3.17 Summary: Resolution Theorem Proving

- Resolution is a machine calculus.
- Subtle interleaving of enumerating instances and proving inconsistency through the use of unification.
- Parameters: atom ordering $\succ$ and selection function sel. On the non-ground level, ordering constraints can (only) be solved approximatively.
- Completeness proof by constructing candidate interpretations from productive clauses $C \vee A$, $A \succ C$.
- *Local* restrictions of inferences via $\succ$ and sel
  $\Rightarrow$ fewer proof variants.
- *Global* restrictions of the search space via elimination of redundancy
  $\Rightarrow$ computing with "smaller"/"easier" clause sets;
  $\Rightarrow$ termination on many decidable fragments.
- However: not good enough for dealing with orderings, equality and more specific algebraic theories (lattices, abelian groups, rings, fields)
  $\Rightarrow$ further specialization of inference systems required.

3.18 Semantic Tableaux

Literature:

M. Fitting: First-Order Logic and Automated Theorem Proving, Springer-Verlag, New York, 1996, chapters 3, 6, 7.

R. M. Smullyan: First-Order Logic, Dover Publ., New York, 1968, revised 1995.

Like resolution, semantic tableaux were developed in the sixties, independently by Zbigniew Lis and Raymond Smullyan on the basis of work by Gentzen in the 30s and of Beth in the 50s.

Idea

Idea (for the propositional case):

A set $\{F \wedge G\} \cup N$ of formulas has a model if and only if $\{F \wedge G,\ F,\ G\} \cup N$ has a model.

A set $\{F \vee G\} \cup N$ of formulas has a model if and only if $\{F \vee G,\ F\} \cup N$ or $\{F \vee G,\ G\} \cup N$ has a model.

(and similarly for other connectives).

To avoid duplication, represent sets as paths of a tree.

Continue splitting until two complementary formulas are found $\Rightarrow$ inconsistency detected.

A Tableau for $\{P \wedge \neg(Q \vee \neg R),\ \neg Q \vee \neg R\}$

1. $P \wedge \neg(Q \vee \neg R)$
2. $\neg Q \vee \neg R$

3. $\neg Q$	4. $\neg R$
5. P	10. P
6. $\neg(Q \vee \neg R)$	11. $\neg(Q \vee \neg R)$
7. $\neg Q$	
8. $\neg\neg R$	
9. R	

This tableau is not "maximal", however the first "path" is. This path is not "closed", hence the set $\{1, 2\}$ is satisfiable. (These notions will all be defined below.)

Properties

Properties of tableau calculi:

analytic: inferences correspond closely to the logical meaning of the symbols.

goal oriented: inferences operate directly on the goal to be proved (unlike, e. g., ordered resolution).

global: some inferences affect the entire proof state (set of formulas), as we will see later.

Propositional Expansion Rules

Expansion rules are applied to the formulas in a tableau and expand the tableau at a leaf. We append the conclusions of a rule (horizontally or vertically) at a *leaf*, whenever the premise of the expansion rule matches a formula appearing *anywhere* on the path from the root to that leaf.

Negation Elimination

$$\frac{\neg\neg F}{F} \qquad \frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top}$$

α-Expansion

(for formulas that are essentially conjunctions: append subformulas α_1 and α_2 one on top of the other)

$$\frac{\alpha}{\begin{array}{l}\alpha_1\\ \alpha_2\end{array}}$$

β-Expansion

(for formulas that are essentially disjunctions:
append β_1 and β_2 horizontally, i. e., branch into β_1 and β_2)

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Classification of Formulas

conjunctive			disjunctive		
α	α_1	α_2	β	β_1	β_2
$F \wedge G$	F	G	$\neg(F \wedge G)$	$\neg F$	$\neg G$
$\neg(F \vee G)$	$\neg F$	$\neg G$	$F \vee G$	F	G
$\neg(F \rightarrow G)$	F	$\neg G$	$F \rightarrow G$	$\neg F$	G

We assume that the binary connective $\leftrightarrow$ has been eliminated in advance.

Tableaux: Notions

A *semantic tableau* is a marked (by formulas), finite, unordered tree and inductively defined as follows: Let $\{F_1, \ldots, F_n\}$ be a set of formulas.

(i) The tree consisting of a single path

$$\begin{array}{c} F_1 \\ \vdots \\ F_n \end{array}$$

is a tableau for $\{F_1, \ldots, F_n\}$. (We do not draw edges if nodes have only one successor.)

(ii) If T is a tableau for $\{F_1, \ldots, F_n\}$ and if T' results from T by applying an expansion rule then T' is also a tableau for $\{F_1, \ldots, F_n\}$.

Note: We may also consider the *limit tableau* of a tableau expansion; this can be an *infinite* tree.

A *path* (from the root to a leaf) in a tableau is called *closed*, if it either contains $\bot$, or else it contains both some formula F and its negation $\neg F$. Otherwise the path is called *open*.

A tableau is called *closed*, if all paths are closed.

A *tableau proof* for F is a closed tableau for $\{\neg F\}$.

A path π in a tableau is called *maximal*, if for each formula F on π that is neither a literal nor $\bot$ nor $\top$ there exists a node in π at which the expansion rule for F has been applied.

In that case, if F is a formula on π, π also contains:

(i) α_1 and α_2, if F is a α-formula,

(ii) β_1 or β_2, if F is a β-formula, and

(iii) F', if F is a negation formula, and F' the conclusion of the corresponding elimination rule.

A tableau is called *maximal*, if each path is closed or maximal.

A tableau is called *strict*, if for each formula the corresponding expansion rule has been applied at most once on each path containing that formula.

A tableau is called *clausal*, if each of its formulas is a clause.

A Sample Proof

One starts out from the negation of the formula to be proved.

$$
\begin{array}{lcr}
1. & \neg\big((P \to (Q \to R)) \to ((P \vee S) \to ((Q \to R) \vee S))\big) & \\
2. & (P \to (Q \to R)) & [1_1] \\
3. & \neg((P \vee S) \to ((Q \to R) \vee S)) & [1_2] \\
4. & P \vee S & [3_1] \\
5. & \neg((Q \to R) \vee S)) & [3_2] \\
6. & \neg(Q \to R) & [5_1] \\
7. & \neg S & [5_2]
\end{array}
$$

$$
\begin{array}{ll}
8.\ \neg P\ [2_1] & 9.\ Q \to R\ [2_2] \\
\quad 10.\ P\ [4_1] \qquad 11.\ S\ [4_2] &
\end{array}
$$

There are three paths, each of them closed.

