Resolution, Unification, and Subsumption:

Fundamental Concepts in Theorem Proving
(In memory of Alan Robinson)

Maria Paola Bonacina

Dipartimento di Informatica, Università degli Studi di Verona
Verona, Italy, EU

Invited talk at the 12th Summer School on Formal Techniques (SSFT)

SRI International and Menlo College, Atherton, California, USA, May 27, 2023

Three fundamental issues in theorem proving

- The ability of instantiating universally quantified variables
- The ability of removing redundant data
- The ability of avoiding generating intermediate inferences

Three answers

- The ability of instantiating universally quantified variables: resolution with unification (1963)
- The ability of removing redundant data: subsumption (1963)
- The ability of avoiding generating intermediate inferences: hyperresolution (1965)

Invented by J. Alan Robinson (1930–2016)
at the Argonne National Laboratory

J. Alan Robinson before Argonne

- BS, University of Cambridge, classics
- MS, University of Oregon, philosophy (adviser: Arthur Papp)
- PhD, University of Princeton, philosophy (adviser: Hilary Putnam) thesis on David Hume
- Job at DuPont, postdoc at U. Pittsburgh
- Alternated summer jobs at the Argonne National Laboratory and Stanford University in 1961-1966, working for Bill Miller, later Provost at Stanford (1971-79) and President and CEO of SRI International (1979-90)

J. Alan Robinson at Argonne

- Initial task: an implementation of the Davis-Putnam (DP) procedure (1960)
- Invented first-order resolution uniting propositional resolution (from the DP procedure) and unification (1962-1964)
- "A machine-oriented logic based on the resolution principle":
  - Unification, resolution, factoring, subsumption
  - Written in 1963: binary resolution and factoring
  - Published on JACM in 1965: resolution with factoring inside
  - In this talk: binary resolution and factoring
- "Automatic deduction with hyper-resolution" (1965)
- With Larry Wos et al. turned Argonne into the cradle of ATP

Larry Wos (1930–2020)

- BS, University of Chicago, mathematics
- MS, University of Chicago, mathematics
- PhD, University of Illinois at Urbana-Champaign, mathematics
- MCS Division, Argonne National Laboratory since 1957
- Leader of the theorem-proving research group
- Founder of CADE, JAR, AAR
- First Herbrand Award in 1992

Other three fundamental issues in theorem proving

- The ability of distinguishing assumptions and conjecture
- The ability of replacing equals by equals
- The ability of generating equations from equations

Three answers

- The ability of distinguishing assumptions and conjecture: the set of support strategy
- The ability of replacing equals by equals: demodulation
- The ability of generating equations from equations: paramodulation

Initiated by Larry Wos (with colleagues at Argonne)

J. Alan Robinson after Argonne

- Professor at Syracuse U.
- Founding Editor of the Journal of Logic Programming
- Milestone Award in Automatic Theorem Proving of the American Mathematical Society in 1985
- Herbrand Award in 1996
- Editor of the Handbook of Automated Reasoning (2001) (with Andrei Voronkov)

The theorem-proving problem

- A set H of formulas viewed as assumptions or hypotheses
- A formula φ viewed as conjecture
- Theorem-proving problem: $H \models^? \varphi$
- Equivalently: is $H \cup \{\neg\varphi\}$ unsatisfiable?
- Refutation: $H \cup \{\neg\varphi\} \vdash^? \bot$
- If success, then φ is a theorem of H, or $H \supset \varphi$ is a theorem
- Clausal form: $H \cup \{\neg\varphi\} \rightsquigarrow S$ set of clauses
- Form of the problem: $S \vdash^? \Box$ (the empty clause)

At the foundations of computer science

- Hilbert: Entscheidungsproblem (first-order validity)
- Completeness of first-order logic:
  - Gödel: $H \vdash \varphi$ iff $H \models \varphi$ (1930)
  - Henkin: $H \cup \{\neg\varphi\}$ unsatisfiable iff $H \cup \{\neg\varphi\}$ inconsistent (1947)
- Turing: Turing machine, first undecidable problem (halting), reduction of the Entscheidungsproblem to halting (1936)
- Herbrand: semidecidability of first-order validity (1930)

[Martin Davis. The Universal Computer–The Road from Leibniz to Turing]

