a type of interpretation which is representative of the entire set of interpretations.

Herbrand Universe

- Let C be a finite set of clauses.
- $CS(C)$ and $FS(C)$ denote the constant symbols and function symbols occuring in C, respectively.

$$H_0 = \begin{cases} CS(C) & CS(C) \neq \emptyset \\ \{a\} & if\, CS(C) = \emptyset \end{cases}$$

$$H_i = H_{i-1} \cup \{f(t_1, \cdots t_n) | f \in FS(C)\ ,\ t_1, \cdots t_n \in H_{i-1}\}$$

- $H(C) = \bigcup_{i=0}^{\infty} H_i$.
- We refer to $H(C)$ as the Herbrand universe of C.

Herbrand Universe

- Consider the clause set

$$\{(\neg P(x) \vee P(f(x)))\ ,\ P(h(x,x))\ ,\ (\neg P(h(u,v)) \vee \neg Q(v))\}$$

$$H_0 = \{a\}$$

$$H_1 = \{a, f(a), h(a,a)\}$$

$$H_2 = \{a, f(a), f(f(a)), f(h(a,a)), h(a,a), h(f(a),a), h(a,f(a))$$
$$h(f(a),f(a)), h(f(a),h(a,a)), h(h(a,a),f(a)), h(h(a,a),h(a,a))\}$$

- Essentially, it is the set of terms constructable from the symbols occurring in the clause set.
- Using the Herbrand universe we can construct a Herbrand interpretation.
- An interpretation with domain $H(C)$ and interpretation function mapping the constructors to themselves.

H-interpretation correspondence

- We will denote interpretations by a triple (D, Φ, I) where D is the domain, Φ interpretation function, and $I : V \to H(C)$ the environment.
- We associate with each interpretation (D, Φ, I) a function $\omega : H(C) \to D$ which is faithful to the construction of the Herbrand universe.
- A corresponding H-interpretation $(H(C), \Phi_H, J)$ is an H-interpretation with the following condition on Φ_H:

$$\Phi_H(P)(t_1, \cdots, t_n) = \Phi(P)(\omega(t_1), \cdots, \omega(t_n))$$

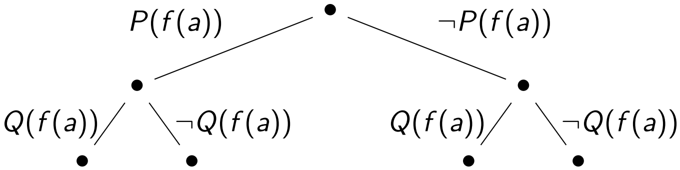
- for all $P \in PS(C)$ and $t_i \in H(C)$
- $PS(C)$ denotes the predicate symbols of C.

Restriction to H-Models

- A set of clauses C is satisfiable iff it has an H-Model.
  - $\leftarrow$ If C has an H-Model then it is trivially satisfiable.
  - $\rightarrow$ we can instead consider:
    If C does not have an H-model then it is unsatisfiable.
- This implies that all H-interpretations falsify C.
- For any interpretation we can construct a corresponding H-interpretation.
- We need to show that reversing this construction preserves falsifiability.
- There is a $d \in C$ which is falsified by the H-Model. In can be shown by induction over term depth that a ground substitution of the terms within the clause exists which coincides with the H-models semantic interpretation.
- The rest follows from the construction of a corresponding H-interpretation.

Semantic Trees

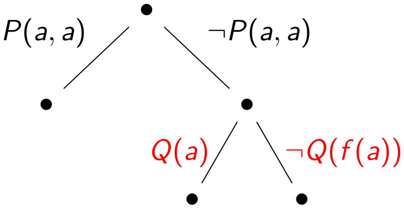
- The restriction to H-Models depends on grounding the terms occurring in C.
- For a given clause set C and Herbrand universe H(C) we can construct a so called Semantic tree containing partial truth assignments of the predicates occurring in C.

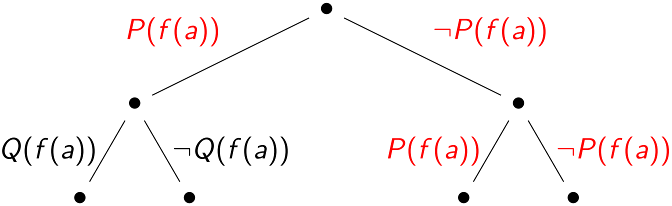


- Every node can be expanded by a positive and negative edge.
- The same symbols and tuple of terms is used on each branch at each expansion step.
- Same expansion step cannot be repeated.

Semantic Trees

- These are not a semantic trees:

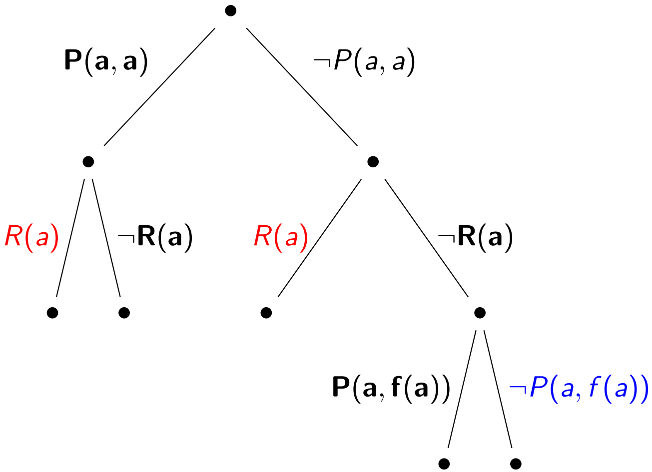




Semantic Trees

- A complete semantic tree can be built iteratively.
- Order the predicate symbols of C and terms of $H(C)$.
- then continuously expand with respect to the order.
- After a number of expansion steps a branch may contain a ground instance which falsifies a clause.
- such nodes of the tree are referred to as failure nodes.
- failure nodes are not expanded.
- If every branch ends in a failure node, then the tree is closed.

