Inference Algorithms

N. Shankar

Computer Science Laboratory
SRI International
Menlo Park, CA

July 3, 2017

Computation and Inference

- Deductive inference, the process of making logically valid steps of reasoning, is central to mathematics.
- There are two ways of solving mathematical problems:
  1. Reduce the problem to another previously solve problem.
  2. Perform a series of inference steps to reduce the problem to a equivalent (possibly unsolvable) solved form.
- For the latter kind of solution, there is a systematic way of thing about these algorithms in the form of inference systems
- The key slogan is

$$\textit{Algorithm} = \textit{Inference} + \textit{Strategy} + \textit{Indexing}$$

  - Inference rules circumscribe the inference rules that are applicable
  - The strategy indicates how one select which inference rule to apply
  - Indexing uses data structures to efficiently execute the strategy.

Preliminaries

- We assume a first-order language of terms and formulas (with equality) over a signature Σ, e.g., $\Sigma = \{0, 1, -, +, <\}$
- A (Σ-)*term* is either a variable or an n-ary function symbol f in Σ applied to n Σ-terms $a_1, \ldots, a_n$ as $f(a_1, \ldots, a_n)$, e.g., $x + (y - 1)$.
- A (Σ-)*literal* l is either an atom $p(t_1, \ldots, t_n)$ or its negation, where p is either equality (hence, of arity 2) or an n-ary predicate symbol in Σ, e.g., $x + 0 = x$, $0 < x$.
- A *constraint* C is a set of literals, and is interpreted as their conjunction, e.g., $\neg p(x, y), p(x, x), x = y$.
- A *clause* Γ is a disjunction of literals, e.g.. $x \not< y \vee y \not< z \vee x < z$. K ranges over sets of clauses.

Preliminaries

- $vars(t), vars(l), vars(A), vars(C), vars(K)$ return the sets of free variables in term t, literal l, formula A, constraint C, or clause K.
- A Σ-*structure* M interprets the symbols in Σ over the domain $|M|$. We extend M to also interpret the free variables so that $M(x)$ is the assignment for x.
- $M \models A$ if $M[\![A]\!] = \top$.
- A is satisfiable iff $M \models A$ for some M, e.g., $x = y$ is satisfiable, but $x \neq y$ is not.
- A and B are equisatisfiable if A is satisfiable iff B is satisfiable.
- Equivalence implies equisatisfiability, but not vice-versa.

Inference Problems

- Typical classes of problems are
  1. *Word problem* (WP): $\mathcal{T} \models p$, for Σ-atom p.
  2. *Uniform word problem* (UWP): $\mathcal{T} \models \bigwedge \Gamma \rightarrow p$ for finite sets of Σ-atoms Γ and Σ-atom p.
  3. *Clausal validity problem* (CVP): $\mathcal{T} \models \bigvee K$ for a Σ-clause K.
  4. *Quantifier-free validity* (QFV): $\mathcal{T} \models \bigvee a$ for quantifier-free formula A.
  5. *Satisfiability*($\mathcal{T}$-SAT): $\mathcal{T} \models A$ for any formula A.
  6. *Quantifier Elimination* (QE): Reduce A to a quantifier-free formula $\widehat{A}$ such that $\mathcal{T} \models A \iff \widehat{A}$.
  7. *Interpolation* (INT): If Γ, Δ is $\mathcal{T}$-unsatisfiable, find a formula A in $\mathcal{L}(\Gamma) \cap \mathcal{L}(\Delta)$ such that $\Gamma \models_{\mathcal{T}} A$ and A, Δ is $\mathcal{T}$-unsatisfiable.
- Clausal validity problems are often presented as unsatisfiability problems for sets of literals since $\mathcal{T} \models l_1 \vee \ldots \vee l_n$ if, and only if the set of literals $\bar{l}_1, \ldots, \bar{l}_n$ is $\mathcal{T}$-unsatisfiable.

What is an Inference Algorithm?