Properties of Propositional Tableaux

We assume that T is a tableau for $\{F_1, \ldots, F_n\}$.

Theorem 3.51 *$\{F_1, \ldots, F_n\}$ satisfiable $\Leftrightarrow$ some path (i. e., the set of its formulas) in T is satisfiable.*

Proof. ($\Leftarrow$) Trivial, since every path contains in particular $F_1, \ldots, F_n$.
($\Rightarrow$) By induction over the structure of T. □

Corollary 3.52 *T closed $\Rightarrow$ $\{F_1, \ldots, F_n\}$ unsatisfiable*

Theorem 3.53 *Every strict propositional tableau expansion is finite.*

Proof. New formulas resulting from expansion are either $\bot$, $\top$ or subformulas of the expanded formula (modulo de Morgan's law), so the number of formulas that can occur is finite. By strictness, on each path a formula can be expanded at most once. Therefore, each path is finite, and a finitely branching tree with finite paths is finite by Lemma 1.9. □

Conclusion: Strict and maximal tableaux can be effectively constructed.

Refutational Completeness

A set $\mathcal{H}$ of propositional formulas is called a Hintikka set, if

(1) there is no $P \in \Pi$ with $P \in \mathcal{H}$ and $\neg P \in \mathcal{H}$;

(2) $\bot \notin \mathcal{H}$, $\neg\top \notin \mathcal{H}$;

(3) if $\neg\neg F \in \mathcal{H}$, then $F \in \mathcal{H}$;

(4) if $\alpha \in \mathcal{H}$, then $\alpha_1 \in \mathcal{H}$ and $\alpha_2 \in \mathcal{H}$;

(5) if $\beta \in \mathcal{H}$, then $\beta_1 \in \mathcal{H}$ or $\beta_2 \in \mathcal{H}$.

Lemma 3.54 (Hintikka's Lemma) *Every Hintikka set is satisfiable.*

Proof. Let $\mathcal{H}$ be a Hintikka set. Define a valuation $\mathcal{A}$ by $\mathcal{A}(P) = 1$ if $P \in \mathcal{H}$ and $\mathcal{A}(P) = 0$ otherwise. Then show that $\mathcal{A}(F) = 1$ for all $F \in \mathcal{H}$ by induction over the size of formulas. $\square$

Theorem 3.55 *Let π be a maximal open path in a tableau. Then the set of formulas on π is satisfiable.*

Proof. We show that set of formulas on π is a Hintikka set: Conditions (3), (4), (5) follow from the fact that π is maximal; conditions (1) and (2) follow from the fact that π is open and from maximality for the second negation elimination rule. $\square$

Note: The theorem holds also for infinite trees that are obtained as the limit of a tableau expansion.

Theorem 3.56 $\{F_1, \dots, F_n\}$ *satisfiable* $\Leftrightarrow$ *there exists no closed strict tableau for* $\{F_1, \dots, F_n\}$.

Proof. ($\Rightarrow$) Clear by Cor. 3.52.
($\Leftarrow$) Let T be a strict maximal tableau for $\{F_1, \dots, F_n\}$ and let π be an open path in T. By the previous theorem, the set of formulas on π is satisfiable, and hence by Theorem 3.51 the set $\{F_1, \dots, F_n\}$, is satisfiable. $\square$

Consequences

The validity of a propositional formula F can be established by constructing a strict maximal tableau for $\{\neg F\}$:

- T closed $\Leftrightarrow$ F valid.
- It suffices to test complementarity of paths w.r.t. atomic formulas (cf. reasoning in the proof of Theorem 3.55).
- Which of the potentially many strict maximal tableaux one computes does not matter. In other words, tableau expansion rules can be applied don't-care non-deterministically ("*proof confluence*").
- The expansion strategy, however, can have a dramatic impact on the tableau size.

A Variant of the β-Rule

Since $F \vee G \models\!\!\!\mid F \vee (G \wedge \neg F)$, the β expansion rule

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

can be replaced by the following variant:

$$\frac{\beta}{\beta_1 \;\Big|\; \begin{array}{l}\beta_2 \\ \neg\beta_1\end{array}}$$

The variant β-rule can lead to much shorter proofs, but it is not always beneficial.

In general, it is most helpful if $\neg\beta_1$ can be at most (iteratively) α-expanded.

3.19 Semantic Tableaux for First-Order Logic

There are two ways to extend the tableau calculus to quantified formulas:

- using ground instantiation,
- using free variables.

Tableaux with Ground Instantiation

Classification of quantified formulas:

universal		existential	
γ	$\gamma(t)$	δ	$\delta(t)$
$\forall x F$	$F\{x \mapsto t\}$	$\exists x F$	$F\{x \mapsto t\}$
$\neg\exists x F$	$\neg F\{x \mapsto t\}$	$\neg\forall x F$	$\neg F\{x \mapsto t\}$

Idea:

Replace universally quantified formulas by appropriate ground instances.

γ-expansion

$$\frac{\gamma}{\gamma(t)} \quad \text{where } t \text{ is some ground term}$$

δ-expansion

$$\frac{\delta}{\delta(c)} \quad \text{where } c \text{ is a new Skolem constant}$$

Skolemization becomes part of the calculus and needs not necessarily be applied in a preprocessing step. Of course, one could do Skolemization beforehand, and then the δ-rule would not be needed.

Note:

Skolem *constants* are sufficient:
In a δ-formula $\exists x\, F$, $\exists$ is the outermost quantifier and x is the only free variable in F.

Problems:

Having to guess ground terms is impractical.

Even worse, we may have to guess *several* ground instances, as strictness for γ is incomplete. For instance, constructing a closed tableau for

$$\{\forall x\,(P(x) \rightarrow P(f(x))),\ P(b),\ \neg P(f(f(b)))\}$$

is impossible without applying γ-expansion twice on one path.

Free-Variable Tableaux

An alternative approach:

Delay the instantiation of universally quantified variables.