Semantic Trees



$$\{P(x, f(x)),\ \neg \mathbf{P}(\mathbf{u}, \mathbf{v}) \vee \mathbf{R}(\mathbf{u}),\ \neg R(a)\}$$

Semantic trees and unsatisfiably

- A set of clauses C is unsatisfiable iff its semantic tree T is closed.
  - $\leftarrow$ T tells us how to build a falsifying H-models.
  - $\rightarrow$ Each branch represents an H-model. We know no H-model satisfies C thus T must eventually close.
- A closed semantic tree T for a set of clauses C is finite.
  - For T to be infinite, being that it is finitely branching, there would have to be an infinite path. But that contradicts the definition of failure node.
- This gives us a crude automated theorem prover, but it is complete.

Semantic Trees and Ground Instances

- Notice that a closed semantic tree T of a set of clauses C proves unsatisfiably by collecting a set of ground instances of the clauses of C.
- This set of ground instances is enough to prove unsatisfiably of C.
- This observation provides a variant of Herbrand's theorem (**Soon!**)

  A set of clauses C is unsatisfiable iff there exists a finite unsatisfiable set of clauses C' such that C' consists of ground instances of clauses in C.
- The method of Davis and Putnam is based on saturation of sets of ground instances of clauses.
- Resolution improves on this and earlier methods by avoiding the search for ground instances.

Propositional (Ground) Resolution

- Let us consider resolution without substitution first:

$$\frac{\Delta \vee C \qquad \Delta' \vee \neg C}{\Delta \ \vee \ \Delta'} \text{Res}$$

- Soundness of the rule is easy to observe:

  if $(\Delta \vee C) \wedge (\Delta' \vee \neg C)$ is true then $\Delta \ \vee \ \Delta'$ is true.

- However, an additional step is needed in some cases:

$$\frac{C \vee C \qquad \neg C \vee \neg C}{C \ \vee \ \neg C} \text{Res}$$

- $(C \vee C) \wedge (\neg C \vee \neg C)$ is pretty unsatisfiable.
- To avoid this issue we need to add a contraction rule (can be built into the resolution rule).
- Often referred to as **factoring**.

Completeness of Propositional Resolution

⋆ If C is an unsatisfiable set of propositional clauses then there exists a refutation of C.

We can prove this statement by induction on the height of the semantic tree T of C
Note that $H(C)$ is trivial for a propositional clause set.

BC If the T has height 1 then C contains $\bot$.

SC Assume ⋆ holds for all clause sets C' whose tree T' is of height n, we show that the statement holds for C whose tree T is of height $n+1$.

- Notice that nodes at level n connect to failure nodes at level $n+1$ through edges labeled by complementary literals.
- Let $C_1 \vee P$ and $C_2 \vee \neg P$ be the clauses corresponding to the failure nodes.

Completeness of Propositional Resolution

- Using resolution and contraction we can build the clause set $C \cup \{C_1' \vee C_2'\}$ where
  - $C_1' \vee C_2'$ is equivalent $C_1 \vee C_2$ after contraction.
  - T' is equivalent to T with the failure nodes corresponding to $C_1 \vee P$ and $C_2 \vee \neg P$ removed.
  - The node at level n is now a failure node for $C_1' \vee C_2'$.
- Repeating this process for all nodes at level n we get a semantic tree of height n.
- Hint: Useless edges may have to be removed.
- This can be easily generalized from propositional (ground) clause sets to first-order.
- The branches need only contain instances of the clauses in C.
- A closed tree can always be grounded.

More General Resolution

- Consider the clause set:

$$P(x, f(y)) \vee P(x, f(x)) \ , \ \neg P(x, y) \vee P(y, x) \ ,$$

$$\neg P(x, y) \vee P(f(x), y) \ , \ \neg P(f(f(x)), x)$$

- It can be refuted as follows:

$$\cfrac{\cfrac{\cfrac{P(x, f(y)) \vee P(x, f(x)) \qquad \neg P(x, y) \vee P(y, x)}{P(f(x), x)}\text{Res} \qquad \neg P(x, y) \vee P(f(x), y)}{P(f(f(x)), x)}\text{Res} \qquad \neg P(f(f(x)), x)}{\bot}\text{Res}$$

- Finding the substitution is similar to the search for ground instantiations.
- However, there is a special type of unifier which allows resolution to be more efficient than ground instantiation methods.

Generality Order

- There are may be infinitely many unifiers of two terms.
- We may order the unifiers by generality in the following sense.
- We say σ_1 is more general than σ_2, $\sigma_1 \leq \sigma_2$, if $\sigma_1\tau = \sigma_2$.
- For example,

$$f(x)\{x \leftarrow g(a,a)\ ,\ y \leftarrow a\} = f(g(y,a))\{x \leftarrow g(a,a)\ ,\ y \leftarrow a\}$$

$$f(x)\{x \leftarrow g(y,a)\} = f(g(y,a))\{x \leftarrow g(a,a)\}$$

- Notice that

$$\{x \leftarrow g(y,a)\ ,\ y \leftarrow y\}\{y \leftarrow a\} = \{x \leftarrow g(a,a)\ ,\ y \leftarrow a\}$$

- Thus, $\{x \leftarrow g(y,a)\} \leq \{x \leftarrow g(a,a)\ ,\ y \leftarrow a\}$

Most General Unifier

- A unifier σ is an mgu if for all unifiers τ, $\sigma \leq \tau$
- For first-order term expressions if two terms are unifiable then there is a unique mgu, up to variable renaming, unifying them.
- $\{x \leftarrow g(y, a)\}$ is the mgu for the previous example.
- It is decidable if two first-order term expressions have an mgu.
- Computing the mgu naively requires exponential time, but it is computable in nearly linear.
- See the **Martelli-Montanari Algorithm** for unification.