- An *Σ-inference structure* $\langle \Psi, \vdash, \Lambda, \mathcal{M} \rangle$ consists of
  - Ψ, a set of *logical states*
  - $\vdash$, the *reduction relation* between states
  - Λ, a map from states to Σ-formulas
  - $\mathcal{M}$, which extracts models from canonical states
- An *inference system* is an inference structure that is
  - *Conservative*: If $\psi \vdash \psi'$, then $\Lambda(\psi)$ and $\Lambda(\psi')$ are equisatisfiable.
  - *Progressive*: $\vdash$ is well-founded.
  - *Canonizing*: If $\psi \nvdash \psi'$ for any ψ', then either ψ is $\bot$ (i.e., unsatisfiable) or ψ is in a canonical form so that $\mathcal{M}(\psi)$ is a model for $\Lambda(\psi)$.
- It is *strongly conservative* if whenever $\psi \vdash \psi'$, then ψ and ψ' are equisatisfiable and any model of ψ' is also a model of ψ.
- We focus here on basic inference systems, but there are interesting variants.

What is an Inference Algorithm?

- An *inference algorithm* is an inference system where the reduction relation is presented as a collection of effective *inference rules* that transform an inference state ψ to an inference state ψ' such that $\psi \vdash \psi'$. **Example:** Ordered resolution is an algorithm for CNF satisfiability.
- Input K is a set of ordered clauses where the literals appear in decreasing order w.r.t. some order e.g., $q \prec \neg q \prec p \prec \neg p$.
- Tautologies, i.e., clauses containing both p and $\neg p$, are deleted from initial input.

Res	$\dfrac{K, p \vee \Gamma_1, \neg p \vee \Gamma_2}{K, p \vee \Gamma_1, \neg p \vee \Gamma_2, \Gamma_1 \vee \Gamma_2}$ $\quad \Gamma_1 \vee \Gamma_2 \notin K$ $\quad \Gamma_1 \vee \Gamma_2$ is not tautological
Contrad	$\dfrac{K, p, \neg p}{\bot}$

- A set of clauses is canonical if it is closed under applications of **Res** and the **Contrad** rule is inapplicable.

Resolution: Example

$$\dfrac{\dfrac{\dfrac{(K_0 =)\ \neg p \vee \neg q \vee r, \quad \neg p \vee q, \quad p \vee r, \quad \neg r}{(K_1 =)\ \neg q \vee r, \quad K_0}\ \textbf{Res}}{(K_2 =)\ q \vee r, \quad K_1}\ \textbf{Res}}{(K_3 =)\ r, \quad K_2}\ \textbf{Res}$$

$$\dfrac{(K_3 =)\ r, \quad K_2}{\bot}\ \textbf{Contrad}$$

- Drop the clause $\neg r$, and we reach an irreducible state from which a truth assignment $\{r \mapsto \top, q \mapsto \bot, p \mapsto \bot\}$ can be constructed.

Resolution as an Inference Algorithm

- The resolution inference system is *strongly conservative*: $\Gamma_1 \vee \Gamma_2$ is satisfiable if $p \vee \Gamma_1$ and $\neg p \vee \Gamma_2$ are.
- It is *progressive*: Bounded number of new clauses in the input variables.
- It is *canonizing*: Build a model M by assigning to atoms p_1 to p_n within a series of partial assignments $M_0, \ldots, M_n$:
  - M_0 is the empty truth assignment.
  - $M_{i+1} = M_i[p_{i+1} \mapsto v]$, where $v = \top$ iff there is some clause $p_{i+1} \vee \Gamma$ in the irreducible state K such that $M_i \models \neg\Gamma$.
- If $M_i \models \neg\Gamma$, then for any clause $\neg p_i \vee \Delta$, $M_i \models \Delta$ since $\Gamma \vee \Delta \in K$.
- **Invariant:** $M_i \models \Gamma$ for all clauses Γ in K in the atoms $p_1, \ldots, p_i$.
- Unordered resolution is also conservative, progressive, and canonizing, but it does not have the same set of canonical states.

Maintaining Equivalence with Union-Find

The logical state is a triple $\langle G, F \rangle$ with the input equalities and disequalities G and the find structure F which is a set of oriented equalities, i.e., orient $y = x$ as $x = y$ if $y \prec x$.

Delete	$\dfrac{x = y, G; F}{G; F}$ if $F(x) \equiv F(y)$
Merge	$\dfrac{x = y, G; F}{G; F' \circ F}$ if $F(x) \not\equiv F(y)$ $F' = \{orient(F(x) = F(y))\}$
Contrad	$\dfrac{x \neq y, G; F}{\bot}$ if $F(x) = F(y)$

- The above inference system is (strongly) conservative, progressive, and canonizing.
- Example: $x = y, x = z, u = v; \emptyset$ reduces to $\emptyset; x = z, y = z, u = v$.