Unification: work with substitutions

- A substitution is a function from variables to terms that is not identity on a finite set of variables
- $\sigma = \{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}$
- $\sigma = \{x \leftarrow a,\ y \leftarrow f(w),\ z \leftarrow w\}$
- Application: $h(x, y, z)\sigma = h(a, f(w), w)$

One-sided unification: Matching

- Given terms or atoms s and t
- $f(x, g(y))$ and $f(g(b), g(a))$
- Find matching substitution: σ s.t. $s\sigma = t$
  $\sigma = \{x \leftarrow g(b), y \leftarrow a\}$
- $s\sigma = t$: t is instance of s
  s is more general than t

Unification and most general unifier

- Given terms or atoms s and t
- $f(g(z), g(y))$ and $f(x, g(a))$
- Find substitution σ s.t. $s\sigma = t\sigma$:
  $\sigma = \{x \leftarrow g(z), y \leftarrow a\}$
- Most general unifier (mgu):
  σ is an mgu
  $\sigma' = \{x \leftarrow g(b), y \leftarrow a, z \leftarrow b\}$ is not

Resolution: canceling out literals of opposite sign

Propositional resolution:

$$\frac{P \vee Q \quad \neg P \vee R}{Q \vee R}$$

One of the inference rule of the Davis-Putnam procedure

Resolution for first-order logic (FOL): add unification

Binary resolution:

$$\frac{L_1 \vee C, \;\; L_2 \vee D}{(C \vee D)\sigma} \quad L_1\sigma = \neg L_2\sigma$$

- L_1 and L_2 have opposite sign
- σ is the most general unifier (mgu): least commitment
- The premises are called parents
- The generated and added clause is called resolvent

Example of binary resolution

$$\frac{P(g(z), g(y)) \vee \neg R(z, y) \quad \neg P(x, g(a)) \vee Q(x, g(x))}{\neg R(z, a) \vee Q(g(z), g(g(z)))}$$

where $\sigma = \{x \leftarrow g(z),\ y \leftarrow a\}$ is the mgu

$\sigma' = \{x \leftarrow g(b), y \leftarrow a, z \leftarrow b\}$ is not an mgu

Resolution for first-order logic

Binary resolution:

$$\frac{S \cup \{L_1 \vee C,\ L_2 \vee D\}}{S \cup \{L_1 \vee C,\ L_2 \vee D,\ (C \vee D)\sigma\}} \quad L_1\sigma = \neg L_2\sigma$$

- Resolution is an expansion inference rule because the resolvent is added to the set of clauses
- Expansion inference rules use unification
- If a parent is a unit clause (one literal): unit resolution

Why is factoring needed?

- For the refutational completeness of resolution
- Consider $P(x) \vee P(y)$ and $\neg P(z) \vee \neg P(w)$
- Binary resolution cannot generate the empty clause!
- Contradiction at the ground level: $P(t)$ and $\neg P(t)$
  x and y are instantiated with the same term t
  z and w are instantiated with the same term t
- Need an inference rule that merges unifiable literals in first-order clauses

How factoring solves the problem

$$\frac{P(x) \vee P(y)}{P(x)}$$

with mgu $\sigma = \{y \leftarrow x\}$

$$\frac{\neg P(z) \vee \neg P(w)}{\neg P(z)}$$

with mgu $\rho = \{w \leftarrow z\}$

Clauses $P(t)$ and $\neg P(t)$ that yield the contradiction at the ground level are instances of factors $P(x)$ and $\neg P(z)$

Factoring

$$\frac{S \cup \{L_1 \vee \ldots \vee L_k \vee C\}}{S \cup \{L_1 \vee \ldots \vee L_k \vee C,\ (L_1 \vee C)\sigma\}} \quad L_1\sigma = L_2\sigma = \ldots L_k\sigma$$

- The substitution σ is the mgu
- The generated and added clause is called factor
- Factoring is an expansion inference rule

Two major research problems

- Robinson's invention of resolution opened six decades of research in theorem proving
- Two major research problems:
  - How to generate fewer resolvents?
  - How to delete redundant resolvents?
- Two instances of the more general problems:
  - How to prevent the generation of redundant clauses
  - How to delete redundant clauses

  that are two sides of the same problem of redundancy

How to tame the growth of inferences

- Hyperresolution [Robinson 1965]
- Set of support strategy [Wos et al. 1965]
- Semantic resolution [Slagle 1967]
- Ordered resolution
  [Hsiang-Rusinowitch 1991] [Bachmair-Ganzinger 1994]
- Ordered resolution integrated with paramodulation/superposition
  [Hsiang-Rusinowitch 1991] [Bachmair-Ganzinger 1994]
- And with demodulation
  [Bachmair-Ganzinger 1994]