Constructing MGUs

- By $diff(t_1, t_2)$ we denote the pairs of subterms (with matching positions) which do not match when decomposing t_1 and t_2 top-down.
- For example $diff(g(\mathbf{x}, f(a, b), h(b)), g(\mathbf{f}(\mathbf{a}, \mathbf{b}), f(y, b), f(b, b)))$ contains

$$(\mathbf{x}, \mathbf{f}(\mathbf{a}, \mathbf{b}))\ ,\ (a, y)\ ,\ (h(b), f(b, b))$$

- Notice that $(h(b), f(b, b))$ cannot be unified and thus, these two term are not unifiable.
- Consider $diff(x, f(a, x)) = \{(x, f(a, x))\}$
- This seems unifiable, but $(x, f(a, x))$ implies our mgu ought to contain the substitution $\{x \leftarrow f(a, x)\}$.

$$x\{x \leftarrow f(a, x)\} = f(a, x) \neq f(a, f(a, x)) = f(a, x)\{x \leftarrow f(a, x)\}$$

- Results in an infinite loop.

Constructing MGUs

- Thus, unification fails if $diff(t_1, t_2)$ contains
  - terms with different head symbols.
  - a pair of the form $(x, t[x])$.

Require: $\sigma = Id$
  while $diff(t_1\sigma, t_2\sigma) \neq \emptyset$ **do**
    if $(x, t[x]), (f(\bar{t}), g(\bar{s})) \in diff(t_1\sigma, t_2\sigma)$ **then**
      return Fail
    else
      Select $(s, t) \in diff(t_1\sigma, t_2\sigma)$
      if s is a variable **then**
        $\sigma = \sigma\{s \leftarrow t\}$
      else
        $\sigma = \sigma\{t \leftarrow s\}$
      end if
    end if
  end while
  return σ

- Exponential behavior occurs because the substitution may apply to itself, i.e. can build full binary trees.

MGUs and Resolution

- We can lift MGU to the clausal level by lifting the generality order to clauses:
  - Let C, D be clauses and C', D' clauses resulting from C, D by contraction. Then $C \leq D$ implies $C' \leq D'$.
  - Let C, D, C', D' be clauses such that $C \leq C'$ and $D \leq D'$. If E' is the result of resolving C' and D' then there exists a resolvent E of C and D with $E \leq E'$.
- This statement is usually referred to as the lifting lemma.
- We can further generalize the lemma to full resolution derivations implying that we only need to consider MGUs when applying resolution.

Lifting Theorem

Theorem

Let C be a set of clauses and C' be a set of instances of clauses in C. Let Δ be a resolution deduction from C'. Then there exists a resolution deduction Γ from C such that $\Gamma \leq \Delta$.

Theorem (completeness)

If C is an unsatisfiable set of clauses then there exists an resolution refutation of C.

Proof.

We can lift ground resolution refutations to non-ground resolution refutation by the lifting theorem. □

How big can refutations get?

- Refutations can be non-elementary in the size of the formula.
  - That is faster growing than $f(0) = 1 \;\;,\;\; f(n+1) = 2^{f(n)}$
- Statman constructed a sequence of clause sets, using Combinatory logic, whose refutations grow faster than $f(n)$.

  In *"Lower bounds on Herbrand's theorem"*.
- Here is an example of a simple but hard to refute clause set:

$$E(x,a) \vee E(x,b) \vee E(x,c) \;,\; \neg E(x,a) \vee \neg E(y,a) \vee L(s(y),x)$$

$$\neg E(x,b) \vee \neg E(y,b) \vee L(s(y),x) \;,\; \neg E(x,c) \vee \neg E(y,c) \vee L(s(y),x)$$

$$\neg L(m(x,y),z) \vee L(x,z) \;,\; \neg L(m(x,y),z) \vee L(y,z) \;,\; L(x,x)$$

- Requires the derivation of 100s of clauses to refute.
- Many interesting problems can be found at

  http://www.tptp.org/

Goals

- Introduce automated reasoning and the resolution calculus
- Completeness of the resolution calculus
- **the Sequent calculus**
- **Cut-elimination**
- **Eliminating cuts using resolution**

Background: Gentzen's Sequent Calculus

- The sequent calculus applies inferences to objects referred to as sequents $\Delta \vdash \Pi$, where Δ and Π are multisets of well-formed formula. Chaining inferences forms proof trees.
- Semantically, a sequent means *given* Δ *we may derive* Π.
- Note that, this interpretation implies that Δ is essentially a conjunction of formula and Π is a disjunction.
- The sequent calculus inferences are as follows:

  Axiom Inferences

$$\frac{}{A \vdash A}\,\mathrm{Ax}$$

Gentzen's Sequent Calculus

<u>Structural Inferences</u>

$$\frac{\Gamma \vdash \Delta}{D, \Gamma \vdash \Delta}\ \text{w:l} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, D}\ \text{w:r}$$

$$\frac{D, D, \Gamma \vdash \Delta}{D, \Gamma \vdash \Delta}\ \text{c:l} \qquad \frac{\Gamma \vdash \Delta, D, D}{\Gamma \vdash \Delta, D}\ \text{c:r}$$

$$\frac{\Gamma \vdash \Delta, C \qquad C, \Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ \text{cut}$$