Normal Forms by Inference

An inference system for building conjunctive normal form (CNF) converts a formula built from negation, conjunction, and disjunction, into a set of clauses, as follows.

Conjunction	$\dfrac{A[l_1 \wedge l_2]; K}{A[x]; K \cup \{\neg x \vee l_1, \neg x \vee l_2, x \vee \overline{l_1} \vee \overline{l_2}\}}$ x is fresh
Disjunction	$\dfrac{A[l_1 \vee l_2]; K}{A[x]; K \cup \{x \vee \overline{l_1}, x \vee \overline{l_2}, \neg x \vee l_1 \vee l_2\}}$ x is fresh
Literal	$\dfrac{l; K}{; K}$

Converting $\neg(p \vee (\neg p \wedge q))$ to 3-CNF yields

$$\neg s, s \vee \neg p, s \vee \neg r, \neg s \vee p \vee r, \neg r \vee \neg p, \neg r \vee q, r \vee p \vee \neg q$$

CDCL Informally

- Goal: Does a given set of clauses K, e.g., $p \vee q, \neg p, \neg q$, have a satisfying assignment?
- If M is a total assignment such that $M \models \Gamma$ for each $\Gamma \in K$, then $M \models K$.
- If M is a partial assignment at level h, then *implied literals* l such that $l \vee \Gamma \in K \cup C$ and $M \models \neg\Gamma$ are *propagated* to M at level h as $l[l \vee \Gamma]$.
- The clause $l \vee \Gamma$ is the explanation for the assignment to l.
- If M detects a conflict at level h, i.e., a clause $\Gamma \in K \cup C$ such that $M \models \neg\Gamma$, the conflict is *analyzed*.
- Conflict analysis resolves the implied literals at level h in the conflict against their explanations until there is a unique literal in the clause at level h.
- If M cannot be extended at level h and no conflict is detected, then an unassigned literal l is *selected* and assigned at level $h + 1$ where the search is continued.

Conflict-Driven Clause Learning (CDCL) SAT

Name	Rule	Condition
Propagate	$\dfrac{h, \langle M \rangle, K, C}{h, \langle M, l[\Gamma] \rangle, K, C}$	$\Gamma \equiv l \vee \Gamma' \in K \cup C$ $M \models \neg\Gamma'$
Select	$\dfrac{h, \langle M \rangle, K, C}{h+1, \langle M; l[] \rangle, K, C}$	$M \not\models l$ $M \not\models \neg l$
Conflict	$\dfrac{0, \langle M \rangle, K, C}{\bot}$	$M \models \neg\Gamma$ for some $\Gamma \in K \cup C$
Backjump	$\dfrac{h+1, \langle M \rangle, K, C}{h', \langle M_{\leq h'}, l[\Gamma'] \rangle, K, C \cup \{\Gamma'\}}$	$M \models \neg\Gamma$ for some $\Gamma \in K \cup C$ $\langle h', \Gamma' \rangle$ $= \mathit{analyze}(\psi)(\Gamma)$ for $\psi = h, \langle M \rangle, K, C$

$\mathit{analyze}(h, \langle M \rangle, K, C)(l \vee \Gamma) = \langle \mathit{level}(\Gamma), l \vee \Gamma \rangle$, if $\mathit{level}(\Gamma) < h$.
$\mathit{analyze}(h, \langle M \rangle, K, C)(l \vee \Gamma) = \mathit{analyze}(h, \langle M \rangle, K, C)(\Gamma' \vee \Gamma)$, for $\bar{l}[\Gamma'] \in M$, otherwise.

