Compactness

Compactness Theorem

If every finite subset Γ_0 of Γ is satisfiable, then Γ is satisfiable.

Proof

Suppose every finite subset Γ_0 of Γ is satisfiable. By soundness, every finite subset is consistent. But since deductions are finite, it follows that Γ is consistent. Thus, by completeness, Γ is satisfiable.

Corollary

If $\Gamma \models \phi$, then for some finite $\Gamma_0 \subseteq \Gamma$ we have $\Gamma_0 \models \phi$.

Proof

Suppose to the contrary that $\Gamma_0 \not\models \phi$ for every finite $\Gamma_0 \subseteq \Gamma$. Then every finite subset of $\Gamma \cup \{\neg\phi\}$ is satisfiable, and thus $\Gamma \cup \{\neg\phi\}$ is satisfiable. It follows that $\Gamma \not\models \phi$.

Reasonable Languages

A *reasonable* language is one whose signature can be effectively enumerated and such that the two relations

$$\{\langle P, n\rangle \mid P \text{ is an } n\text{-ary predicate symbol }\}$$

and

$$\{\langle f, n\rangle \mid f \text{ is an } n\text{-ary function symbol }\}$$

are decidable.

Any language constructed from a finite signature is reasonable. A reasonable language must be countable.

Enumerability Theorem

Theorem

For a reasonable language, if Γ is a decidable set of formulas, then the set of all theorems of Γ is effectively enumerable.

Proof

First note that for a reasonable language, the set Λ of axioms is decidable. Given an expression, we can effectively check whether it is well-formed and whether it is a tautology or a syntactic instance of any of the other axiom groups.

Recall from lecture 2 that the set of tautological consequences of an effectively enumerable set is effectively enumerable. But as we proved earlier, ϕ is a tautological consequence of $\Gamma \cup \Lambda$ iff $\Gamma \vdash \phi$. And $\Gamma \vdash \phi$ iff $\Gamma \models \phi$ by soundness and completeness.

Corollaries

Corollary

The set of valid formulas in a reasonable language is effectively enumerable.

Corollary

If Γ is a decidable set of formulas in a reasonable language and for any sentence σ, either $\Gamma \models \sigma$ or $\Gamma \models \neg\sigma$, then the set of sentences implied by Γ is decidable.

Proof

Enumerate the theorems of Γ until either σ or $\neg\sigma$ is obtained.

Definability of a Class of Models

A *theory* is a set of sentences. For a given signature Σ, a Σ*-theory* is a set of sentences, each of which is a Σ-formula.

If $\mathcal{K}$ is a class of models with signature Σ, we say that a Σ-theory T *axiomatizes* $\mathcal{K}$ or is a *set of axioms* for $\mathcal{K}$ if $\mathcal{K}$ is the class of all models of T.

For an arbitrary Σ-theory T, let $Mod\,T$ be the class of all models of T (over signature Σ).

A class $\mathcal{K}$ of models is *first-order definable*, also known as an *elementary class* (EC), iff $\mathcal{K} = Mod\,\tau$ for some sentence τ.

A class $\mathcal{K}$ of models is *first-order axiomatizable*, or *generalized first-order definable*, also known as an *elementary class in the wider sense* (EC_Δ), iff $\mathcal{K} = Mod\,T$ for some set of sentences T.

Example

Consider a signature (P) with a single binary predicate symbol P.

A model (A, R) is an *ordered set* iff R is transitive and satisfies the trichotomy condition (which states that for any $a, b \in A$ exactly one of $\langle a, b\rangle \in R$, $a = b$, $\langle b, a\rangle \in R$ holds).

This can be written as a first-order sentence as follows:

$$\forall x\,\forall y\,\forall z\,(Pxy \to Pyz \to Pxz) \wedge$$
$$\forall x\,\forall y\,(Pxy \vee x = y \vee Pyx) \wedge$$
$$\forall x\,\forall y\,(Pxy \to \neg Pyx).$$

Because these can be be translated into a sentence, the class of (nonempty) ordered sets is first-order definable.

Example

Consider a signature $(\circ)$ with a single binary function symbol $\circ$.

The class of all groups is defined by the following sentence:

$$\forall x \forall y \forall z \, (x \circ (y \circ z) = (x \circ y) \circ z) \wedge$$
$$\forall x \forall y \exists z \, (x \circ z = y) \wedge$$
$$\forall x \forall y \exists z \, (z \circ x = y).$$

The class of all *infinite* groups is first-order axiomatizable. To see this, let

$$\lambda_2 = \exists x \exists y \, x \neq y,$$
$$\lambda_3 = \exists x \exists y \exists z \, (x \neq y \wedge x \neq z \wedge y \neq z),$$
$$\ldots$$
$\lambda_k =$ there are at least k distinct objects

Then the class of infinite groups is axiomatized by the sentence for groups together with the set of sentences $\{\lambda_2, \lambda_3, \ldots\}$.

Finite Models

A similar trick can be used to show the following:

Theorem

If a set Γ of sentences has arbitrarily large finite models, then it has an infinite model.

Proof

As before, for each integer $k \geq 2$, let λ_k be the sentence that translates, "there are at least k distinct objects".

Now, consider the set $\Gamma \cup \{\lambda_2, \lambda_3, \ldots\}$. By hypothesis, any finite subset has a model. So by compactness the entire set has a model, which clearly must be infinite.

Corollary

The class $\mathcal{K}_f$ of all finite models (for a fixed signature) is not EC_Δ, i.e. there is no set of sentences Γ such that $\mathcal{K}_f = \mathit{Mod}\ \Gamma$.