Gentzen's Sequent Calculus

Logical Inferences

$$\frac{\Gamma \vdash \Delta, D}{\neg D, \Gamma \vdash \Delta}\ \neg{:}\mathrm{l} \qquad \frac{D, \Gamma \vdash \Delta}{\Gamma \vdash \Delta, \neg D}\ \neg{:}\mathrm{r} \qquad \frac{C, \Gamma \vdash \Delta}{C \wedge D, \Gamma \vdash \Delta}\ \wedge{:}\mathrm{l}$$

$$\frac{D, \Gamma \vdash \Delta}{C \wedge D, \Gamma \vdash \Delta}\ \wedge{:}\mathrm{l} \qquad \frac{\Gamma \vdash \Delta, C}{\Gamma \vdash \Delta, C \vee D}\ \vee{:}\mathrm{r} \qquad \frac{\Gamma \vdash \Delta, D}{\Gamma \vdash \Delta, C \vee D}\ \vee{:}\mathrm{r}$$

$$\frac{\Gamma \vdash \Delta, C \qquad \Gamma \vdash \Delta, D}{\Gamma \vdash \Delta, C \wedge D}\ \wedge{:}\mathrm{r} \qquad \frac{C, \Gamma \vdash \Delta \qquad D, \Gamma \vdash \Delta}{C \vee D, \Gamma \vdash \Delta}\ \vee{:}\mathrm{l}$$

$$\frac{C, \Gamma \vdash \Delta, D}{\Gamma \vdash \Delta, C \rightarrow D}\ \rightarrow{:}\mathrm{r} \qquad \frac{\Gamma \vdash \Delta, C \qquad D, \Gamma \vdash \Delta}{C \rightarrow D, \Gamma \vdash \Delta}\ \rightarrow{:}\mathrm{l}$$

Gentzen's Sequent Calculus

Quantifier Inferences

$$\frac{\Gamma \vdash \Delta, F(\alpha)}{\Gamma \vdash \Delta, \forall x F(x)} \forall{:}\mathrm{r} \qquad \frac{F(t), \Gamma \vdash \Delta}{\forall x F(x), \Gamma \vdash \Delta} \forall{:}\mathrm{l}$$

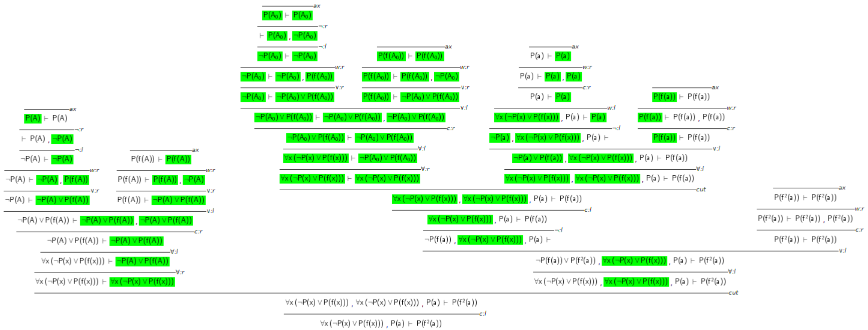
$$\frac{\Gamma \vdash \Delta, F(t)}{\Gamma \vdash \Delta, \exists x F(x)} \exists{:}\mathrm{r} \qquad \frac{F(\alpha), \Gamma \vdash \Delta}{\exists x F(x), \Gamma \vdash \Delta} \exists{:}\mathrm{l}$$

- Note that for $\exists : l$ and $\forall : r$ α may not occur in Γ or Δ. These rules are referred to as strong quantification, i.e. require an eigenvariable, the other rules are referred to as weak.

Simple Sequent Calculus Proof

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(\alpha) \vdash P(\alpha)}{P(\alpha) \vdash P(\alpha), P(\beta)}\ \text{w:r}}{\vdash P(\alpha), P(\alpha) \rightarrow P(\beta)}\ \Longrightarrow : r}{\vdash P(\alpha), \forall y(P(\alpha) \rightarrow P(y))}\ \forall : r}{\vdash P(\alpha), \exists x \forall y(P(x) \rightarrow P(y))}\ \exists : r}{P(a) \vdash P(\alpha), \exists x \forall y(P(x) \rightarrow P(y))}\ w : l}{\vdash P(a) \rightarrow P(\alpha), \exists x \forall y(P(x) \rightarrow P(y))}\ \Longrightarrow : r}{\vdash \forall y(P(a) \rightarrow P(y)), \exists x \forall y(P(x) \rightarrow P(y))}\ \forall : r}{\vdash \exists x \forall y(P(x) \rightarrow P(y)), \exists x \forall y(P(x) \rightarrow P(y))}\ \exists : r}{\vdash \exists x \forall y(P(x) \rightarrow P(y))}\ c : r
$$

Sequent Calculus Proof With Cut



- Green formulas denote **cuts** and their ancestors.
- The cut rule is used in introduce *auxiliary arguments* (**Lemmas**) into a proof.
  - That is concepts external to the statement being proven.

Proof without Cut

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$P(a) \vdash P(a)$}
\RightLabel{$\neg{:}l$}
\UnaryInfC{$\neg P(a), P(a) \vdash$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$P(f(a)) \vdash P(f(a))$}
\RightLabel{$\vee{:}l$}
\BinaryInfC{$\neg P(a) \vee P(f(a)), P(a) \vdash P(f(a))$}
\RightLabel{$\forall{:}l$}
\UnaryInfC{$\forall x(\neg P(x) \vee P(f(x))), P(a) \vdash P(f(a))$}
\RightLabel{$\neg{:}l$}
\UnaryInfC{$\neg P(f(a)), \forall x(\neg P(x) \vee P(f(x))), P(a) \vdash$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$P(f^2(a)) \vdash P(f^2(a))$}
\RightLabel{$\vee{:}l$}
\BinaryInfC{$\neg P(f(a)) \vee P(f^2(a)), \forall x(\neg P(x) \vee P(f(x))), P(a) \vdash P(f^2(a))$}
\RightLabel{$\forall{:}l$}
\UnaryInfC{$\forall x(\neg P(x) \vee P(f(x))), \forall x(\neg P(x) \vee P(f(x))), P(a) \vdash P(f^2(a))$}
\RightLabel{$c{:}l$}
\UnaryInfC{$\forall x(\neg P(x) \vee P(f(x))), P(a) \vdash P(f^2(a))$}
\end{prooftree}
$$

- Proofs without cuts are referred to as **analytic** proofs.
  - Every formula is a subformula of the end sequent.
- Observe that **cut** is the only rule that removes formula from the sequent.
- If one could derive $\bot$ using the sequent calculus, then it would involve **cut**.