Replace universally quantified variables by new free variables.

Intuitively, the free variables are universally quantified *outside* of the entire tableau.

γ-expansion

$$\frac{\gamma}{\gamma(x)} \quad \text{where } x \text{ is a new free variable}$$

δ-expansion

$$\frac{\delta}{\delta(f(x_1, \dots, x_n))}$$

where f is a new Skolem function, and the x_i are the free variables in δ

Application of expansion rules has to be supplemented by a *substitution rule:*

(iii) If T is a tableau for $\{F_1, \dots, F_n\}$ and if σ is a substitution, then $T\sigma$ is also a tableau for $\{F_1, \dots, F_n\}$.

The substitution rule may, potentially, modify all the formulas of a tableau. This feature is what makes the tableau method a *global proof method.* (Resolution, by comparison, is a local method.)

One can show that it is sufficient to consider substitutions σ for which there is a path in T containing two *literals* $\neg A$ and B such that $\sigma = \mathrm{mgu}(A, B)$. Such tableaux are called *AMGU-Tableaux*.

Example

1.	$\neg\big(\exists w \forall x\ P(x,w,f(x,w)) \rightarrow \exists w \forall x \exists y\ P(x,w,y)\big)$	
2.	$\exists w \forall x\ P(x,w,f(x,w))$	1_1 $[\alpha]$
3.	$\neg\exists w \forall x \exists y\ P(x,w,y)$	1_2 $[\alpha]$
4.	$\forall x\ P(x,c,f(x,c))$	$2(c)$ $[\delta]$
5.	$\neg\forall x \exists y\ P(x,v_1,y)$	$3(v_1)$ $[\gamma]$
6.	$\neg\exists y\ P(b(v_1),v_1,y)$	$5(b(v_1))$ $[\delta]$
7.	$P(v_2,c,f(v_2,c))$	$4(v_2)$ $[\gamma]$
8.	$\neg P(b(v_1),v_1,v_3)$	$6(v_3)$ $[\gamma]$

7. and 8. are complementary (modulo unification):

$$\{v_2 \doteq b(v_1),\ c \doteq v_1,\ f(v_2,c) \doteq v_3\}$$

is solvable with an mgu $\sigma = \{v_1 \mapsto c,\ v_2 \mapsto b(c),\ v_3 \mapsto f(b(c),c)\}$, and hence, $T\sigma$ is a closed (linear) tableau for the formula in 1.

Problem:

Strictness for γ is still incomplete. For instance, constructing a closed tableau for

$$\{\forall x\,(P(x) \rightarrow P(f(x))),\ P(b),\ \neg P(f(f(b)))\}$$

is impossible without applying γ-expansion twice on one path.

Semantic Tableaux vs. Resolution

- Tableaux: global, goal-oriented, “backward”.
- Resolution: local, “forward”.
- Goal-orientation is a clear advantage if only a small subset of a large set of formulas is necessary for a proof. (Note that resolution provers saturate also those parts of the clause set that are irrelevant for proving the goal.)
- Resolution can be combined with more powerful redundancy elimination methods; because of its global nature this is more difficult for the tableau method.
- Resolution can be refined to work well with equality; for tableaux this seems to be impossible.
- On the other hand tableau calculi can be easily extended to other logics; in particular tableau provers are very successful in modal and description logics.

3.20 Other Deductive Systems

- Instantiation-based methods
  Resolution-based instance generation
  Disconnection calculus
  ...
- Natural deduction
- Sequent calculus/Gentzen calculus
- Hilbert calculus

Instantiation-Based Methods for FOL

Idea:

Overlaps of complementary literals produce instantiations (as in resolution);

However, contrary to resolution, clauses are not recombined.

Instead: treat remaining variables as constant and use efficient propositional proof methods, such as CDCL.

There are both saturation-based variants, such as partial instantiation (Hooker et al. 2002) or resolution-based instance generation (Inst-Gen) (Ganzinger and Korovin 2003), and tableau-style variants, such as the disconnection calculus (Billon 1996; Letz and Stenz 2001).

Successful in practice for problems that are "almost propositional" (i. e., no non-constant function symbols, no equality).

Natural Deduction

Idea:

Model the concept of proofs from assumptions as humans do it.

To prove $F \rightarrow G$, assume F and try to derive G.

Initial ideas: Jaśkowski (1934), Gentzen (1934); extended by Prawitz (1965).

Popular in interactive proof systems.

Sequent Calculus

Idea:

Assumptions internalized into the data structure of sequents

$$F_1, \dots, F_m \vdash G_1, \dots, G_k$$

meaning

$$F_1 \wedge \dots \wedge F_m \rightarrow G_1 \vee \dots \vee G_k$$

Inferences rules, e.g.:

$$\frac{\Gamma \vdash \Delta}{\Gamma, F \vdash \Delta}\ (WL) \qquad \frac{\Gamma, F \vdash \Delta \qquad \Sigma, G \vdash \Pi}{\Gamma, \Sigma, F \vee G \vdash \Delta, \Pi}\ (\vee L)$$

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash F, \Delta}\ (WR) \qquad \frac{\Gamma \vdash F, \Delta \qquad \Sigma \vdash G, \Pi}{\Gamma, \Sigma \vdash F \wedge G, \Delta, \Pi}\ (\wedge R)$$

Initial idea: Gentzen 1934.

Perfect symmetry between the handling of assumptions and their consequences; interesting for proof theory.

Can be used both backwards and forwards.

Allows to simulate both natural deduction and semantic tableaux.

Hilbert Calculus

Idea:

Direct proof method (proves a theorem from axioms, rather than refuting its negation)

Axiom schemes, e. g.,

$$F \rightarrow (G \rightarrow F)$$
$$(F \rightarrow (G \rightarrow H)) \rightarrow ((F \rightarrow G) \rightarrow (F \rightarrow H))$$

plus Modus ponens:

$$\frac{F \qquad F \rightarrow G}{G}$$

Unsuitable for finding or reading proofs, but sometimes used for *specifying* (e.g. modal) logics.

4 First-Order Logic with Equality

Equality is the most important relation in mathematics and functional programming.