Proof

Suppose such a set Γ existed. Then since Γ has models of arbitrarily large finite size, it must also have an infinite model, which is a contradiction.

Corollary

The class of all infinite models is EC_Δ but not EC.

Proof

The set $\lambda_2, \lambda_3, \lambda_4, \ldots$ is a first-order axiomatization of the class of all infinite models. Suppose that the class is EC. Then it is equal to $\mathit{Mod}\,\tau$ for some first-order sentence τ. But then $\mathcal{K}_f = \mathit{Mod}\,\neg\tau$ and we know that $\mathcal{K}_f$ is not even EC_Δ.

For a model M over a given signature Σ, define the *theory of* M as

$$\mathit{Th}\ M = \{\sigma \mid \sigma \text{ is a } \Sigma\text{-sentence which is true in } M\}.$$

Theorem

For a finite model M in a finite language, $\mathit{Th}\ M$ is decidable.

Proof

1. Any finite model M of size n is isomorphic to a model whose domain is $1 \ldots n$.
2. We can check whether a model with domain $1 \ldots n$ satisfies a sentence by building an evaluation tree which enumerates all possible assignments for each quantifier. Because the domain and the sentence are both finite, the evaluation tree will also be finite.

Theorem

For a finite language, $\{\sigma | \sigma$ has a finite model $\}$ is effectively enumerable.

Proof

1. Given a sentence σ and a positive integer n, we can effectively decide whether or not σ has an n-element model. This is because there are only finitely many models to check.
2. This gives us a semi-decision procedure for whether σ has a finite model. First check if it has a model of size 1, then of size 2, ...

Corollary

If Φ is the set of sentences true in every finite model, then its complement, $\overline{\Phi}$, is effectively enumerable.

Theorem

For a finite language, $\{\sigma | \sigma$ has a finite model $\}$ is effectively enumerable.

Proof

1. Given a sentence σ and a positive integer n, we can effectively decide whether or not σ has an n-element model. This is because there are only finitely many models to check.
2. This gives us a semi-decision procedure for whether σ has a finite model. First check if it has a model of size 1, then of size 2, ...

Corollary

If Φ is the set of sentences true in every finite model, then its complement, $\overline{\Phi}$, is effectively enumerable.

$\sigma \in \overline{\Phi}$ iff $(\neg\sigma)$ has a finite model.

Size of Models

The cardinality $|L|$ of a language L is the least infinite cardinal greater than or equal to the number of symbols in the signature of L.

The cardinality $|M|$ of a model M is the cardinality of its domain $dom(M)$.

Löwenheim-Skolem (LS) Theorem

Let Γ be a satisfiable set of formulas in a language L, then Γ is satisfiable in some model of cardinality $\kappa \leq |L|$.

Proof

By soundness, Γ is consistent, and is thus satisfiable by the model constructed in the proof of the completeness theorem. But the domain of that model is M/E which has cardinality $\leq |M|$, and $|M| = |L|$.

"Skolem's paradox"

Let A_{ST} be your favorite set of axioms for set theory. If they are consistent, they have a model. Because the signature of the language of set theory is finite, there is a countable model. But we can prove, starting with the axioms of set theory, that there are "uncountably" many sets.

How is this possible?

The answer is that in the countable model of set theory, things do not correspond to what we normally think of as the model of set theory. Thus, the model of the "natural numbers" in the countable model cannot be put in one-to-one correspondence with all of the elements of the model. But this does not mean that the size of the model is truly uncountable.

LST Theorem

Let Γ be a set of formulas in a language of cardinality κ, and assume that Γ is satisfiable in some infinite model. Then for every cardinal $\lambda \geq \kappa$, there is a model of cardinality λ in which Γ is satisfiable.

Proof

Let M be an infinite model where Γ is satisfiable. Expand the language by adding a set C of λ new constant symbols. Let $\Delta = \{c_1 \neq c_2 \mid c_1, c_2$ are distinct members of $C\}$. Then, any finite subset Γ_0 of $\Gamma \cup \Delta$ is satisfiable in M' where M' is M extended to map all constants in Γ_0 to different elements of M. By compactness, $\Gamma \cup \Delta$ is satisfiable. By the **LS Theorem**, there is a model of cardinality $\leq \lambda$. But any model must have at least cardinality λ. Thus there is a model of cardinality λ.

Corollary

If Γ is a set of sentences in a countable language, then if Γ has some infinite model, it has models of every infinite cardinality.

Two models M and M' with the same signature are *elementarily equivalent* ($M \equiv M'$) iff for any sentence σ, $\models_M \sigma$ iff $\models_{M'} \sigma$.

Corollary

If M is an infinite model for a countable language, then for any infinite cardinal λ, there is a model M' of cardinality λ such that $M \equiv M'$.

Proof

Let Γ be the set of all sentences true in M. By the corollary above, Γ has a model M' of cardinality λ. But note that for every sentence σ, either $\sigma \in \Gamma$ or $\neg\sigma \in \Gamma$ (why?). Thus, $M \equiv M'$.

Theories

Last time, we defined a *theory* as a set of first-order sentences.

For this lecture we will refine our definition to be a set of first-order sentences *closed under logical implication*.

Thus, T is a theory iff T is a set of sentences and if $T \models \sigma$, then $\sigma \in T$ for every sentence σ.

What is the smallest possible theory?

For a given signature, the smallest possible theory consists of exactly the valid sentences over that signature.

What is the largest possible theory?

The largest theory for a given signature is the set of all sentences. It is the only unsatisfiable theory.

Why?