Eliminating Cut:Proof

- **Gentzen**'s *Hauptsatz*: cuts can be eliminated.
- We assume proofs have been **Regularized**.
  - Unique eigenvariable for each Strong quantifier.
- Additionally, a generalization of the **cut rule** is used.

$$\frac{\Gamma \vdash \Delta, C \qquad C, \Gamma \vdash \Delta}{\Gamma' \vdash \Delta'}\,\text{mix}$$

where Γ' and Δ' are equivalent to Γ and Δ but with every instance of C removed.

Eliminating Cut: Proof

- Assume a proof with a single mix as the last inference

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash \Delta, C} \quad \dfrac{\vdots}{C, \Gamma \vdash \Delta}}{\Gamma' \vdash \Delta'} \text{mix}$$

- The induction is on two properties of C:
  - **Grade**: logical complexity of C.
  - **Rank**: Distance from introduction. (Not just axiom rule)

Eliminating Cut: Proof

- Grade is computed as follows:
  - If F is atomic then $G(F) = 1$,
  - If $F = \neg A$ where A is a formula of arbitrary complexity then $G(F) = G(A) + 1$.
  - If $F = A \star B$ where A and B are formula of arbitrary complexity and $\star \in \{\wedge, \vee, \implies\}$ then $G(F) = G(A) + G(B) + 1$,
  - If $F = QxA(x)$ where $A(x)$ is a formula of arbitrary complexity and $Q \in \{\exists, \forall\}$ then $G(F) = G(A) + 1$.
- Rank is computed as follows:

$$rank_l(P) = \max_{\mathcal{T}}(rank(\mathcal{T}, P))$$

$$rank_r(P) = \max_{\mathcal{T}}(rank(\mathcal{T}, P))$$

$$rank(P) = rank_l(P) + rank_r(P)$$

- Here $\mathcal{T}$ is a **thread**.

  Connected path from P in the cut to it's introduction.
- Contraction creates multiple instances.

Eliminating Cut: Rank 2

- Atomic formula introduced right before the mix:

$$\frac{A \vdash A \qquad \Gamma \vdash \Delta}{A, \Gamma' \vdash \Delta}\ \text{mix} \qquad\qquad \frac{A \vdash A \qquad \Gamma \vdash \Delta}{\Gamma \vdash \Delta', A}\ \text{mix}$$

- A proof without mix is possible using contraction and/or weakening:

$$\frac{\dfrac{\Gamma \vdash \Delta}{\vdots}}{\Gamma', A \vdash \Delta, A} \qquad\qquad \frac{\dfrac{\Gamma \vdash \Delta}{\vdots}}{\Gamma \vdash \Delta', A}$$

- What if A was introduced by weakening?

Eliminating Cut: Rank 2

- Four cases, only two below:

$$\dfrac{\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A} \qquad \Pi \vdash \Lambda}{\Gamma, \Pi' \vdash \Delta, \Lambda}\ \text{mix} \qquad\qquad \dfrac{\Pi \vdash \Lambda \qquad \dfrac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}}{\Gamma, \Pi \vdash \Delta, \Lambda'}\ \text{mix}$$

- A proof without mix is possible using weakening:

$$\dfrac{\dfrac{\Gamma \vdash \Delta}{\vdots}}{\Gamma, \Pi' \vdash \Delta, \Lambda} \qquad\qquad \dfrac{\dfrac{\Gamma \vdash \Delta}{\vdots}}{\Gamma, \Pi \vdash \Delta, \Lambda'}$$

- Now we need to consider logic rules (Still rank 2)?

Eliminating Cut: Rank 2

- We only consider a few principle cases.
- For conjunction we have the following:

$$\dfrac{\dfrac{\Gamma \vdash \Delta, B \qquad \Gamma \vdash \Delta, C}{\Gamma \vdash \Delta, B \wedge C} \qquad \dfrac{B, \Pi \vdash \Lambda}{B \wedge C, \Pi \vdash \Lambda}}{\Gamma, \Pi' \vdash \Delta', \Lambda}\text{mix}$$

- Notice that B occurs both on the left side and right side :

$$\dfrac{\Gamma \vdash \Delta, B \qquad B, \Pi \vdash \Lambda}{\Gamma, \Pi' \vdash \Delta', \Lambda}\text{mix}$$

- The resulting proof has lower logical complexity and can be handled by the earlier cases.

Eliminating Cut: Rank 2

- We only consider a few principle cases.
- For the universal quantifier we have the following:

$$\dfrac{\dfrac{\Gamma \vdash \Delta F(a)}{\Gamma \vdash \Delta, \forall x F(x)} \qquad \dfrac{F(t), \Pi \vdash \Lambda}{\forall x F(x), \Pi \vdash \Lambda}}{\Gamma, \Pi' \vdash \Delta', \Lambda}\text{mix}$$

- We can replace the eigenvariable by the term on the right side (remember regularization)

$$\dfrac{\Gamma \vdash \Delta, F(t) \qquad F(t), \Pi \vdash \Lambda}{\Gamma, \Pi' \vdash \Delta', \Lambda}\text{mix}$$

- The resulting proof has lower logical complexity and can be handled by the earlier cases.

Eliminating Cut: Rank > 2

- Consider the case of an atomic formula

$$\frac{\Gamma \vdash \Delta, A \qquad \Pi, A \vdash \Lambda}{\Gamma, \Pi' \vdash \Delta', \Lambda} \text{mix}$$

- The rank is the maximum thread length.
- We can reduce it by introducing contractions prior to the **mix**

$$\frac{\dfrac{\overline{\Gamma \vdash \Delta}}{\vdots} \qquad \dfrac{\overline{\Pi \vdash \Lambda}}{\vdots}}{\begin{array}{cc}\Gamma \vdash \Delta^*, A & A, \Pi^* \vdash \Lambda\end{array}}$$

$$\frac{\Gamma \vdash \Delta^*, A \qquad A, \Pi^* \vdash \Lambda}{\Gamma, \Pi' \vdash \Delta', \Lambda} \text{mix}$$

- The resulting proof has a lower rank with respect to A and can be handled by the earlier cases.