Motivation for the set of support strategy

- Resolution is too prolific
- Too many irrelevant inferences (do not appear in any proof)
- $H \cup \{\neg\varphi\} \leadsto S$: distinction between H and $\neg\varphi$ forgotten
- Larry Wos was interested in problems from mathematics
- In math problems $H \models^? \varphi$ the set H is known to be consistent (e.g., presentation of a theory)
- Then what is the point in expanding H?
  It won't give a contradiction!

The set of support strategy

- $H \leadsto A$: clausal form of H
- $\neg\varphi \leadsto SOS$: clausal form of $\neg\varphi$: goal clauses
- SOS is the input set of support
- If H is consistent, so is A: no point in expanding A
- A resolution step must have at least one parent from SOS
- All resolvents are added to SOS: only SOS grows (the factors of clauses in A are added to A upfront)
- A goal-sensitive strategy

The original given-clause algorithm for set of support

- Two lists sos and axioms initialized with *SOS* and *A*
- Loop until:
  - Either proof found: input unsatisfiable
  - Or sos empty: input satisfiable
- At every iteration: pick a given-clause *C* from sos
- Move *C* from sos to axioms
- Perform all expansion steps between *C* and clauses in axioms
- Add all newly generated clauses to sos
- No inference whose premises are both in *A*

(Bill McCune with OTTER)

The given-clause algorithm for expansion rules

- Two lists `to-be-selected` and `already-selected`
- Initialization for saturation:
  all input clauses in `to-be-selected`
  `already-selected` empty

(Bill McCune with OTTER and then many others)

Semantic resolution

A more general concept than set of support: semantic resolution

- Assume a fixed Herbrand interpretation $\mathcal{I}$ for semantic guidance
- Generate only resolvents that are false in $\mathcal{I}$

[Slagle 1967]

Semantic resolution as an inference rule

$$\frac{S \cup \{N,\ E_1, \ldots, E_k\}}{S \cup \{N,\ E_1, \ldots, E_k,\ R\}} \quad \mathcal{I} \not\models R$$

- Nucleus: $N = L_1 \vee \ldots \vee L_k \vee C$
- Satellites: $E_1 = M_1 \vee D_1, \ldots, E_k = M_k \vee D_k$
- Simultaneous mgu σ such that $L_i\sigma = \neg M_i\sigma$ for $i = 1 \ldots k$
- Semantic resolvent $R = (C \vee D_1 \vee \ldots \vee D_k)\sigma$
- Key requirement: $\mathcal{I} \not\models R$
- Hyperinference that embeds multiple resolution steps

Hyperresolution as instance of semantic resolution

- $\mathcal{I}$ contains all negative literals:
  - Positive hyperresolution
  - Resolve away all negative literals in the nucleus with positive satellites to generate a positive hyperresolvent
- $\mathcal{I}$ contains all positive literals:
  - Negative hyperresolution
  - Resolve away all positive literals in the nucleus with negative satellites to generate a negative hyperresolvent

[Robinson 1965]

Resolution with *SOS* as instance of semantic resolution

- $H \leadsto A$: clausal form of H
- $\neg\varphi \leadsto SOS$: clausal form of $\neg\varphi$: goal clauses
- Assume an interpretation $\mathcal{I}$ such that
  - $\mathcal{I} \models A$ and
  - $\mathcal{I} \not\models SOS$
- It generates only resolvents that are false in $\mathcal{I}$
- Not by hyperinferences, but by premise selection

Subsumption as in Robinson's paper (1963 version)

$$\frac{S \cup \{C,\ D\}}{S \cup \{C\}} \quad C\sigma \subseteq D \ \wedge\ |C| \leq |D|$$

- Idea: remove a clause implied by a more general one
- σ is a matching substitution
- Clauses as sets of literals
- $|C|$: number of literals in clause C
- $P(x) \vee P(y)$ does not subsume $P(z)$
- Prevents a clause from subsuming its factors