For a class $\mathcal{K}$ of models over a given signature Σ, define the *theory of* $\mathcal{K}$ as

$$Th\,\mathcal{K} = \{\sigma \mid \sigma \text{ is a } \Sigma\text{-sentence which is true in every model in } \mathcal{K}\}.$$

Theorem

$Th\,\mathcal{K}$ is indeed a theory.

Proof

Suppose $Th\,\mathcal{K} \models \sigma$. We know that $\models_M Th\,\mathcal{K}$ for each M in $\mathcal{K}$. It follows that $\models_M \sigma$ for each M in $\mathcal{K}$, and thus $\sigma \in Th\,\mathcal{K}$.

Suppose Γ is a set of sentences.

Define the set $Cn\ \Gamma$ of *consequences* of Γ to be $\{\sigma \mid \Gamma \models \sigma\}$.

Then $Cn\ \Gamma = Th\,Mod\ \Gamma$.

A theory T is *complete* iff for every sentence σ, either $\sigma \in T$ or $(\neg\sigma) \in T$.

Note that if M is a model, then $Th\ \{M\}$ is complete. In fact, for a class $\mathcal{K}$ of models, $Th\,\mathcal{K}$ is complete iff any two members of $\mathcal{K}$ are elementarily equivalent.

A theory T is *axiomatizable* iff there is a decidable set Γ of sentences such that $T = Cn\ \Gamma$.

A theory T is *finitely axiomatizable* iff $T = Cn\ \Gamma$ for some finite set Γ of sentences.

Theorem

If $Cn\ \Gamma$ is finitely axiomatizable, then there is a finite $\Gamma_0 \subseteq \Gamma$ such that $Cn\ \Gamma_0 = Cn\ \Gamma$.

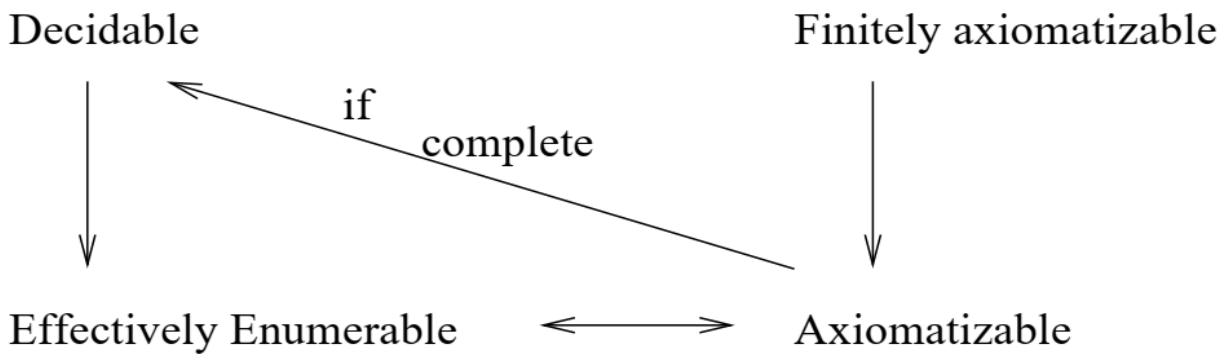
Proof

If $Cn\ \Gamma$ is finitely axiomatizable, then for some sentence τ, $Cn\ \Gamma = Cn\ \tau$. Clearly, $\Gamma \models \tau$. By compactness, we have that there exists $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \models \tau$. Thus, $Cn\ \tau \subseteq Cn\ \Gamma_0 \subseteq Cn\ \Gamma$, and since $Cn\ \Gamma = Cn\ \tau$, it follows that $Cn\ \Gamma_0 = Cn\ \Gamma$.

Using the above terminology, we can restate our earlier results as follows:

- An axiomatizable theory (in a reasonable language) is effectively enumerable.
- A complete axiomatizable theory (in a reasonable language) is decidable.

Our results about theories can be summarized in the following diagram.



Los-Vaught Test

For a theory T and a cardinal λ, say that T is *λ-categorical* iff all models of T having cardinality λ are isomorphic.

Theorem

Let T be a theory in a countable language such that

- T is λ-categorical for some infinite cardinal λ.
- All models of T are infinite.

Then T is complete.

Proof

It suffices to show that for any two models M and M' of T, $M \equiv M'$. Since M and M' are infinite, there exist (by **LST**) elementarily equivalent models of cardinality λ. But these models must be isomorphic, and by the homomorphism theorem, isomorphic models are elementarily equivalent.

Validity and Satisfiability Modulo Theories

Given a Σ-theory T, a Σ-formula ϕ is

1. *T-valid* if $\models_M \phi[s]$ for all models M of T and all variable assignments s.
2. *T-satisfiable* if there exists some model M of T and variable assignment s such that $\models_M \phi[s]$.
3. *T-unsatisfiable* if $\not\models_M \phi[s]$ for all models M of T and all variable assignments s.

The *validity problem* for T is the problem of deciding, for each Σ-formula ϕ, whether ϕ is T-valid.

The *satisfiability problem* for T is the problem of deciding, for each Σ-formula ϕ, whether ϕ is T-satisfiable.

Similarly, one can define the *quantifier-free validity problem* and the *quantifier-free satisfiability problem* for a Σ-theory T by restricting the formula ϕ to be quantifier-free.

A decision problem is *decidable* if there exists an effective procedure which always terminates with an answer for any given instance of the problem.

For example, the validity problem for a Σ-theory T is decidable if there exists an effective procedure for determining whether $T \models \phi$ for every Σ-formula ϕ.