Eliminating Cut: Rank > 2

- Consider structural rules (contraction and Weakening):

$$\dfrac{\Gamma \vdash \Delta \qquad \dfrac{\Phi \vdash \Psi}{\Pi \vdash \Lambda}\, J}{\Gamma, \Pi' \vdash \Delta', \Lambda}\, \text{mix}$$

- the main fromula of J cannot be the same as the **mix** (rank 2 case)

$$\dfrac{\dfrac{\Gamma \vdash \Delta \qquad \Phi \vdash \Psi}{\Gamma, \Phi^* \vdash \Delta^*, \Psi}\, \text{mix}}{\Gamma, \Pi^* \vdash \Delta^*, \Lambda}\, J$$

- We can move the mix above J and reduce the rank.
- The resulting proof has a lower rank with respect to the main formula of the mix

Eliminating Cut: Rank > 2

- Consider binary logic rules:

$$\dfrac{\Gamma \vdash \Delta \qquad \dfrac{C, \Pi_1 \vdash \Lambda_1 \qquad \Pi_2 \vdash \Lambda_2, B}{B \rightarrow C, \Pi_1, \Pi_2 \vdash \Lambda_1, \Lambda_2} \rightarrow: l}{\Gamma, \Pi_1^*, \Pi_2^* \vdash \Delta^*, \Lambda_1, \Lambda_2} \text{mix}$$

- The main formula of the mix is contained in both Π_1 and Π_2

$$\dfrac{\dfrac{\Gamma \vdash \Delta \qquad C, \Pi_1 \vdash \Lambda_1}{C, \Gamma, \Pi_1^* \vdash \Delta^*, \Lambda_1} \text{mix} \qquad \dfrac{\Gamma \vdash \Delta \qquad \Pi_2 \vdash \Lambda_2, B}{\Gamma, \Pi_2^* \vdash \Delta^*, \Lambda_2, B} \text{mix}}{B \rightarrow C, \Gamma, \Pi_1^*, \Pi_2^* \vdash \Delta^*, \Lambda_1 \Lambda_2} \rightarrow: l$$

- This introduces an additional mix, but the rank is reduced.
- The induction hypothesis holds on the two branches.

Eliminating Cut: Rank > 2

- Consider the existential quantifier:

$$\dfrac{\Gamma \vdash \Delta \qquad \dfrac{F(a), \Pi \vdash \Lambda}{\exists x F(x), \Pi \vdash \Lambda}}{\Gamma, \Pi * \vdash \Delta^*, \Lambda}\,\text{mix}$$

- If regularized a fresh eignenvariable is unnecessary.

$$\dfrac{\Gamma \vdash \Delta \qquad \dfrac{\dfrac{\Gamma \vdash \Delta \qquad F(b), \Pi \vdash \Lambda}{F(b), \Gamma, \Pi^* \vdash \Delta^*, \Lambda}\,\text{mix}}{\exists x F(x), \Gamma, \Pi^* \vdash \Delta^*, \Lambda}}{\Gamma, \Pi * \vdash \Delta^*, \Lambda}$$

- Swapping the quantifier rule and mix reduces rank.

Eliminating Cut: Algorithm

- The above rules only apply to the upper most mix (**cut**).
- To eliminate cuts from proofs we first eliminate the upper most cuts.
- Then we incrementally proceed towards the root.
- What's the complexity of the algorithm??

- Then What is it good for?

Eliminating Cut: Algorithm

- The above rules only apply to the upper most mix (**cut**).
- To eliminate cuts from proofs we first eliminate the upper most cuts.
- Then we incrementally proceed towards the root.
- What's the complexity of the algorithm??

  "Don't Eliminate Cut" by George Boolos
- Eliminating cut can produce a proof with size non-elementarily larger than the input proof!!
- Remember the refutation from earlier?
- Then What is it good for?

Cut-freeness and the Herbrand Instances

Theorem (Mid-Sequent Theorem)

Let S be a sequent of prenex formulas then there exists a cut-free proof π of S s.t. π contains a sequent S' s.t.

- *S' is quantifier free.*
- *Every inference above S' is structural or propositional.*
- *Every inference below S' is structural or a quantifier inference.*

What if we limit S to a sequent only containing weak quantification.

We can generalize **Skolemization** from clause sets to proofs.

Cut-freeness and the Herbrand Instances

- No strong quantification means no eigenvariables and thus all terms are existential witnesses.
- Collecting those witnesses gives us Herbrand's Theorem

Cut-freeness and the Herbrand Instances

- No strong quantification means no eigenvariables and thus all terms are existential witnesses.
- Collecting those witnesses gives us Herbrand's Theorem

Theorem (Herbrand's Theorem)

Let S be a sequent of the form $\forall \bar{x} \varphi(\bar{x}) \vdash \exists \bar{x} \psi(\bar{x})$. S is valid if and only if there exists a sequence of term vectors $\bar{t}_1, \cdots, \bar{t}_n$ s.t.

$$\bigwedge_{i=0}^{k} \varphi(\bar{t}_i) \vdash \bigvee_{i=0}^{k} \psi(\bar{t}_i)$$

is valid.

Cut-freeness and the Herbrand Instances

- No strong quantification means no eigenvariables and thus all terms are existential witnesses.
- Collecting those witnesses gives us Herbrand's Theorem

Theorem (Herbrand's Theorem)

Let S be a sequent of the form $\forall \bar{x} \varphi(\bar{x}) \vdash \exists \bar{x} \psi(\bar{x})$. S is valid if and only if there exists a sequence of term vectors $\bar{t}_1, \cdots, \bar{t}_n$ s.t.

$$\bigwedge_{i=0}^{k} \varphi(\bar{t}_i) \vdash \bigvee_{i=0}^{k} \psi(\bar{t}_i)$$

is valid.

- Cut-free (weakly quantified end sequent) $\implies$ *weak* mid-sequent $\implies$ Herbrand instances.