In principle, problems in first-order logic with equality can be handled by any prover for first-order logic without equality:

4.1 Handling Equality Naively

Proposition 4.1 *Let F be a closed first-order formula with equality. Let $\sim \notin \Pi$ be a new predicate symbol. The set $Eq(\Sigma)$ contains the formulas*

$$\begin{gathered}
\forall x\,(x \sim x)\\
\forall x, y\,(x \sim y \rightarrow y \sim x)\\
\forall x, y, z\,(x \sim y \wedge y \sim z \rightarrow x \sim z)\\
\forall \vec{x}, \vec{y}\,(x_1 \sim y_1 \wedge \cdots \wedge x_n \sim y_n \rightarrow f(x_1, \dots, x_n) \sim f(y_1, \dots, y_n))\\
\forall \vec{x}, \vec{y}\,(x_1 \sim y_1 \wedge \cdots \wedge x_m \sim y_m \wedge P(x_1, \dots, x_m) \rightarrow P(y_1, \dots, y_m))
\end{gathered}$$

for every $f/n \in \Omega$ and $P/m \in \Pi$. Let $\tilde{F}$ be the formula that one obtains from F if every occurrence of $\approx$ is replaced by $\sim$. Then F is satisfiable if and only if $Eq(\Sigma) \cup \{\tilde{F}\}$ is satisfiable.

Proof. Let $\Sigma = (\Omega, \Pi)$, let $\Sigma_1 = (\Omega, \Pi \cup \{\sim/2\})$.

For the "only if" part assume that F is satisfiable and let $\mathcal{A}$ be a Σ-model of F. Then we define a Σ_1-algebra $\mathcal{B}$ in such a way that $\mathcal{B}$ and $\mathcal{A}$ have the same universe, $f_{\mathcal{B}} = f_{\mathcal{A}}$ for every $f \in \Omega$, $P_{\mathcal{B}} = P_{\mathcal{A}}$ for every $P \in \Pi$, and $\sim_{\mathcal{B}}$ is the identity relation on the universe. It is easy to check that $\mathcal{B}$ is a model of both $\tilde{F}$ and of $Eq(\Sigma)$.

For the "if" part assume that the Σ_1-algebra $\mathcal{B} = (U_{\mathcal{B}},\ (f_{\mathcal{B}} : U_{\mathcal{B}}^n \to U_{\mathcal{B}})_{f \in \Omega},\ (P_{\mathcal{B}} \subseteq U_{\mathcal{B}}^m)_{P \in \Pi \cup \{\sim\}})$ is a model of $Eq(\Sigma) \cup \{\tilde{F}\}$. Then the interpretation $\sim_{\mathcal{B}}$ of $\sim$ in $\mathcal{B}$ is a congruence relation on $U_{\mathcal{B}}$ with respect to the functions $f_{\mathcal{B}}$ and the predicates $P_{\mathcal{B}}$.

We will now construct a Σ-algebra $\mathcal{A}$ from $\mathcal{B}$ and the congruence relation $\sim_{\mathcal{B}}$. Let $[a]$ be the congruence class of an element $a \in U_{\mathcal{B}}$ with respect to $\sim_{\mathcal{B}}$. The universe $U_{\mathcal{A}}$ of $\mathcal{A}$ is the set $\{\, [a] \mid a \in U_{\mathcal{B}} \,\}$ of congruence classes of the universe of $\mathcal{B}$. For a function symbol $f \in \Omega$, we define $f_{\mathcal{A}}([a_1], \dots, [a_n]) = [f_{\mathcal{B}}(a_1, \dots, a_n)]$, and for a predicate symbol $P \in \Pi$, we define $([a_1], \dots, [a_n]) \in P_{\mathcal{A}}$ if and only if $(a_1, \dots, a_n) \in P_{\mathcal{B}}$. Observe that this is well-defined: If we take different representatives of the same congruence class, we get the same result by congruence of $\sim_{\mathcal{B}}$. For any $\mathcal{A}$-assignment γ choose some $\mathcal{B}$-assignment β such that $\mathcal{B}(\beta)(x) \in \mathcal{A}(\gamma)(x)$ for every x, then for every Σ-term t we have $\mathcal{A}(\gamma)(t) = [\mathcal{B}(\beta)(t)]$, and analogously for every Σ-formula G, $\mathcal{A}(\gamma)(G) = \mathcal{B}(\beta)(\tilde{G})$. Both properties can easily shown by structural induction. Therefore, $\mathcal{A}$ is a model of F. $\square$

An analogous proposition holds for *sets* of closed first-order formulas with equality.

By giving the equality axioms explicitly, first-order problems with equality can in principle be solved by a standard resolution or tableaux prover.

But this is unfortunately not efficient (mainly due to the transitivity and congruence axioms).

Equality is theoretically difficult: First-order functional programming is Turing-complete.

But: resolution theorem provers cannot even solve equational problems that are intuitively easy.

Consequence: to handle equality efficiently, knowledge must be integrated into the theorem prover.

Roadmap

How to proceed:

- This semester: Equations (unit clauses with equality)

  Term rewrite systems
  Expressing semantic consequence syntactically
  Knuth-Bendix-Completion
  Entailment for equations

- Next semester: Equational clauses

  Combining resolution and KB-completion → Superposition
  Entailment for clauses with equality

4.2 Rewrite Systems

Let E be a set of (implicitly universally quantified) equations.

The *rewrite relation* $\rightarrow_E \subseteq \mathrm{T}_\Sigma(X) \times \mathrm{T}_\Sigma(X)$ is defined by

$$
\begin{array}{lll}
s \rightarrow_E t & \text{iff} & \text{there exist } (l \approx r) \in E,\ p \in \mathrm{pos}(s), \\
 & & \text{and } \sigma : X \rightarrow \mathrm{T}_\Sigma(X), \\
 & & \text{such that } s|_p = l\sigma \text{ and } t = s[r\sigma]_p.
\end{array}
$$

An instance of the lhs (left-hand side) of an equation is called a *redex* (reducible expression). *Contracting* a redex means replacing it with the corresponding instance of the rhs (right-hand side) of the rule.

An equation $l \approx r$ is also called a *rewrite rule*, if l is not a variable and $\mathrm{var}(l) \supseteq \mathrm{var}(r)$.

Notation: $l \to r$.