Note that validity problems can always be reduced to satisfiability problems:

ϕ is T-valid iff $\neg\phi$ is T-unsatisfiable.

We will consider a few examples of theories which are of particular interest in verification applications.

The Theory $T_{\mathcal{E}}$ of Equality

The theory $T_{\mathcal{E}}$ of equality is the theory $Cn\ \emptyset$.

Note that the exact set of sentences in $T_{\mathcal{E}}$ depends on the signature in question.

The theory does not restrict the possible values of symbols in any way. For this reason, it is sometimes called the theory of *equality with uninterpreted functions (EUF)*.

The satisfiability problem for $T_{\mathcal{E}}$ is just the satisfiability problem for first order logic, which is undecidable.

The satisfiability problem for conjunctions of literals in $T_{\mathcal{E}}$ is decidable in polynomial time using *congruence closure*.

The Theory $T_{\mathcal{Z}}$ of Integers

Let $\Sigma_{\mathcal{Z}}$ be the signature $(0, 1, +, -, \leq)$.

Let $\mathcal{A}_{\mathcal{Z}}$ be the standard model of the integers with domain $\mathcal{Z}$.

Then $T_{\mathcal{Z}}$ is defined to be $Th\,\mathcal{A}_{\mathcal{Z}}$.

As showed by Presburger in 1929, the validity problem for $T_{\mathcal{Z}}$ is decidable, but its complexity is triply-exponential.

The quantifier-free satisfiability problem for $T_{\mathcal{Z}}$ is "only" NP-complete.

Let $\Sigma_{\mathcal{Z}}^{\times}$ be the same as $\Sigma_{\mathcal{Z}}$ with the addition of the symbol $\times$ for multiplication, and define $\mathcal{A}_{\mathcal{Z}}^{\times}$ and $T_{\mathcal{Z}}^{\times}$ in the obvious way.

The satisfiability problem for $T_{\mathcal{Z}}^{\times}$ is undecidable (a consequence of Gödel's incompleteness theorem).

In fact, even the quantifier-free satisfiability problem for $T_{\mathcal{Z}}^{\times}$ is undecidable.

The Theory $T_{\mathcal{R}}$ of Reals

Let $\Sigma_{\mathcal{R}}$ be the signature $(0, 1, +, -, \leq)$.

Let $\mathcal{A}_{\mathcal{R}}$ be the standard model of the reals with domain $\mathcal{R}$.

Then $T_{\mathcal{R}}$ is defined to be $\mathit{Th}\,\mathcal{A}_{\mathcal{R}}$.

The satisfiability problem for $T_{\mathcal{R}}$ is decidable, but the complexity is doubly-exponential.

The quantifier-free satisfiability problem for conjunctions of literals (atomic formulas or their negations) in $T_{\mathcal{R}}$ is solvable in polynomial time, though exponential methods (like Simplex or Fourier-Motzkin) often perform better in practice.

Let $\Sigma_{\mathcal{R}}^{\times}$ be the same as $\Sigma_{\mathcal{R}}$ with the addition of the symbol $\times$ for multiplication, and define $\mathcal{A}_{\mathcal{R}}^{\times}$ and $T_{\mathcal{R}}^{\times}$ in the obvious way.

In contrast to the theory of integers, the satisfiability problem for $T_{\mathcal{R}}^{\times}$ is decidable.

The Theory $T_{\mathcal{A}}$ of Arrays

Let $\Sigma_{\mathcal{A}}$ be the signature $(\mathit{read}, \mathit{write})$.

Let $\Lambda_{\mathcal{A}}$ be the following axioms:

$$\forall a \, \forall i \, \forall v \, (\mathit{read}(\mathit{write}(a, i, v), i) = v)$$
$$\forall a \, \forall i \, \forall j \, \forall v \, (i \neq j \rightarrow \mathit{read}(\mathit{write}(a, i, v), j) = \mathit{read}(a, j))$$
$$\forall a \, \forall b \, ((\forall i \, (\mathit{read}(a, i) = \mathit{read}(b, i))) \rightarrow a = b)$$

Then $T_{\mathcal{A}} = \mathit{Cn} \ \Lambda_{\mathcal{A}}$.

The satisfiability problem for $T_{\mathcal{A}}$ is undecidable, but the quantifier-free satisfiability problem for $T_{\mathcal{A}}$ is decidable (the problem is NP-complete).

Theories of Inductive Data Types

An *inductive data type* (IDT) defines one or more *constructors*, and possibly also *selectors* and *testers*.

Example: *list* **of** *int*

- Constructors: $\mathit{cons} : (\mathit{int}, \mathit{list}) \rightarrow \mathit{list}$, $\mathit{null} : \mathit{list}$
- Selectors: $\mathit{car} : \mathit{list} \rightarrow \mathit{int}$, $\mathit{cdr} : \mathit{list} \rightarrow \mathit{list}$
- Testers: *is_cons*, *is_null*

The *first order theory* of a inductive data type associates a function symbol with each constructor and selector and a predicate symbol with each tester.

Example: $\forall x : \mathit{list}. \, (x = \mathit{null} \vee \exists y : \mathit{int}, z : \mathit{list}. \, x = \mathit{cons}(y, z))$

For IDTs with a single constructor, a conjunction of literals is decidable in polynomial time.

For more general IDTs, the problem is NP-complete, but reasonbly efficient algorithms exist in practice.

Other Interesting Theories

Some other interesting theories include:

- Theories of bit-vectors
- Fragments of set theory

Congruence Closure

Let $G = (V, E)$ be a directed graph such that for each vertex v in G, the successors of v are ordered.

Let C be any equivalence relation on V.