An Herbrand Sequent

- Consider the sequent $F \vdash$ where F contains:

$$\bigvee_{i=0}^{n} f(x) = 0 \vee f(x) = 1,$$

$$s(x) \not\leq y \vee f(x) \neq 0 \vee f(y) \neq 0$$

$$s(x) \not\leq y \vee f(x) \neq 1 \vee f(y) \neq 1$$

$$max(x, y) \leq z \rightarrow x \leq z$$

$$max(x, y) \leq z \rightarrow y \leq z$$

$$\forall x(x \leq x)$$

- Note that $F \vdash$ is provable.
- and we can prove it without cut.
- what does the Herbrand sequent look like?

An Herbrand Sequent

$$\mathbf{1:}\ \forall A_0 \forall B \forall C \begin{array}{c} \langle\, g^2(U)\,,\, g(U)\,,\, max(g^2(U), g(U))\,\rangle \\ \langle\, g(U)\,,\, g^2(U)\,,\, max(g(U), g^2(U))\,\rangle \\ \langle\, g(U)\,,\, g(U)\,,\, max(g(U), g(U))\,\rangle \end{array} (\,\neg\, LEQ(max(A_0, B), C) \vee LEQ(B, C)\,)$$

$$\mathbf{2:}\ \forall A_0 \forall B \forall C \begin{array}{c} \langle\, g^2(U)\,,\, g(U)\,,\, max(g^2(U), g(U))\,\rangle \\ \langle\, g(U)\,,\, g^2(U)\,,\, max(g(U), g^2(U))\,\rangle \\ \langle\, g(U)\,,\, g(U)\,,\, max(g(U), g(U))\,\rangle \end{array} (\,\neg\, LEQ(max(A_0, B), C) \vee LEQ(A_0, C)\,)$$

$$\mathbf{3:}\ \forall A \begin{array}{c} \langle\, g(U)\,\rangle \\ \langle\, max(g(U), g(U))\,\rangle \\ \langle\, U\,\rangle \\ \langle\, max(g^2(U), g(U))\,\rangle \\ \langle\, max(g(U), g^2(U))\,\rangle \end{array} (\,E(f(A), s(0)) \vee E(f(A), 0)\,)$$

$$\mathbf{4:}\ \forall A \begin{array}{c} \langle\, g(U)\,\rangle \\ \langle\, max(g(U), g(U))\,\rangle \\ \langle\, max(g(U), g^2(U))\,\rangle \\ \langle\, max(g^2(U), g(U))\,\rangle \end{array} LEQ(A, A)$$

$$\mathbf{5:}\ \forall B_1 \forall A_2 \begin{array}{c} \langle\, U\,,\, max(g^2(U), g(U))\,\rangle \\ \langle\, U\,,\, g(U)\,\rangle \\ \langle\, U\,,\, max(g(U), g(U))\,\rangle \\ \langle\, g(U)\,,\, max(g(U), g^2(U))\,\rangle \end{array} (\,(\,\neg\, LEQ(g(B_1), A_2) \vee \neg\, E(f(B_1), s(0))\,) \vee \neg\, E(f(A_2), s(0))\,)$$

$$\mathbf{6:}\ \forall B_0 \forall A_1 \begin{array}{c} \langle\, U\,,\, g(U)\,\rangle \\ \langle\, U\,,\, max(g(U), g(U))\,\rangle \\ \langle\, g(U)\,,\, max(g^2(U), g(U))\,\rangle \\ \langle\, U\,,\, max(g(U), g^2(U))\,\rangle \end{array} (\,(\,\neg\, LEQ(g(B_0), A_1) \vee \neg\, E(f(B_0), 0)\,) \vee \neg\, E(f(A_1), 0)\,)$$

How else to think about cut?

- How different are these two rules?

$$\frac{\Gamma \vdash \Delta, C \qquad C, \Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,\text{cut} \qquad \frac{\Gamma \vdash \Delta, C \qquad D, \Gamma \vdash \Delta}{C \rightarrow D, \Gamma \vdash \Delta}\,\rightarrow\text{:l}$$

- What if we replace **cuts** by $\rightarrow$**:l**

$$\frac{\Gamma \vdash \Delta, C \qquad C, \Gamma' \vdash \Delta'}{\Gamma, \Gamma', C \rightarrow C \vdash \Delta, \Delta'}\,\text{cut?}$$

How else to think about cut?

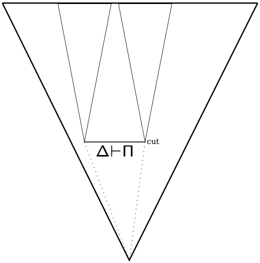
$$\dfrac{\vdots \qquad\qquad\qquad\qquad \vdots}{\Gamma, C_1 \to C_1, \cdots, C_n \to C_n \vdash \Delta}\ \text{cut?}$$

- What if we drop the context Γ and Δ?

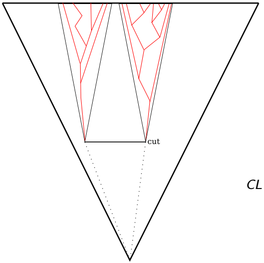
$$\dfrac{\vdots \qquad\qquad\qquad\qquad \vdots}{C_1 \to C_1, \cdots, C_n \to C_n \vdash}\ \text{cut?}$$

- Tautology in the antecedent $\Rightarrow$ Contradiction (**Unsatisfiable**)
- We have seen this before... Resolution
- Can we exploit this?