CDCL Example

- Let K be
  $\{p \vee q, \neg p \vee q, p \vee \neg q, s \vee \neg p \vee q, \neg s \vee p \vee \neg q, \neg p \vee r, \neg q \vee \neg r\}$.
-

step	h	M	K	C	Γ
select s	1	$; s$	K	$\emptyset$	_
select r	2	$; s; r$	K	$\emptyset$	_
propagate	2	$; s; r, \neg q[\neg q \vee \neg r]$	K	$\emptyset$	_
propagate	2	$; s; r, \neg q, p[p \vee q]$	K	$\emptyset$	_
conflict	2	$; s; r, \neg q, p$	K	$\emptyset$	$\neg p \vee q$

CDCL Example (contd.)

step	h	M	K	C	Γ
conflict	2	$; s; r, \neg q, p$	K	$\emptyset$	$\neg p \vee q$
backjump	0	$\emptyset$	K	q	_
propagate	0	$q[q]$	K	q	_
propagate	0	$q, p[p \vee \neg q]$	K	q	_
propagate	0	$q, p, r[\neg p \vee r]$	K	q	_
conflict	0	q, p, r	K	q	$\neg q \vee \neg r$

CDCL Correctness

- **Progress:** Each backjump step adds a new assignment at the level h' so that $\sum_{i=0}^{h} |M_i| * (N+1)^{(N-h)}$ increases toward the bound $(N+1)^{(N+1)}$ for $N = |vars(K)|$. In the example, $N = 4$, the backjump step goes from a value 1300 in base 5 to the value 10000 which is closer to the bound 40000.
- **Conservation:** In each transition from $\langle M, K, C \rangle$ to $\langle M', K', C' \rangle$ (or $\bot$), the clause sets $M_0 \cup K \cup C$ and $M_0 \cup K' \cup C'$ are equisatisfiable.
- **Canonicity:** In an irreducible non-$\bot$ state, M is total assignment and there is no conflict, so for each clause Γ in $K \cup C$, $M \models \Gamma$.

Implementation Notes

- The input clauses can be preprocessed by resolution, e.g., to eliminate a variable, and subsumption to discard a clause when a subclause is already available.
- The *selection* heuristic typically picks the the variable that has been active in recent conflicts — the VSIDS heuristic of Chaff.
- It also helps to retain the previous assignment for the variable. [RSAT]
- Propagation uses *two-watched literals* per clause, so that a clause is visited only when a watched literal is falsified. [GRASP]
- Learned clauses can be *deleted* when they are not used in the partial assignment and not recently active in conflicts.
- Frequent *restarts* are good for learning useful short clauses in order to better direct the search.
- All level 0 inferences can be applied permanently.
- There are a number of other preprocessing and inprocessing rules that we are omitting.

Superposition

- The atoms are now of the form $x = y$ with an ordering $\succ$ on variables.
- Equalities are kept ordered so that if $x = y$, $x \succ y$.
- Ordering is lifted to literals so that $x = y \succ x' = y'$ ($x \neq y \succ x' = y'$) iff $x \succ x'$ or $x \equiv x'$ and $y \succ y'$, and $x \neq y \succ x = y'$ for any y, y'.
- Literals of the form $x \neq x$ are deleted from input clauses.
- Clauses are maintained in decreasing order, and tautologies containing l and $\bar{l}$ or $x = x$ are deleted from the input.
- E.g., $y = z, x = y \vee x = z, x \neq y \vee x \neq z$.

Superposition Inference Rules

Assume that $x \succ y, y' \succ z$, all equalities are oriented $x = y$, and all clauses are sorted in decreasing order.

Right	$\dfrac{(\Gamma' =)\ x = y \vee L, x = z \vee K, \Gamma}{y = z \vee L \vee K, \Gamma'}$
Left	$\dfrac{(\Gamma' =)\ x = y \vee L, x \neq y' \vee K, \Gamma}{y \neq y' \vee L \vee K, \Gamma'}$
EqRes	$\dfrac{(\Gamma' =)\ x \neq x \vee L, \Gamma}{L, \Gamma'}$ L nonempty
Factor	$\dfrac{(\Gamma' =)\ x = y \vee x = z \vee L, \Gamma}{x = z \vee y \neq z \vee L, \Gamma'}$
Contrad	$\dfrac{x \neq x, \Gamma}{\bot}$