The *congruence closure* C^* of C is the finest equivalence relation on V that contains C and satisfies the following property for all vertices v and w.

> Let v and w have successors $v_1, \ldots, v_k$ and $w_1, \ldots, w_l$ respectively. If $k = l$ and $(v_i, w_i) \in C^*$ for $1 \leq i \leq k$, then $(v, w) \in C^*$.

In other words, if the corresponding successors of v and w are equivalent under C^*, then v and w are themselves equivalent under C^*.

Often, the vertices are labeled by some labeling function λ. In this case, the property becomes:

> If $\lambda(v) = \lambda(w)$ and if $k = l$ and $(v_i, w_i) \in C^*$ for $1 \leq i \leq k$, then $(v, w) \in C^*$.

A Simple Algorithm

Let $C_0 = C$ and $i = 0$.

1. Number the equivalence classes in C_i consecutively from 1.
2. Let α assign to each vertex v the number $\alpha(v)$ of the equivalence class containing v.
3. For each vertex v construct a *signature* $s(v) = \lambda(v)(\alpha(v_1), \ldots, \alpha(v_k))$, where $v_1, \ldots, v_k$ are the successors of v.

4. Group the vertices into classes of vertices having equal signatures.
5. Let C_{i+1} be the finest equivalence relation on V such that two vertices equivalent under C_i or having the same signature are equivalent under C_{i+1}.
6. If $C_{i+1} = C_i$, let $C^* = C_i$; otherwise increment i and repeat.

Congruence Closure and $T_{\mathcal{E}}$

Recall that $T_{\mathcal{E}}$ is the empty theory with equality over some signature Σ containing only function symbols.

If Γ is a set of ground Σ-equalities and Δ is a set of ground Σ-disequalities, then the satisfiability of $\Gamma \cup \Delta$ can be determined as follows.

- Let G be a graph which corresponds to the abstract syntax trees of terms in $\Gamma \cup \Delta$, and let v_t denote the vertex of G associated with the term t.
- Let C be the equiavlence relation on the vertices of G induced by Γ.
- $\Gamma \cup \Delta$ is satisfiable iff for each $s \neq t \in \Delta$, $(v_s, v_t) \notin C^*$.

An Algorithm for $T_{\mathcal{E}}$

union and *find* are abstract operations for manipulating equivalence classes.

union(x, y) merges the equivalence classes of x and y.

find(x) returns a unique representative of the equivalence class of x.

```
CC(Γ, Δ)
    Construct G(V, E) from terms in Γ and Δ.
    while Γ ≠ ∅
      Remove some equality a = b from Γ;
      Merge(a, b);
    if find(a) = find(b) for some a ≠ b ∈ Δ then
      return false;
    return true;

Merge(a, b)
    if find(a) = find(b) then return;
    Let A be the set of all predecessors
      of all vertices equivalent to a;
    Let B be the set of all predecessors
      of all vertices equivalent to b;
    union(a, b);
    foreach x ∈ A and y ∈ B
      if signature(x) = signature(y) then Merge(x, y);
```

Congruence Closure

DST Algorithm

The Downey-Sethi-Tarjan Congruence Closure algorithm is more efficient. It makes use of some additional data structures and methods.

Additional Helper Methods

- *union*(a, b) in this algorithm, the *first* argument always becomes the new equivalence class representative.
- *list*(e) returns the list of vertices with at least one successor in equivalence class e.
- *enter*(v) stores $(v, \textit{signature}(v))$ in a signature table.
- *delete*(v) removes $(v, \textit{signature}(v))$ from the signature table if it is there. Note that this operation does *not* remove any other entry, even if it has the same signature as v.
- *query*(v) if there is an entry $(w, \textit{signature}(w)$ in the signature table, and $\textit{signature}(w) = \textit{signature}(v)$, then return w; otherwise, return $\perp$.

$CC(\Gamma, \Delta)$

Construct $G(V, E)$ from terms in Γ and Δ.

$Merge(\Gamma)$;

if $find(a) = find(b)$ for some $a \neq b \in \Delta$ **then**

return *false* ;

return *true* ;

```
Merge(combine)
  pending := set of all vertices;
  while pending ≠ ∅
    foreach v ∈ pending
      if query(v) = ⊥ then enter(v);
      else add (v, query(v)) to combine;
    pending := ∅;
    foreach (a, b) ∈ combine
      if find(a) ≠ find(b) then
        if |list(find(a))| < |list(find(b))| then swap a and b;
        foreach u ∈ list(find(b))
          delete(u); add u to pending;
        union(find(a), find(b));
    combine := ∅;
```

Shostak's Method

Robert Shostak published a paper in 1984 which detailed a particular strategy for deciding validity of quantifier-free formulas in certain kinds of theories.

Unfortunately, the original algorithm contained many errors and a number of papers have since been dedicated to correcting them.

We will look at a simplified version of Shostak's algorithm which is easily proved correct, yet still contains most of the essential ideas introduced by the original paper.

Equations in Solved Form

A set $\mathcal{E}$ of equations is said to be in *solved form* iff the left-hand side of each equation in $\mathcal{E}$ is a variable which appears only once in $\mathcal{E}$.

We will refer to these variables which appear only on the left-hand sides as *solitary* variables.

A set $\mathcal{E}$ of equations in solved form defines an idempotent substitution: the one which replaces each solitary variable with its corresponding right-hand side.

If X is an expression or set of expressions, we denote the result of applying this substitution to X by $\mathcal{E}(X)$.