Subsumption with clauses as multisets

$$\frac{S \cup \{C,\ D\}}{S \cup \{C\}} \quad C\sigma \subseteq D$$

- Clauses as multisets of literals (ex.: $\{P(a), P(a), Q(b)\}$)
- $P(x) \vee P(y)$ does not subsume $P(z)$
- Prevents a clause from subsuming its factors
- If C is a unit clause: unit subsumption
- Subsumption is a contraction inference rule
- Contraction inference rules use matching

Subsumption with the subsumption ordering

$$\frac{S \cup \{C,\ D\}}{S \cup \{C\}} \quad C \stackrel{\cdot}{\leq} D$$

- $C \stackrel{\cdot}{\leq} D$ if $C\sigma \subseteq D$
- Clauses as multisets of literals
- However, the relations $\subseteq$, $\leq$, and $\stackrel{\cdot}{\leq}$ are not well-founded!
  [Kowalski 1970], [Loveland 1978]

Example of bad behavior I

- (1) $P(x, a)$
- (2) $P(f(x), y) \vee \neg P(x, y)$
- (3) $\neg Q(y) \vee \neg P(x, y)$
- (4) $Q(a)$

$SOS = \{(1)\ P(x, a)\}$

1. Resolve (1) $P(x, a)$ and (2) yielding (5) $P(f(x), a)$
2. Resolve (1) $P(x, a)$ and (3) yielding (6) $\neg Q(a)$

$SOS = \{(5)\ P(f(x), a),\ (6)\ \neg Q(a)\}$

Example of bad behavior II

3. Resolve (5) $P(f(x), a)$ and (2) yielding (7) $P(f(f(x)), a)$
4. Resolve (5) $P(f(x), a)$ and (3) yielding (8) $\neg Q(a)$

$SOS = \{(6)\ \neg Q(a),\ (7)\ P(f(f(x)), a),\ (8)\ \neg Q(a)\}$

5. (8) subsumes (6)
6. Resolve (7) $P(f(f(x)), a)$ and (2) yielding (9) $P(f(f(f(x))), a)$
7. Resolve (7) $P(f(f(x)), a)$ and (3) yielding (10) $\neg Q(a)$

$SOS = \{(8)\ \neg Q(a),\ (9)\ P(f(f(f(x))), a),\ (10)\ \neg Q(a)\}$

8. (10) subsumes (8)
9. Infinite loop: subsumption prevents ever resolving $\neg Q(a)$ and $Q(a)$

An operational solution

- Distinguish between forward subsumption and backward subsumption
- Forward subsumption: apply existing clauses to try to subsume every newly generated clause
- Backward subsumption: apply a newly generated clause to try to subsume pre-existing clauses
- Apply forward subsumption before backward subsumption

[Kowalski 1970]

Subsumption in the given clause algorithm I

- Forward subsumption: apply clauses in
  `already-selected` $\bigcup$ `to-be-selected`
  to try to subsume every newly generated clause
  prior to its addition to `to-be-selected`
- Backward subsumption: apply every newly generated clauses, just added to `to-be-selected`, to try to subsume clauses in
  `already-selected` $\bigcup$ `to-be-selected`

[Bill McCune, OTTER prover]

Subsumption in the given clause algorithm II

- Ignore `to-be-selected` for the purpose of contraction
- Forward subsumption: apply clauses in `already-selected` to try to subsume the newly selected given clause, prior to its addition to `already-selected`
- Backward subsumption: apply the given clause just added to `already-selected` to try to subsume other clauses in `already-selected`
- Delete orphans (descendants of subsumed clauses in `already-selected`)

[Denzinger-Kronenburg-Schulz, DISCOUNT prover], [Schulz, E prover]

Subsumption

$$\frac{S \cup \{C,\ D\}}{S \cup \{C\}} \quad (C, n) \lessdot_2 (D, m)$$

- Every generated clause gets a natural number as its index
- $C \lessdot D$ if $C\sigma \subseteq D$
- $<$ ordering on $\mathbb{N}$ (the natural numbers)
- $\lessdot_2$: lexicographic combination of $\lessdot$ and $<$ applied to pairs (C, n) where n is the index of C
- If $C\sigma \subseteq D$ and $D\sigma \subseteq C$: the oldest is retained

Towards ordered resolution: orderings

Reduction ordering:

- Well-founded
- Stable: $t \succ u$ implies $t\sigma \succ u\sigma$ for all substitutions σ
- Monotonic: $t \succ u$ implies $c[t] \succ c[u]$ for all contexts c
  - KBO: Knuth-Bendix Orderings [Knuth-Bendix 1970]
  - RPO: Recursive Path Orderings [Dershowitz 1982]
  - LPO: Lexicographic (recursive) Path Orderings [Kamin-Lévy 1980]
- In general these orderings are partial, not total!