Superposition Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\Gamma_1 =)\ y = z,\quad x = y \vee x = z,\quad x \neq y \vee x \neq z}{(\Gamma_2 =)\ x = z \vee y \neq z,\quad \Gamma_1}\ \textbf{Factor}}{(\Gamma_3 =)\ x \neq z \vee y \neq z,\quad \Gamma_2}\ \textbf{Left}(\Gamma_{2.1}, \Gamma_{1.3})}{(\Gamma_4 =)\ z \neq z \vee y \neq z,\quad \Gamma_3}\ \textbf{Left}(\Gamma_{3.1}, \Gamma_{2.1})}{(\Gamma_5 =)\ y \neq z,\quad \Gamma_4}\ \textbf{EqRes}}{z \neq z,\quad \Gamma_5}\ \textbf{Left}(\Gamma_{5.1}, \Gamma_{1.1})}{\bot}\ \textbf{Contrad}$$

Correctness

Progress: For n variables, there are $n * (n+1)/2$ possible atoms, and therefore $2 * 2^{n*(n+1)/2}$ possible clauses.
Conservation: Easily checked.
Canonicity: Given irreducible Γ closed under the inference rules with variables $x_1, \ldots, x_n$ where $x_i \prec x_{i+1}$, iteratively build a union-find structure $\langle F_i, D_i \rangle$, where

$$
\begin{array}{rcl}
\langle F_0, D_0 \rangle & = & \langle \emptyset, \emptyset \rangle \\
F_{i+1} & = & \begin{cases} F_i \cup \{x_{i+1} = y'\}, \text{ where} \\ \quad x_{j+1} = y \vee L \text{ is the minimal clause in } \Gamma \\ \quad \text{unassigned in } M_i \quad \text{and } y' \equiv F_i(y) \\ F_i, \text{ otherwise} \end{cases} \\
D_{i+1} & = & D_i \cup \{x_{i+1} \neq x_j | j \leq i, F_{i+1}(x_j) \not\equiv F_{i+1}(x_{i+1})\} \\
M_i & = & \langle F_i, D_i \rangle
\end{array}
$$

Congruence Closure

Contrad	$\dfrac{G; F; D; U}{\bot}$ if $F(x) \equiv F(y)$ for $x \neq y \in D$
Delete	$\dfrac{x = y, G; F; D; U}{G; F; D; U}$ if $x' \equiv F(x) \equiv F(y) \equiv y'$
InputEq	$\dfrac{x = y, G; F; D; U}{G; \mathit{orient}(x' = y') \circ F; D; U}$ if $x' \equiv F(x) \not\equiv F(y) \equiv y'$
InputDeq	$\dfrac{x \neq y, G; F; D; U}{G; F; D, x \neq y; U}$
Replace	$\dfrac{G\{f(x_1, \ldots, x_n)\}; F; D; U}{(G\{x\}; F; D); U}$ if $x = f(y_1, \ldots, y_n) \in U$, $F(x_i) \equiv F(y_i)$, for $1 \leq i \leq n$
Abstract	$\dfrac{G\{f(x_1, \ldots, x_n)\}; F; D; U}{G\{x\}; F; D; x = f(x_1, \ldots, x_n), U}$ for fresh x
Propagate	$\dfrac{G; F; D; U}{G; \mathit{orient}(x' = y') \circ F; D; U}$ if $x' \equiv F(x) \not\equiv F(y) \equiv y'$, $x = f(x_1, \ldots, x_n) \in U$, $y = f(y_1, \ldots, y_n) \in U$, $F(x_i) \equiv F(y_i)$, for $1 \leq i \leq n$

- The CDCL inference system can be extended to handle theory atoms, e.g., $2x + 3y < 18$ instead of $p, \neg q$, etc.
- In addition to the rules **Propagate**, **Select**, **Conflict**, **Backjump**, we can add a single rule:

TBackjump	$\dfrac{h+1, \langle M \rangle, K, C}{h', \langle M_{\leq h'}, l[\Gamma'] \rangle, K', C'}$	$\models_{\mathcal{T}} \Gamma$ for $\Gamma = l_1 \vee \ldots \vee l_n$ $\{\overline{l_1}, \ldots, \overline{l_n}\} \in M$ $\langle h', \Gamma' \rangle$ $= \mathit{analyze}(\psi)(\Gamma)$ for $\psi = h, \langle M \rangle, K, C$ $K' = K \cup \{\Gamma\}$, $C' = C \cup \{\Gamma'\}$

Semantic Inference System: MCSAT & CDSAT

- When the inference state ψ contains a partial assignment M as in CDCL, it is a *Semantic Inference System*.
- MCSAT [de Moura/Jovanovic] and CDSAT [Bonacina/Graham–Lengrand/S] generalize the CDCL strategy to theories and theory combinations, respectively.
- The trail now includes Boolean assignments $p \mapsto \top$ as well as non-Boolean assignment $x \mapsto 3$.
- When an assignment to a variable is impossible, there is a conflict, and the theory produces a lemma.
- The trail is backjumped to explore a new path of assignments.
- A conflict at level 0 signals unsatisfiability.