Another interesting property of equations in solved form is that the question of whether such a set $\mathcal{E}$ entails some formula ϕ in a theory T can be answered simply by determining the validity of $\mathcal{E}(\phi)$ in T.

Solved Form Theorem

If T is a theory with signature Σ and $\mathcal{E}$ is a set of Σ-equations in solved form, then $T \cup \mathcal{E} \models \phi$ iff $T \models \mathcal{E}(\phi)$.

Proof

Clearly, $T \cup \mathcal{E} \models \phi$ iff $T \cup \mathcal{E} \models \mathcal{E}(\phi)$.

Thus we only need to show that $T \cup \mathcal{E} \models \mathcal{E}(\phi)$ iff $T \models \mathcal{E}(\phi)$.

The "if" direction is trivial.

To show the other direction, assume that $T \cup \mathcal{E} \models \mathcal{E}(\phi)$. Any model of T can be made to satisfy $T \cup \mathcal{E}$ by assigning any value to the non-solitary variables of $\mathcal{E}$, and then choosing the value of each solitary variable to match the value of its corresponding right-hand side.

Since none of the solitary variables occur anywhere else in $\mathcal{E}$ this assignment is well-defined and satisfies $\mathcal{E}$.

By assumption then, this model and assignment also satisfy $\mathcal{E}(\phi)$, but none of the solitary variables appear in $\mathcal{E}(\phi)$, so the initial arbitrary assignment to non-solitary variables must be sufficient to satisfy $\mathcal{E}(\phi)$.

Thus it must be the case that every model of T satisfies $\mathcal{E}(\phi)$ with every variable assignment.

By setting ϕ to $false$, the following corollary is obtained.

Corollary

If T is a satisfiable theory with signature Σ and $\mathcal{E}$ is a set of Σ-equations in solved form, then $T \cup \mathcal{E}$ is satisfiable.

Shostak Theories

A consistent theory T with signature Σ is a *Shostak* theory if the following conditions hold.

1. Σ does not contain any predicate symbols.
2. T is convex. A theory T is said to be *convex* if for any conjunction of literals ϕ and set of equations between variables $x_1 = y_1, \ldots x_n = y_n$, if $T \cup \phi \models x_1 = y_1 \vee \cdots \vee x_n = y_n$, then in fact $T \cup \phi \models x_i = y_i$ for some $1 \leq i \leq n$.
3. There exists a *canonizer* *canon*, a computable function from Σ-terms to Σ-terms, with the property that $T \models a = b$ iff *canon*$(a) \equiv$ *canon*(b).
4. There exists a *solver* *solve*, a computable function from Σ-equations to sets of formulas defined as follows:
   (a) If $T \models a \neq b$, then *solve*$(a = b) \equiv \{false\}$.
   (b) Otherwise, *solve*$(a = b)$ returns a set $\mathcal{E}$ of equations in solved form such that $T \models [(a = b) \leftrightarrow \exists\, \overline{w}.\ \mathcal{E}]$, where $\overline{w}$ is a set of fresh variables which appear in $\mathcal{E}$ but not in a or b.

Canonizer

The canonizer is used to determine whether a specific equality is entailed by a set of equations in solved form.

Theorem (canon)

If $\mathcal{E}$ is a set of Σ-equations in solved form, then

$$T \cup \mathcal{E} \models a = b \text{ iff } \mathit{canon}(\mathcal{E}(a)) \equiv \mathit{canon}(\mathcal{E}(b)).$$

Proof

By the **Solved Form Theorem**, $T \cup \mathcal{E} \models a = b$ iff $T \models \mathcal{E}(a) = \mathcal{E}(b)$. But $T \models \mathcal{E}(a) = \mathcal{E}(b)$ iff $\mathit{canon}(\mathcal{E}(a)) \equiv \mathit{canon}(\mathcal{E}(b))$ by the definition of *canon*.

Algorithm Sh

Algorithm Sh checks the satisfiability in T of a set of equalities, Γ, and an set of disequalities, Δ.

$Sh(\Gamma, \Delta, \mathit{canon}, \mathit{solve})$

1. $\mathcal{E}$:= $\emptyset$;
2. **while** $\Gamma \neq \emptyset$ **do begin**
3. Remove some equality $a = b$ from Γ;
4. a^* := $\mathcal{E}(a)$; b^* := $\mathcal{E}(b)$;
5. $\mathcal{E}^*$:= $\mathit{solve}(a^* = b^*)$;
6. **if** $\mathcal{E}^* = \{false\}$ **then return** *false* ;
7. $\mathcal{E}$:= $\mathcal{E}^*(\mathcal{E}) \cup \mathcal{E}^*$;
8. **end**
9. **if** $\mathit{canon}(\mathcal{E}(a)) \equiv \mathit{canon}(\mathcal{E}(b))$ for some $a \neq b \in \Delta$ **then**
   return *false* ;
10. **return** *true* ;

Correctness of Algorithm Sh

Termination of the algorithm is trivial since each step terminates and each time line 3 is executed the size of Γ is reduced.

The following lemmas are needed before proving correctness.

Lemma 1

If T' is a theory, Γ and Θ are sets of formulas, and $\mathcal{E}$ is a set of equations in solved form, then for any formula ϕ,

$$T' \cup \Gamma \cup \Theta \cup \mathcal{E} \models \phi \text{ iff } T' \cup \Gamma \cup \mathcal{E}(\Theta) \cup \mathcal{E} \models \phi.$$

Proof

Follows trivially from the fact that $\Theta \cup \mathcal{E}$ and $\mathcal{E}(\Theta) \cup \mathcal{E}$ are satisfied by exactly the same models and variable assignments.