A set of rewrite rules is called a *term rewrite system (TRS)*.

We say that a set of equations E or a TRS R is terminating, if the rewrite relation $\to_E$ or $\to_R$ has this property.

(Analogously for other properties of abstract reduction systems).

Note: If E is terminating, then it is a TRS.

E-Algebras

Let E be a set of universally quantified equations. A model of E is also called an *E-algebra*.

If $E \models \forall\vec{x}(s \approx t)$, i. e., $\forall\vec{x}(s \approx t)$ is valid in all E-algebras, we write this also as $s \approx_E t$.

Goal:
Use the rewrite relation $\to_E$ to express the semantic consequence relation syntactically:

$s \approx_E t$ if and only if $s \leftrightarrow_E^* t$.

Let E be a set of equations over $\mathrm{T}_\Sigma(X)$. The following inference system allows to derive consequences of E:

$E \vdash t \approx t$ for every $t \in \mathrm{T}_\Sigma(X)$ *(Reflexivity)*

$$\frac{E \vdash t \approx t'}{E \vdash t' \approx t} \qquad \textit{(Symmetry)}$$

$$\frac{E \vdash t \approx t' \qquad E \vdash t' \approx t''}{E \vdash t \approx t''} \qquad \textit{(Transitivity)}$$

$$\frac{E \vdash t_1 \approx t_1' \quad \dots \quad E \vdash t_n \approx t_n'}{E \vdash f(t_1, \dots, t_n) \approx f(t_1', \dots, t_n')} \qquad \textit{(Congruence)}$$

$E \vdash t\sigma \approx t'\sigma$ if $(t \approx t') \in E$ and $\sigma : X \to \mathrm{T}_\Sigma(X)$ *(Instance)*

Lemma 4.2 *The following properties are equivalent:*

(i) $s \leftrightarrow_E^* t$

(ii) $E \vdash s \approx t$ *is derivable.*

Proof. (i)⇒(ii): $s \leftrightarrow_E t$ implies $E \vdash s \approx t$ by induction on the depth of the position where the equation is applied; then $s \leftrightarrow_E^* t$ implies $E \vdash s \approx t$ by induction on the number of rewrite steps in $s \leftrightarrow_E^* t$.

(ii)⇒(i): By induction on the size (number of symbols) of the derivation for $E \vdash s \approx t$. □

Constructing a *quotient algebra:*

Let X be a set of variables.

For $t \in \mathrm{T}_\Sigma(X)$ let $[t] = \{\, t' \in \mathrm{T}_\Sigma(X) \mid E \vdash t \approx t' \,\}$ be the *congruence class* of t.

Define a Σ-algebra $\mathrm{T}_\Sigma(X)/E$ (abbreviated by $\mathcal{T}$) as follows:

$U_\mathcal{T} = \{\, [t] \mid t \in \mathrm{T}_\Sigma(X) \,\}$.

$f_\mathcal{T}([t_1], \dots, [t_n]) = [f(t_1, \dots, t_n)]$ for $f/n \in \Omega$.

Lemma 4.3 $f_\mathcal{T}$ *is well-defined: If* $[t_i] = [t_i']$, *then* $[f(t_1, \dots, t_n)] = [f(t_1', \dots, t_n')]$.

Proof. Follows directly from the *Congruence* rule for $\vdash$. □

Lemma 4.4 $\mathcal{T} = \mathrm{T}_\Sigma(X)/E$ *is an E-algebra.*

Proof. Let $\forall x_1 \dots x_n (s \approx t)$ be an equation in E; let β be an arbitrary assignment.

We have to show that $\mathcal{T}(\beta)(\forall \vec{x}(s \approx t)) = 1$, or equivalently, that $\mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t)$ for all $\gamma = \beta[\, x_i \mapsto [v_i] \mid 1 \leq i \leq n \,]$ with $[v_i] \in U_\mathcal{T}$.

Let $\sigma = \{x_1 \mapsto v_1, \dots, x_n \mapsto v_n\}$, then we get by structural induction that $u\sigma \in \mathcal{T}(\gamma)(u)$ for every $u \in \mathrm{T}_\Sigma(\{x_1, ..., x_n\})$. In particular, $s\sigma \in \mathcal{T}(\gamma)(s)$ and $t\sigma \in \mathcal{T}(\gamma)(t)$.

By the *Instance* rule, $E \vdash s\sigma \approx t\sigma$ is derivable, hence $\mathcal{T}(\gamma)(s) = [s\sigma] = [t\sigma] = \mathcal{T}(\gamma)(t)$. □

Lemma 4.5 *Let X be a countably infinite set of variables; let $s, t \in \mathrm{T}_\Sigma(Y)$. If $\mathrm{T}_\Sigma(X)/E \models \forall\vec{x}(s \approx t)$, then $E \vdash s \approx t$ is derivable.*

Proof. Without loss of generality, we assume that all variables in $\vec{x}$ are contained in X. (Otherwise, we rename the variables in the equation. Since X is countably infinite, this is always possible.) Assume that $\mathcal{T} \models \forall\vec{x}(s \approx t)$, i. e., $\mathcal{T}(\beta)(\forall\vec{x}(s \approx t)) = 1$. Consequently, $\mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t)$ for all $\gamma = \beta[\, x_i \mapsto [v_i] \mid 1 \leq i \leq n\,]$ with $[v_i] \in U_{\mathcal{T}}$.

Choose $v_i := x_i$, then by structural induction $[u] = \mathcal{T}(\gamma)(u)$ for every $u \in \mathrm{T}_\Sigma(\{x_1, ..., x_n\})$, so $[s] = \mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t) = [t]$. Therefore $E \vdash s \approx t$ is derivable by definition of $\mathcal{T}$. $\square$

Theorem 4.6 ("Birkhoff's Theorem") *Let X be a countably infinite set of variables, let E be a set of (universally quantified) equations. Then the following properties are equivalent for all $s, t \in \mathrm{T}_\Sigma(X)$:*

(i) $s \leftrightarrow_E^* t$.

(ii) $E \vdash s \approx t$ *is derivable.*

(iii) $s \approx_E t$, *i. e.,* $E \models \forall\vec{x}(s \approx t)$.

(iv) $\mathrm{T}_\Sigma(X)/E \models \forall\vec{x}(s \approx t)$.