Complete simplification ordering

- Subterm property: $c[t] \succeq t$
- Stable: $t \succ u$ implies $t\sigma \succ u\sigma$ for all substitutions σ
- Monotonic: $t \succ u$ implies $c[t] \succ c[u]$ for all contexts c
- These three properties imply well-founded
- Total on ground terms
  - Knuth-Bendix orderings
  - Recursive path orderings (not all)
  - Lexicographic path orderings

Multiset extension of an ordering

- Multisets, e.g., $\{P(a), P(a), Q(b)\}$, $\{5, 4, 4, 4, 3, 1, 1\}$
- From $\succ$ to $\succ_{mul}$:
  - $M \succ_{mul} \emptyset$ if $M \neq \emptyset$
  - $M \cup \{a\} \succ_{mul} N \cup \{a\}$ if $M \succ_{mul} N$
  - $M \cup \{a\} \succ_{mul} N \cup \{b\}$ if $a \succ b$ and $M \cup \{a\} \succ_{mul} N$
- $\{5\} \succ_{mul} \{4, 4, 4, 3, 1, 1\}$
- If $\succ$ is well-founded then $\succ_{mul}$ is well-founded

[Nachum Dershowitz & Zohar Manna 1979]

From ordering terms to ordering literals

- Complete or completable reduction ordering (all KBO's, RPO's, LPO's)
- Read a positive literal L as $L \simeq \top$ and $\neg L$ as $L \not\simeq \top$ where $\top$ is a new symbol such that $t \succ \top$ for all terms t
- Equality as the only predicate symbol
- Treat $p \simeq q$ as the multiset $\{p, q\}$ and $p \not\simeq q$ as the multiset $\{p, p, q, q\}$
- Apply the multiset extension of the ordering on terms

[Leo Bachmair & Harald Ganzinger 1994]

Maximal literals

- Clauses as multisets of literals
- Literal L is maximal in clause C if
  $\neg(\exists M \in C.\ M \succ L)$ or equivalently $\forall M \in C.\ L \not\prec M$
  The other literals can only be smaller, equal, or uncomparable
- Literal L is strictly maximal in clause C if
  $\neg(\exists M \in C.\ M \succeq L)$ or equivalently $\forall M \in C.\ L \not\preceq M$
  The other literals can only be smaller or uncomparable

Ordered Resolution

$$\frac{S \cup \{L_1 \vee C,\ L_2 \vee D\}}{S \cup \{L_1 \vee C,\ L_2 \vee D,\ (C \vee D)\sigma\}}$$

- $L_1\sigma = \neg L_2\sigma$ (σ mgu)
- $\forall M \in C.\ L_1\sigma \not\preceq M\sigma$ (strictly maximal)
- $\forall M \in D.\ L_2\sigma \not\preceq M\sigma$ (strictly maximal)

[Jieh Hsiang & Michaël Rusinowitch 1991]

Example

$$\frac{P(g(z), g(y)) \vee \neg R(z, y), \;\; \neg P(x, g(a)) \vee Q(x, g(x))}{\neg R(z, a) \vee Q(g(z), g(g(z)))}$$

- $\sigma = \{x \leftarrow g(z), \; y \leftarrow a\}$
- Check that $P(g(z), g(a)) \not\preceq \neg R(z, a)$
- Check that $P(g(z), g(a)) \not\preceq Q(g(z), g(g(z)))$
- Allowed with precedence $P > R > Q > g$
- Not allowed with precedence $Q > R > P > g > a$

Ordered Factoring

$$\frac{S \cup \{L_1 \vee \ldots \vee L_k \vee C\}}{S \cup \{L_1 \vee \ldots \vee L_k \vee C,\ (L_1 \vee C)\sigma\}}$$

- $L_1\sigma = L_2\sigma = \ldots L_k\sigma$ (σ mgu)
- $\forall M \in C.\ L_1\sigma \not\preceq M\sigma$ (strictly maximal)

[Jieh Hsiang & Michaël Rusinowitch 1991]

And equality?