Characteristic Formula Extraction



LK-Proof with cuts



Paths to cut ancestors

$$CL(A \vdash A) \equiv \{A\}$$
$$CL(A \vdash A) \equiv \{\neg A\}$$
$$CL(A \vdash A) \equiv \{\neg A \vee A\}$$

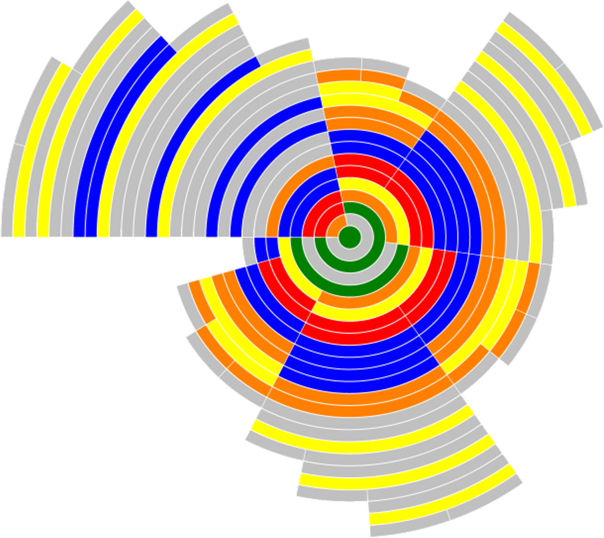
$$CL\left(\frac{\Delta \vdash \Pi}{\Delta' \vdash \Pi'}\ \rho \right) \equiv CL(\Delta \vdash \Pi)$$

$$CL\left(\frac{\Delta \vdash \Pi \qquad \Delta' \vdash \Pi'}{\Delta'' \vdash \Pi''}\ \rho \right) \equiv$$

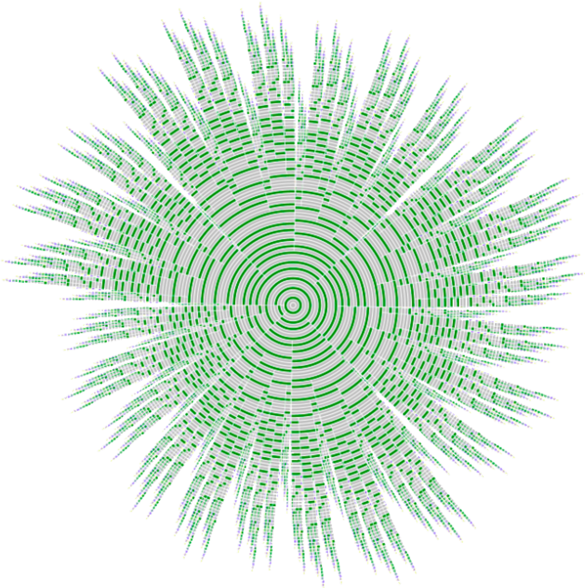
$$\begin{cases} CL(\Delta \vdash \Pi) \wedge CL(\Delta' \vdash \Pi') \\ CL(\Delta \vdash \Pi) \vee CL(\Delta' \vdash \Pi') \end{cases}$$

- The result is a formula which can be transformed into a clause set.
- Always unsatisfiable.

Refuting the Characteristic Formula



Refuting the Characteristic Formula



Refuting the Characteristic Formula

$$\dfrac{\vdots}{CL(\Pi) \vdash} \text{Res}$$

- Using **Herbrand's Theorem** we build the following **LK**-proof.

$$\dfrac{\vdots}{CL(\Pi)\sigma_1, \cdots, CL(\Pi)\sigma_n \vdash}$$

- This is still not an **LK**-proof of the original sequent.
- We need the following proof Projections

$$\dfrac{\vdots}{\Delta \vdash CL(\Pi), \Sigma}$$

Refuting the Characteristic Formula

Π	Projection
$$\dfrac{}{A \vdash A}\,\text{Ax}$$	$$\dfrac{}{\vdash A, \neg A}\,\text{Ax}$$
$$\dfrac{}{A \vdash A}\,\text{Ax}$$	$$\dfrac{}{\vdash A, A}\,\text{Ax}$$
$$\dfrac{}{A \vdash A}\,\text{Ax}$$	$$\dfrac{}{\vdash A \vee \neg A}\,\text{Ax}$$

Refuting the Characteristic Formula

Π	Projection
$\dfrac{\Pi, A \vdash \Sigma \qquad \Pi, B \vdash \Sigma}{\Pi, A \vee B \vdash \Sigma} \vee : l$	$\dfrac{\Pi \vdash A, \Sigma \qquad \Pi \vdash B, \Sigma}{\Pi \vdash A \vee B, \Sigma} \vee : l$
$\dfrac{\Pi, A \vdash \Sigma \qquad \Pi, B \vdash \Sigma}{\Pi, A \vee B \vdash \Sigma} \vee : l$	$\dfrac{\Pi A \vdash C_1, \Sigma \qquad \Pi B \vdash C_2, \Sigma}{\Pi \vdash C_1 \vee C_2, \Sigma} \vee : l$

- Unary rules do not change anything.
- Strong quantifiers on cut ancestors are dropped.

Propositional Cuts Only

$$
\dfrac{
\dfrac{
\dfrac{\vdots}{\Delta \vdash CL(\Pi)\sigma_1, \Sigma}
\qquad
\dfrac{\vdots}{CL(\Pi)\sigma_1, \cdots, CL(\Pi)\sigma_n \vdash}
}{\Delta, CL(\Pi)\sigma_2, \cdots, CL(\Pi)\sigma_n \vdash \Sigma}
\qquad
\dfrac{\vdots}{\Delta \vdash CL(\Pi)\sigma_2, \Sigma}
}{\Delta, \Delta, CL(\Pi)\sigma_3, \cdots, CL(\Pi)\sigma_n \vdash \Sigma, \Sigma}
$$

$$
\dfrac{\vdots}{\Delta, \cdots, \Delta \vdash \Sigma, \cdots, \Sigma}
$$

$$
\dfrac{\vdots}{\Delta \vdash \Sigma}
$$

Propositional Cuts Only

- Proof might not be cut-free
- **Observe:** propositional cuts do not obfuscate Herbrand instances.
- We could avoid this construction by projecting to the clauses of $CL(\Pi)$.
- Projection computation is more complex.
- Method works for other logics:
  - Intuitionistic Logic
  - Gödel Logic
  - Higher-order Logic