Proof. (i)⇔(ii): Lemma 4.2.

(ii)⇒(iii): By induction on the size of the derivation for $E \vdash s \approx t$.

(iii)⇒(iv): Obvious, since $\mathcal{T} = \mathrm{T}_\Sigma(X)/E$ is an E-algebra.

(iv)⇒(ii): Lemma 4.5. $\square$

Universal Algebra

$\mathrm{T}_\Sigma(X)/E = \mathrm{T}_\Sigma(X)/{\approx_E} = \mathrm{T}_\Sigma(X)/{\leftrightarrow_E^*}$ is called the *free E-algebra* with generating set $X/{\approx_E} = \{\, [x] \mid x \in X \,\}$:

> Every mapping $\varphi : X/{\approx_E} \to \mathcal{B}$ for some E-algebra $\mathcal{B}$ can be extended to a homomorphism $\hat{\varphi} : \mathrm{T}_\Sigma(X)/E \to \mathcal{B}$.

$\mathrm{T}_\Sigma(\emptyset)/E = \mathrm{T}_\Sigma(\emptyset)/{\approx_E} = \mathrm{T}_\Sigma(\emptyset)/{\leftrightarrow_E^*}$ is called the *initial E-algebra*.

${\approx_E} = \{\, (s,t) \mid E \models s \approx t \,\}$ is called the *equational theory* of E.

${\approx_E^I} = \{\, (s,t) \mid \mathrm{T}_\Sigma(\emptyset)/E \models s \approx t \,\}$ is called the *inductive theory* of E.

Example:

> Let $E = \{\forall x(x + 0 \approx x),\ \forall x \forall y(x + s(y) \approx s(x + y))\}$. Then $x + y \approx_E^I y + x$, but $x + y \not\approx_E y + x$.

4.3 Confluence

Let $(A, \to)$ be an abstract reduction system.

b and $c \in A$ are *joinable*, if there is a a such that $b \to^* a \leftarrow^* c$.
Notation: $b \downarrow c$.

The relation $\to$ is called

Church-Rosser, if $b \leftrightarrow^* c$ implies $b \downarrow c$.

confluent, if $b \leftarrow^* a \to^* c$ implies $b \downarrow c$.

locally confluent, if $b \leftarrow a \to c$ implies $b \downarrow c$.

convergent, if it is confluent and terminating.

Theorem 4.7 *The following properties are equivalent:*

(i) $\to$ *has the Church-Rosser property.*

(ii) $\to$ *is confluent.*

Proof. (i)⇒(ii): trivial.

(ii)⇒(i): by induction on the number of peaks in the derivation $b \leftrightarrow^* c$. □

Lemma 4.8 *If* $\to$ *is confluent, then every element has at most one normal form.*

Proof. Suppose that some element $a \in A$ has normal forms b and c, then $b \leftarrow^* a \to^* c$. If $\to$ is confluent, then $b \to^* d \leftarrow^* c$ for some $d \in A$. Since b and c are normal forms, both derivations must be empty, hence $b \to^0 d \leftarrow^0 c$, so b, c, and d must be identical. □

Corollary 4.9 *If* $\to$ *is normalizing and confluent, then every element* b *has a unique normal form.*

Proposition 4.10 *If* $\to$ *is normalizing and confluent, then* $b \leftrightarrow^* c$ *if and only if* $b{\downarrow} = c{\downarrow}$.

Proof. Either using Thm. 4.7 or directly by induction on the length of the derivation of $b \leftrightarrow^* c$. □

Confluence and Local Confluence

Theorem 4.11 ("Newman's Lemma") *If a terminating relation $\rightarrow$ is locally confluent, then it is confluent.*

Proof. Let $\rightarrow$ be a terminating and locally confluent relation. Then $\rightarrow^{+}$ is a well-founded ordering. Define $\phi(a) \;\Leftrightarrow\; \big(\forall b, c : b \leftarrow^{*} a \rightarrow^{*} c \Rightarrow b \downarrow c\big)$.

We prove $\phi(a)$ for all $a \in A$ by well-founded induction over $\rightarrow^{+}$:

Case 1: $b \leftarrow^{0} a \rightarrow^{*} c$: trivial.

Case 2: $b \leftarrow^{*} a \rightarrow^{0} c$: trivial.

Case 3: $b \leftarrow^{*} b' \leftarrow a \rightarrow c' \rightarrow^{*} c$: use local confluence, then use the induction hypothesis. $\square$

Rewrite Relations

Corollary 4.12 *If E is convergent (i. e., terminating and confluent), then $s \approx_E t$ if and only if $s \leftrightarrow_E^{*} t$ if and only if $s{\downarrow_E} = t{\downarrow_E}$.*

Corollary 4.13 *If E is finite and convergent, then $\approx_E$ is decidable.*

Reminder:
If E is terminating, then it is confluent if and only if it is locally confluent.

Problems:

Show local confluence of E.

Show termination of E.

Transform E into an equivalent set of equations that is locally confluent and terminating.

4.4 Critical Pairs

Showing local confluence (Sketch):

Problem: If $t_1 \leftarrow_E t_0 \rightarrow_E t_2$, does there exist a term s such that $t_1 \rightarrow_E^* s \leftarrow_E^* t_2$?

If the two rewrite steps happen in different subtrees (disjoint redexes): yes.

If the two rewrite steps happen below each other (overlap at or below a variable position): yes.

If the left-hand sides of the two rules overlap at a non-variable position: needs further investigation.

Question:
Are there rewrite rules $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ such that some subterm $l_1|_p$ and l_2 have a common instance $(l_1|_p)\sigma_1 = l_2\sigma_2$?

Observation:
If we assume w.l.o.g. that the two rewrite rules do not have common variables, then only a single substitution is necessary: $(l_1|_p)\sigma = l_2\sigma$.

Further observation:
The mgu of $l_1|_p$ and l_2 subsumes all unifiers σ of $l_1|_p$ and l_2.

Let $l_i \rightarrow r_i$ $(i = 1, 2)$ be two rewrite rules in a TRS R whose variables have been renamed such that $\text{var}(l_1) \cap \text{var}(l_2) = \emptyset$. (Remember that $\text{var}(l_i) \supseteq \text{var}(r_i)$.)