The equality axioms in clausal form:

$$x \simeq x \qquad (\mathit{Reflexivity})$$

$$x \not\simeq y \vee y \simeq x \qquad (\mathit{Symmetry})$$

$$x \not\simeq y \vee y \not\simeq z \vee x \simeq z \qquad (\mathit{Transitivity})$$

$$\bigvee_{i=1}^{n} x_i \not\simeq y_i \vee f(\bar{x}) \simeq f(\bar{y}) \qquad (\mathit{Function\ Substitutivity})$$

$$\bigvee_{i=1}^{n} x_i \not\simeq y_i \vee \neg P(\bar{x}) \vee P(\bar{y}) \qquad (\mathit{Predicate\ Substitutivity})$$

Added to the input for resolution: not practical!

The first paramodulation inference rule

$$\frac{S \cup \{l \simeq r \vee C,\ M[t] \vee D\}}{S \cup \{l \simeq r \vee C,\ M[t] \vee D,\ (C \vee M[r] \vee D)\sigma\}} \quad l\sigma = t\sigma$$

- $\simeq$ is symmetric and σ is the mgu of l and t
- C and D are disjunctions of literals
- $l \simeq r \vee C$ is the para-from clause
- $l \simeq r$ is the para-from literal
- $M[t] \vee D$ is the para-into clause
- $M[t]$ is the para-into literal
- $(C \vee M[r] \vee D)\sigma$ is called paramodulant

[Larry Wos - George Robinson 1969]

The challenge of the Wos–Robinson conjecture

- Wos–Robinson conjecture:
  paramodulation is refutationally complete
  without paramodulating into variables and
  without functionally reflexive axioms
  Functionally reflexive axioms: $f(\bar{x}) \simeq f(\bar{x})$ for all function symbols f

Unfailing or ordered completion

- $E \models^? \forall \bar{x}.s \simeq t$
- Negating $\forall \bar{x}.s \simeq t$ yields $\exists \bar{x}.s \not\simeq t$ and hence $\hat{s} \not\simeq \hat{t}$ where $\hat{s}$ is s with all vars replaced by Skolem constants
- Refutationally: $E \cup \{\hat{s} \not\simeq \hat{t}\} \vdash^? \Box$
- Apply completion to E and reduce $\hat{s}$ and $\hat{t}$ whenever possible
- Refutation found if $\hat{s} \xrightarrow{*} u$ and $\hat{t} \xrightarrow{*} u$ so that $u \not\simeq u$ contradicts $x \simeq x$
- State of the derivation: $(E; \hat{s} \not\simeq \hat{t})$
  E: set of equations

[Hsiang-Rusinowitch 1987] [Bachmair-Dershowitz-Plaisted 1989]

Superposition of equations

$$\frac{E \cup \{l \simeq r,\ p[t] \simeq q\}}{E \cup \{l \simeq r,\ p[t] \simeq q,\ p[r]\sigma \simeq q\sigma\}} \quad t \notin X,\ l\sigma = t\sigma$$

- $l\sigma \not\preceq r\sigma$
- $p[t]\sigma \not\preceq q\sigma$
- $l \simeq r$ and $p[t] \simeq q$ superpose only if their instances by σ are either orientable ($l\sigma \succ r\sigma$) or uncomparable
- Equivalently: only if $l\sigma$ is strictly maximal in $\{l\sigma, r\sigma\}$ and $p[t]\sigma$ is strictly maximal in $\{p[t]\sigma,\ q\sigma\}$

[Hsiang-Rusinowitch 1987] [Bachmair-Dershowitz-Plaisted 1989]

Example

$$\frac{f(z,e) \simeq z \quad f(l(x,y),y) \simeq x}{l(x,e) \simeq x}$$

- $f(z,e)\sigma = f(l(x,y),y)\sigma$
- $\sigma = \{z \leftarrow l(x,e),\ y \leftarrow e\}$ most general unifier
- $f(l(x,e),e) \succ l(x,e)$ (by the subterm property)
- $f(l(x,e),e) \succ x$ (by the subterm property)
- Superposing two equations yields a peak:
  $l(x,e) \leftarrow f(l(x,e),e) \rightarrow x$