Lemma 2

If Γ is any set of formulas, then for any formula ϕ, and Σ-terms a and b,

$$T \cup \Gamma \cup \{a = b\} \models \phi \text{ iff } T \cup \Gamma \cup \mathit{solve}(a = b) \models \phi.$$

Proof

$\Rightarrow$: Given that $T \cup \Gamma \cup \{a = b\} \models \phi$, suppose that $M \models_\rho T \cup \Gamma \cup \mathit{solve}(a = b)$. It is easy to see from the definition of *solve* that $M \models_\rho a = b$ and hence by the hypothesis, $M \models_\rho \phi$.

$\Leftarrow$: Given that $T \cup \Gamma \cup \mathit{solve}(a = b) \models \phi$, suppose that $M \models_\rho T \cup \Gamma \cup \{a = b\}$. Then, since $T \models (a = b) \leftrightarrow \exists \overline{w}.\ \mathit{solve}(a = b)$, there exists a modified assignment ρ^* which assigns values to all the variables in $\overline{w}$ and satisfies $\mathit{solve}(a = b)$ but is otherwise equivalent to ρ. Then, by the hypothesis, $M \models_{\rho^*} \phi$. But the variables in $\overline{w}$ are fresh variables, so they do not appear in ϕ, meaning that changing their values cannot affect whether ϕ is true. Thus, $M \models_\rho \phi$.

Lemma 3

If Γ, $\{a = b\}$, and $\mathcal{E}$ are sets of Σ-formulas, with $\mathcal{E}$ in solved form, and if $\mathcal{E}^* = \mathit{solve}(\mathcal{E}(a = b))$ then if $\mathcal{E}^* \neq \{false\}$, then for every formula ϕ,

$$T \cup \Gamma \cup \{a = b\} \cup \mathcal{E} \models \phi \text{ iff } T \cup \Gamma \cup \mathcal{E}^* \cup \mathcal{E}^*(\mathcal{E}) \models \phi.$$

Proof

$$\begin{array}{lll} T \cup \Gamma \cup \{a = b\} \cup \mathcal{E} \models \phi & \Leftrightarrow & T \cup \Gamma \cup \{\mathcal{E}(a = b)\} \cup \mathcal{E} \models \phi \quad \textbf{Lemma 1} \\ & \Leftrightarrow & T \cup \Gamma \cup \mathcal{E}^* \cup \mathcal{E} \models \phi \quad \textbf{Lemma 2} \\ & \Leftrightarrow & T \cup \Gamma \cup \mathcal{E}^* \cup \mathcal{E}^*(\mathcal{E}) \models \phi \quad \textbf{Lemma 1} \end{array}$$

Lemma 4

During the execution of Algorithm Sh, $\mathcal{E}$ is always in solved form.

Proof

Clearly, $\mathcal{E}$ is in solved form initially. Consider one iteration. By construction, a^* and b^* do not contain any of the solitary variables of $\mathcal{E}$, and thus by the definition of *solve*, $\mathcal{E}^*$ doesn't either. Furthermore, if $\mathcal{E}^* = \{false\}$ then the algorithm terminates at line 6. Thus, at line 7, $\mathcal{E}^*$ must be in solved form. Applying $\mathcal{E}^*$ to $\mathcal{E}$ guarantees that none of the solitary variables of $\mathcal{E}^*$ appear in $\mathcal{E}$, so the new value of $\mathcal{E}$ is also in solved form.

Lemma 5

Let Γ_n and $\mathcal{E}_n$ be the values of Γ and $\mathcal{E}$ after the while loop in Algorithm Sh has been executed n times. Then for each n, and any formula ϕ, the following invariant holds:

$$T \cup \Gamma_0 \models \phi \text{ iff } T \cup \Gamma_n \cup \mathcal{E}_n \models \phi.$$

Proof

The proof is by induction on n. For $n = 0$, the invariant holds trivially. Now suppose the invariant holds for some $k \geq 0$. Consider the next iteration.

$T \cup \Gamma_0 \models \phi$	$\Leftrightarrow$	$T \cup \Gamma_k \cup \mathcal{E}_k \models \phi$	Induction Hypothesis
	$\Leftrightarrow$	$T \cup \Gamma_{k+1} \cup \{a = b\} \cup \mathcal{E}_k \models \phi$	Line 3
	$\Leftrightarrow$	$T \cup \Gamma_{k+1} \cup \mathcal{E}^* \cup \mathcal{E}^*(\mathcal{E}_k) \models \phi$	**Lemmas 3** and **4**
	$\Leftrightarrow$	$T \cup \Gamma_{k+1} \cup \mathcal{E}_{k+1} \models \phi$	Line 7

Theorem

Suppose T is a Shostak theory with signature Σ, canonizer *canon*, and solver *solve*. If Γ is a set of Σ-equalities and Δ is a set of Σ-disequalities, then $T \cup \Gamma \cup \Delta$ is satisfiable iff $\mathrm{Sh}(\Gamma, \Delta, \mathit{canon}, \mathit{solve}) = \mathit{true}$.

Proof

Suppose $\mathrm{Sh}(\Gamma, \Delta, \mathit{canon}, \mathit{solve}) = \mathit{false}$.

If the algorithm terminates at line 9, then, $\mathit{canon}(\mathcal{E}(a)) \equiv \mathit{canon}(\mathcal{E}(b))$ for some $a \neq b \in \Delta$.

It follows from the **canon** theorem and **Lemma 5** that $T \cup \Gamma \models a = b$, so clearly $T \cup \Gamma \cup \Delta$ is not satisfiable.