Let $p \in \text{pos}(l_1)$ be a position such that $l_1|_p$ is not a variable and σ is an mgu of $l_1|_p$ and l_2.

Then $r_1\sigma \leftarrow l_1\sigma \rightarrow (l_1\sigma)[r_2\sigma]_p$.

$\langle r_1\sigma, (l_1\sigma)[r_2\sigma]_p\rangle$ is called a *critical pair* of R.

The critical pair is *joinable* (or: converges), if $r_1\sigma \downarrow_R (l_1\sigma)[r_2\sigma]_p$.

Theorem 4.14 ("Critical Pair Theorem") *A TRS R is locally confluent if and only if all its critical pairs are joinable.*

Proof. "only if": obvious, since joinability of a critical pair is a special case of local confluence.

"if": Suppose s rewrites to t_1 and t_2 using rewrite rules $l_i \rightarrow r_i \in R$ at positions $p_i \in \text{pos}(s)$, where $i = 1, 2$. Without loss of generality, we can assume that the two rules are variable disjoint, hence $s|_{p_i} = l_i\theta$ and $t_i = s[r_i\theta]_{p_i}$.

We distinguish between two cases: Either p_1 and p_2 are in disjoint subtrees ($p_1 \parallel p_2$), or one is a prefix of the other (w.l.o.g., $p_1 \leq p_2$).

Case 1: $p_1 \parallel p_2$.

Then $s = s[l_1\theta]_{p_1}[l_2\theta]_{p_2}$, and therefore $t_1 = s[r_1\theta]_{p_1}[l_2\theta]_{p_2}$ and $t_2 = s[l_1\theta]_{p_1}[r_2\theta]_{p_2}$.

Let $t_0 = s[r_1\theta]_{p_1}[r_2\theta]_{p_2}$. Then clearly $t_1 \to_R t_0$ using $l_2 \to r_2$ and $t_2 \to_R t_0$ using $l_1 \to r_1$.

Case 2: $p_1 \leq p_2$.

Case 2.1: $p_2 = p_1 q_1 q_2$, where $l_1|_{q_1}$ is some variable x.

In other words, the second rewrite step takes place at or below a variable in the first rule. Suppose that x occurs m times in l_1 and n times in r_1 (where $m \geq 1$ and $n \geq 0$).

Then $t_1 \to_R^* t_0$ by applying $l_2 \to r_2$ at all positions $p_1 q' q_2$, where q' is a position of x in r_1.

Conversely, $t_2 \to_R^* t_0$ by applying $l_2 \to r_2$ at all positions $p_1 q q_2$, where q is a position of x in l_1 different from q_1, and by applying $l_1 \to r_1$ at p_1 with the substitution θ', where $\theta' = \theta[x \mapsto (x\theta)[r_2\theta]_{q_2}]$.

Case 2.2: $p_2 = p_1 p$, where p is a non-variable position of l_1.

Then $s|_{p_2} = l_2\theta$ and $s|_{p_2} = (s|_{p_1})|_p = (l_1\theta)|_p = (l_1|_p)\theta$, so θ is a unifier of l_2 and $l_1|_p$.

Let σ be the mgu of l_2 and $l_1|_p$, then $\theta = \tau \circ \sigma$ and $\langle r_1\sigma, (l_1\sigma)[r_2\sigma]_p\rangle$ is a critical pair.

By assumption, it is joinable, so $r_1\sigma \to_R^* v \leftarrow_R^* (l_1\sigma)[r_2\sigma]_p$.

Consequently, $t_1 = s[r_1\theta]_{p_1} = s[r_1\sigma\tau]_{p_1} \to_R^* s[v\tau]_{p_1}$ and $t_2 = s[r_2\theta]_{p_2} = s[(l_1\theta)[r_2\theta]_p]_{p_1} = s[(l_1\sigma\tau)[r_2\sigma\tau]_p]_{p_1} = s[((l_1\sigma)[r_2\sigma]_p)\tau]_{p_1} \to_R^* s[v\tau]_{p_1}$.

This completes the proof of the Critical Pair Theorem. □

Note: Critical pairs between a rule and (a renamed variant of) itself must be considered – except if the overlap is at the root (i. e., $p = \varepsilon$).

Corollary 4.15 *A terminating TRS R is confluent if and only if all its critical pairs are joinable.*

Proof. By Newman's Lemma and the Critical Pair Theorem. □

Corollary 4.16 *For a finite terminating TRS, confluence is decidable.*

Proof. For every pair of rules and every non-variable position in the first rule there is at most one critical pair $\langle u_1, u_2\rangle$.

Reduce every u_i to some normal form u_i'. If $u_1' = u_2'$ for every critical pair, then R is confluent, otherwise there is some non-confluent situation $u_1' \leftarrow_R^* u_1 \leftarrow_R s \to_R u_2 \to_R^* u_2'$. □

4.5 Termination

Termination problems:

Given a finite TRS R and a term t, are all R-reductions starting from t terminating?

Given a finite TRS R, are all R-reductions terminating?

Proposition 4.17 *Both termination problems for TRSs are undecidable in general.*

Proof. Encode Turing machines using rewrite rules and reduce the (uniform) halting problems for TMs to the termination problems for TRSs. □

Consequence:

Decidable criteria for termination are not complete.

Two Different Scenarios

Depending on the application, the TRS whose termination we want to show can be

(i) fixed and known in advance, or

(ii) evolving (e.g., generated by some saturation process).

Methods for case (ii) are also usable for case (i).
Many methods for case (i) are not usable for case (ii).

We will first consider case (ii);
additional techniques for case (i) will be considered later.

Reduction Orderings

Goal:

Given a finite TRS R, show termination of R by looking at finitely many rules $l \to r \in R$, rather than at infinitely many possible replacement steps $s \to_R s'$.

A binary relation $\sqsupset$ over $\mathrm{T}_\Sigma(X)$ is called *compatible with* Σ*-operations*, if $s \sqsupset s'$ implies $f(t_1, \dots, s, \dots, t_n) \sqsupset f(t_1, \dots, s', \dots, t_n)$ for all $f \in \Omega$ and $s, s', t_i \in \mathrm{T}_\Sigma(X)$.