A further challenge

How to obtain an inference system for FOL+= that

- Avoids paramodulating or superposing into variables
- Is restricted by the ordering
- Is refutationally complete also in the presence of contraction
- Reduces to completion for an input of the form $E \cup \{\hat{s} \not\simeq \hat{t}\}$

Toward ordered paramodulation / superposition

- Para-from clause: $l \simeq r \vee C$
- Para-into clause:
  - $M[t] \vee D$
  - $p[t] \simeq q \vee D$
  - $p[t] \not\simeq q \vee D$
- $l\sigma = t\sigma$ (mgu σ)
- The subterm t is not a variable ($t \notin X$)

Four ordering-based conditions

(i) Para-from literal strictly maximal: $\forall Q \in C.\ (l \simeq r)\sigma \not\preceq Q\sigma$

(ii) Left-hand side of para-from literal strictly maximal: $l\sigma \not\preceq r\sigma$

(iii.a) Para-into literal strictly maximal: $\forall Q \in D.\ M[t]\sigma \not\preceq Q\sigma$
$\forall Q \in D.\ (p[t] \simeq q)\sigma \not\preceq Q\sigma$

(iii.b) Or maximal if it is a negated equation:
$\forall Q \in D.\ (p[t] \not\simeq q)\sigma \not\prec Q\sigma$

(iv) Left-hand side of positive equational para-into literal strictly maximal: $p[t]\sigma \not\preceq q\sigma$

Ordered paramodulation

$$\frac{S \cup \{l \simeq r \vee C,\ M[t] \vee D\}}{S \cup \{l \simeq r \vee C,\ M[t] \vee D,\ (C \vee M[r] \vee D)\sigma\}} \quad (i)\ (ii)\ (iii.a)$$

The refutational completeness of the Ordered Literal Inference System with ordered resolution, ordered factoring, and ordered paramodulation settled the Wos–Robinson conjecture

[Jieh Hsiang & Michaël Rusinowitch 1991]

The superposition calculus $\mathcal{SP}$

Affords all four ordering-based conditions:

$$\frac{S \cup \{l \simeq r \vee C,\ p[t] \simeq q \vee D\}}{S \cup \{l \simeq r \vee C,\ p[t] \simeq q \vee D,\ (C \vee p[r] \simeq q \vee D)\sigma\}}$$

with (i), (ii), (iii.a), and (iv)

$$\frac{S \cup \{l \simeq r \vee C,\ p[t] \not\simeq q \vee D\}}{S \cup \{l \simeq r \vee C,\ p[t] \not\simeq q \vee D,\ (C \vee p[r] \not\simeq q \vee D)\sigma\}}$$

with (i), (ii), (iii.b), and (iv)
and solved also the problem of generalizing completion to FOL+=
[Leo Bachmair & Harald Ganzinger 1994]

Replacing equals by equals: demodulation

The first demodulation inference rule:

$$\frac{S \cup \{l \simeq r,\ C[l\sigma]\}}{S \cup \{l \simeq r,\ C[r\sigma]\}} \quad \| C[l\sigma] \| \ > \ \| C[r\sigma] \|$$

- $l \simeq r$ is called demodulant or demodulator
- σ is a matching substitution
- $\| C \|$ is the number of symbols in C
- Decreasing the number of symbols is well-founded because the ordering on the natural numbers is well-founded

[Wos et al. 1967]

Problems opened by Larry Wos' demodulation

- What if the number of symbols does not change?
  Ex.: $x + y \simeq y + x$
- What if we wanted to increase the number of symbols?
  Ex.: $x * (y + z) \simeq x * y + x * z$
- Does resolution remain refutationally complete if we add demodulation?

Demodulation in ordered completion

Simplification:

$$\frac{(E \cup \{l \simeq r\}; \hat{s}[l\sigma] \not\simeq \hat{t})}{(E \cup \{l \simeq r\}; \hat{s}[r\sigma] \not\simeq \hat{t})} \quad l\sigma \succ r\sigma$$

$$\frac{(E \cup \{p[l\sigma] \simeq q,\ l \simeq r\}; \hat{s} \not\simeq \hat{t})}{(E \cup \{p[r\sigma] \simeq q,\ l \simeq r\}; \hat{s} \not\simeq \hat{t})}$$

- $l\sigma \succ r\sigma$
- $p[l\sigma] \triangleright l \ \vee \ q \succ p[r\sigma]$

What is $\triangleright$?