The other possibility when $\mathrm{Sh}(\Gamma, \Delta, \mathit{canon}, \mathit{solve}) = \mathit{false}$ is that the algorithm terminates at line 6.

Suppose the loop has been executed n times and that Γ_n and $\mathcal{E}_n$ are the values of Γ and $\mathcal{E}$ at the end of the last loop.

It must be the case that $T \models a^* \neq b^*$, so $T \cup \{a^* = b^*\}$ is unsatisfiable.

Clearly then, $T \cup \{a^* = b^*\} \cup \mathcal{E}_n$ is unsatisfiable, so by **Lemma 1**, $T \cup \{a = b\} \cup \mathcal{E}_n$ is unsatisfiable.

But $\{a = b\}$ is a subset of Γ_n, so $T \cup \Gamma_n \cup \mathcal{E}_n$ must be unsatisfiable.

Thus by **Lemma 5**, $T \cup \Gamma$ is unsatisfiable.

Finally, suppose that $\mathtt{Sh}(\Gamma, \Delta, \mathit{canon}, \mathit{solve}) = \mathit{true}$. Then the algorithm terminates at line 10.

By **Lemma 4**, $\mathcal{E}$ is in solved form. Let $\overline{\Delta}$ be the disjunction of equalities equivalent to $\neg(\Delta)$.

Since the algorithm does not terminate at line 9, $T \cup \mathcal{E}$ does not entail any equality in $\overline{\Delta}$.

Because T is convex, it follows that $T \cup \mathcal{E} \not\models \overline{\Delta}$.

Now, since $T \cup \mathcal{E}$ is satisfiable by the corollary to the **Solved Form Theorem**, it follows that $T \cup \mathcal{E} \cup \Delta$ is satisfiable.

But by **Lemma 5**, $T \cup \Gamma \models \phi$ iff $T \cup \mathcal{E} \models \phi$, so in particular $T \cup \mathcal{E} \models \Gamma$.

Thus $T \cup \mathcal{E} \cup \Delta \cup \Gamma$ is satisfiable, and hence $T \cup \Gamma \cup \Delta$ is satisfiable.

Example

The most obvious example of a Shostak theory is $T_{\mathcal{R}}$ (without $\leq$).

- Step 1: Use the solver to convert Γ into an equisatisfiable set $\mathcal{E}$ of equations in solved form.

Γ	$\mathcal{E}$
$-x - 3y + 2z = 1$ $x - y - 6z = 1$ $2x + y - 10z = 3$	

Γ	$\mathcal{E}$
$-x - 3y + 2z = 1$ $x - y - 6z = 1$ $2x + y - 10z = 3$	

Γ	$\mathcal{E}$
$x = -3y + 2z + 1$ $x - y - 6z = 1$ $2x + y - 10z = 3$	

Γ	$\mathcal{E}$
$x - y - 6z = 1$ $2x + y - 10z = 3$	$x = -3y + 2z + 1$

Γ	$\mathcal{E}$
$x - y - 6z = 1$ $2x + y - 10z = 3$	$x = -3y + 2z + 1$

The most obvious example of a Shostak theory is $T_{\mathcal{R}}$ (without $\leq$).

- Step 1: Use the solver to convert Γ into an equisatisfiable set $\mathcal{E}$ of equations in solved form.

Γ	$\mathcal{E}$
$-4y - 4z + 1 = 1$ $2x + y - 10z = 3$	$x = -3y + 2z + 1$

Γ	$\mathcal{E}$
$y = -z$ $2x + y - 10z = 3$	$x = -3y + 2z + 1$

Γ	$\mathcal{E}$
$2x + y - 10z = 3$	$x = 5z + 1$ $y = -z$

Γ	$\mathcal{E}$
$2x + y - 10z = 3$	$x = 5z + 1$ $y = -z$

Γ	$\mathcal{E}$
$z = -1$	$x = 5z + 1$ $y = -z$

The most obvious example of a Shostak theory is $T_{\mathcal{R}}$ (without $\leq$).

- Step 1: Use the solver to convert Γ into an equisatisfiable set $\mathcal{E}$ of equations in solved form.

Γ	$\mathcal{E}$
$z = -1$	$x = 5(-1) + 1$ $y = -(-1)$

Γ	$\mathcal{E}$
	$x = -4$ $y = 1$ $z = -1$

Γ	$\mathcal{E}$
	$x = -4$ $y = 1$ $z = -1$

Note that for this theory, the main loop of Shostak's algorithm is equivalent to Gaussian elimination with back-substitution.

Example

The most obvious example of a Shostak theory is $T_{\mathcal{R}}$ (without $\leq$).

- Step 1: Use the solver to convert Γ into an equisatisfiable set $\mathcal{E}$ of equations in solved form.

- Step 2: Use $\mathcal{E}$ and *canon* to check if any disequality is violated:

  For each $a \neq b \in \Delta$, check if $\mathit{canon}(\mathcal{E}(a)) \equiv \mathit{canon}(\mathcal{E}(b))$.

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$x \neq 4y$ $x + w \neq w + z - 3y$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$x \neq 4y$ $x + w \neq w + z - 3y$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$-4 \neq 4(1)$ $x + w \neq w + z - 3y$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$-4 \neq 4$ $x + w \neq w + z - 3y$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$-4 \neq 4$ $x + w \neq w + z - 3y$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$-4 \neq 4$ $-4 + w \neq w + (-1) - 3(1)$

$\mathcal{E}$	Δ
$x = -4$ $y = 1$ $z = -1$	$-4 \neq 4$ $w + (-4) \neq w + (-4)$