Modularity [Ganzinger/Ruess/S, 2004]

- Inference modules generalize generalize inference systems to theory combinations.
- Each inference module has a public state and a private state.
- The public state is shared with other inference modules.
- When each of the theory states is irreducible and non-$\perp$, amalgamation can be used to compose a joint model in the union of the theories.

Algorithms as Inference: GCD

- Let $gcd(x, y, z)$ represent the formula

$$\begin{array}{rl} & x \geq 0 \wedge y \geq 0 \wedge x + y > 0 \wedge z > 0 \\ \wedge & z|x \wedge z|y \\ \wedge & (\forall z' : z'|x \wedge z'|y \implies z' \leq z) \end{array}$$

- There is only one inference step:

$$\frac{gcd(x, y, z)}{gcd((x \sqcup y) - (x \sqcap y), x \sqcap y, z)} \text{ if } x > 0, y > 0$$

- *Conservative*, because any divisor of x, y such that $x > y > 0$ is also a divisor of $x - y$.
- *Progressive*, because $x + y$ decreases with each inference.
- *Canonizing*, because if $x = 0$ or $y = 0$, then $z = x \sqcup y$.

Algorithms as Inference: Sorting

- Given an array A of k integers, such an array is sorted if $\forall i < j < k : (A(i) \not> A(j))$, i.e., there are no inversions.
- The predicate $sort(A, B)$ holds if B is a sorted, permutation of A.
- The initial state is $sort(a, B)$.
- The inference rule is

$$\frac{sort(A, B)}{sort(A', B)}\ i < j, A(i) > A(j), A' = swap(A, i, j)$$

- *Conservative*: Each swap step preserves solvability.
- *Progressive*: Each swap is a decreasing step in a lexicographic ordering.
- *Canonizing*: When no swaps are possible, the array is sorted.
- Without a swapping strategy, the number of swaps can be quadratic in addition to the indexing cost of finding swaps.

Algorithms as Inference: Dijkstra

- Given a weighted directed graph $G = (V, E, W)$, with non-negative edge weights, find the minimal path weight from a source vertex s to each vertex, i.e., solve for $|V|$ variables $P(v)$, for $v \in V$, such that

$$\begin{aligned} P(s) &= 0 \\ P(v) &= \bigsqcap\{P(u) + W(u,v) \mid u \in V\}, \text{ for } v \neq s. \end{aligned}$$

- With $post(X)(v) = \bigsqcap\{X(u) + W(u,v) \mid u \in dom(X)\}$, the logical state has two partial maps P and Q:
  1. Each $v \in V$ is either in $dom(P)$ or $dom(Q)$, but not both,
  2. $post(P)(v) = P(v)$ for $v \in dom(P)$,
  3. $post(P)(v) = Q(v)$ for $v \in dom(Q)$, and
  4. $P(u) \leq Q(v)$ for $u \in dom(P)$ and $v \in dom(Q)$.
- Initially, $P = [s \mapsto 0]$, and $Q = [v \mapsto \infty \mid v \neq s]$.
- Each inference step takes $\langle P, Q\rangle$ to $\langle P', Q'\rangle$, where $u = argmin_u Q(u)$, $P' = P[u \mapsto Q(u)]$, and $Q' = [v \mapsto Q(v) \sqcap (Q(u) + W(u,v)) \mid v \in dom(Q) - \{u\}]$.

Conclusions

- Inference systems offer a mathematical foundation for the study of algorithmic problem solving.
- Algorithms can be presented with simpler proofs and stronger separation of concerns between inference, strategy, and indexing.
- Many conventional algorithms also can be presented lucidly as inference systems.
- Seeing computing as inference, not calculation, offers a more illuminating metatheory for computation.