The encompassment ordering

- Encompassment: $t \trianglerighteq s$ if $t = c[s\vartheta]$
- ϑ is a substitution
- Strict: either c is not empty or ϑ is not a variable renaming (A variable renaming is a substitution that maps variables to variables and is injective)

The side condition for simplification of equations

- $p[l\sigma] \triangleright l \ \vee \ q \succ p[r\sigma]$
- It lets $l \simeq r$ simplify $p[l\sigma] \simeq q$ when $p[l\sigma]$ is a variant of l provided that $q \succ p[r\sigma]$
- Apply $f(e,y) \simeq y$ to simplify $f(e,x) \simeq h(x)$?
  Yes because $h(x) \succ x$
- Apply $f(e,y) \simeq y$ to simplify $f(e,x) \simeq x$?
  No because $x \not\succ y$
- Apply $f(e,x) \simeq h(x)$ to simplify $f(e,y) \simeq y$?
  No because $y \not\succ h(y)$

Example of simplification

1. $f(x) \simeq g(x)$
2. $g(h(y)) \simeq k(y)$
3. $f(h(b)) \not\simeq k(b)$ (target theorem)

- Precedence: $f > g > h > k > b$
- (1) simplifies the target to $g(h(b)) \not\simeq k(b)$
  with matching substitution $\sigma = \{x \leftarrow h(b)\}$
  since $f(h(b)) \succ g(h(b))$
- (2) simplifies $g(h(b)) \not\simeq k(b)$ to $k(b) \not\simeq k(b)$
  with matching substitution $\vartheta = \{y \leftarrow b\}$
  since $g(h(b)) \succ k(b)$

A simplification inference rule for clauses

$$\frac{S \cup \{C[l\sigma],\ l \simeq r\}}{S \cup \{C[r\sigma],\ l \simeq r\}} \quad l\sigma \succ r\sigma, \qquad C[l\sigma] \succ (l\sigma \simeq r\sigma)$$

In the superposition calculus $\mathcal{SP}$

[Leo Bachmair & Harald Ganzinger 1994]

The above example revisited

1. $f(x) \simeq g(x)$
2. $g(h(y)) \simeq k(y)$
3. $f(h(b)) \not\simeq k(b)$ (target theorem)

- Precedence: $f > g > h > k > b$
- (1) simplifies the target to $g(h(b)) \not\simeq k(b)$
  with matching substitution $\sigma = \{x \leftarrow h(b)\}$
  since $\{f(h(b)), f(h(b)), k(b), k(b)\} \succ_{mul} \{f(h(b)), g(h(b))\}$
- (2) simplifies $g(h(b)) \not\simeq k(b)$ to $k(b) \not\simeq k(b)$
  with matching substitution $\vartheta = \{y \leftarrow b\}$
  since $\{g(h(b)), g(h(b)), k(b), k(b)\} \succ_{mul} \{g(h(b)), k(b)\}$

Another example

1. $f(x) \simeq b$
2. $f(b) \simeq c$

- Precedence: $b \succ c$
- Simplification of completion allows (1) to simplify (2) to $b \simeq c$ with matching substitution $\sigma = \{x \leftarrow b\}$ because $f(b) \succ b$ and $f(b) \trianglerighteq f(x)$
- But $\{f(b), c\} \succ_{mul} \{f(b), b\}$ does not hold
- Simplification of $\mathcal{SP}$ does not apply
- Encompassment demodulation for $\mathcal{SP}$

  [André Duarte and Konstantin Korovin at IJCAR 2022]

  [André Duarte's PhD thesis 2023]

References

- Maria Paola Bonacina. Set of support, demodulation, paramodulation: a historical perspective.
  Journal of Automated Reasoning 66(4):463–497, 2022
  DOI = 10.1007/s10817-022-09628-0.
- Michael Beeson, Maria Paola Bonacina, Michael Kinyon, and Geoff Sutcliffe. Larry Wos – Visions of automated reasoning.
  Journal of Automated Reasoning 66(4):439–461, 2022
  DOI = 10.1007/s10817-022-09620-8.

Thank you!