Lemma 4.18 *The relation $\sqsupset$ is compatible with Σ-operations, if and only if $s \sqsupset s'$ implies $t[s]_p \sqsupset t[s']_p$ for all $s, s', t \in \mathrm{T}_\Sigma(X)$ and $p \in \mathrm{pos}(t)$.*

Note: *compatible with Σ-operations* = *compatible with contexts*.

A binary relation $\sqsupset$ over $T_\Sigma(X)$ is called *stable under substitutions*, if $s \sqsupset s'$ implies $s\sigma \sqsupset s'\sigma$ for all $s, s' \in T_\Sigma(X)$ and substitutions σ.

A binary relation $\sqsupset$ is called a *rewrite relation*, if it is compatible with Σ-operations and stable under substitutions.

Example: If R is a TRS, then $\to_R$ is a rewrite relation.

A strict partial ordering over $T_\Sigma(X)$ that is a rewrite relation is called *rewrite ordering*.

A well-founded rewrite ordering is called *reduction ordering*.

Theorem 4.19 *A TRS R terminates if and only if there exists a reduction ordering $\succ$ such that $l \succ r$ for every rule $l \to r \in R$.*

Proof. "if": $s \to_R s'$ if and only if $s = t[l\sigma]_p$, $s' = t[r\sigma]_p$. If $l \succ r$, then $l\sigma \succ r\sigma$ and therefore $t[l\sigma]_p \succ t[r\sigma]_p$. This implies $\to_R \subseteq \succ$. Since $\succ$ is a well-founded ordering, $\to_R$ is terminating.

"only if": Define $\succ = \to_R^+$. If $\to_R$ is terminating, then $\succ$ is a reduction ordering. □

The Interpretation Method

Proving termination by interpretation:

Let $\mathcal{A}$ be a Σ-algebra; let $\succ$ be a well-founded strict partial ordering on its universe.

Define the ordering $\succ_\mathcal{A}$ over $T_\Sigma(X)$ by $s \succ_\mathcal{A} t$ iff $\mathcal{A}(\beta)(s) \succ \mathcal{A}(\beta)(t)$ for all assignments $\beta : X \to U_\mathcal{A}$.

Is $\succ_\mathcal{A}$ a reduction ordering?

Lemma 4.20 *$\succ_\mathcal{A}$ is stable under substitutions.*

Proof. Let $s \succ_\mathcal{A} s'$, that is, $\mathcal{A}(\beta)(s) \succ \mathcal{A}(\beta)(s')$ for all assignments $\beta : X \to U_\mathcal{A}$. Let σ be a substitution. We have to show that $\mathcal{A}(\gamma)(s\sigma) \succ \mathcal{A}(\gamma)(s'\sigma)$ for all assignments $\gamma : X \to U_\mathcal{A}$. Choose $\beta = \gamma \circ \sigma$, then by the substitution lemma, $\mathcal{A}(\gamma)(s\sigma) = \mathcal{A}(\beta)(s) \succ \mathcal{A}(\beta)(s') = \mathcal{A}(\gamma)(s'\sigma)$. Therefore $s\sigma \succ_\mathcal{A} s'\sigma$. □

A function $\phi : U_\mathcal{A}^n \to U_\mathcal{A}$ is called *monotone* (with respect to $\succ$), if $a \succ a'$ implies $\phi(b_1, \dots, a, \dots, b_n) \succ \phi(b_1, \dots, a', \dots, b_n)$ for all $a, a', b_i \in U_\mathcal{A}$.

Lemma 4.21 *If the interpretation $f_\mathcal{A}$ of every function symbol f is monotone w. r. t. $\succ$, then $\succ_\mathcal{A}$ is compatible with Σ-operations.*

Proof. Let $s \succ_{\mathcal{A}} s'$, that is, $\mathcal{A}(\beta)(s) \succ \mathcal{A}(\beta)(s')$ for all $\beta : X \to U_{\mathcal{A}}$. Let $\beta : X \to U_{\mathcal{A}}$ be an arbitrary assignment. Then

$$\begin{aligned}\mathcal{A}(\beta)(f(t_1, \dots, s, \dots, t_n)) &= f_{\mathcal{A}}(\mathcal{A}(\beta)(t_1), \dots, \mathcal{A}(\beta)(s), \dots, \mathcal{A}(\beta)(t_n)) \\ &\succ f_{\mathcal{A}}(\mathcal{A}(\beta)(t_1), \dots, \mathcal{A}(\beta)(s'), \dots, \mathcal{A}(\beta)(t_n)) \\ &= \mathcal{A}(\beta)(f(t_1, \dots, s', \dots, t_n))\end{aligned}$$

Therefore $f(t_1, \dots, s, \dots, t_n) \succ_{\mathcal{A}} f(t_1, \dots, s', \dots, t_n)$. □

Theorem 4.22 *If the interpretation $f_{\mathcal{A}}$ of every function symbol f is monotone w. r. t. $\succ$, then $\succ_{\mathcal{A}}$ is a reduction ordering.*

Proof. By the previous two lemmas, $\succ_{\mathcal{A}}$ is a rewrite relation. If there were an infinite chain $s_1 \succ_{\mathcal{A}} s_2 \succ_{\mathcal{A}} \dots$, then it would correspond to an infinite chain $\mathcal{A}(\beta)(s_1) \succ \mathcal{A}(\beta)(s_2) \succ \dots$ (with β chosen arbitrarily). Thus $\succ_{\mathcal{A}}$ is well-founded. Irreflexivity and transitivity are proved similarly. □

Polynomial Orderings

Polynomial orderings:

Instance of the interpretation method:

The carrier set $U_{\mathcal{A}}$ is $\mathbb{N}$ or some subset of $\mathbb{N}$.

To every function symbol f/n we associate a polynomial $P_f(X_1, \dots, X_n) \in \mathbb{N}[X_1, \dots, X_n]$ with coefficients in $\mathbb{N}$ and indeterminates $X_1, \dots, X_n$. Then we define $f_{\mathcal{A}}(a_1, \dots, a_n) = P_f(a_1, \dots, a_n)$ for $a_i \in U_{\mathcal{A}}$.

Requirement 1:

If $a_1, \dots, a_n \in U_{\mathcal{A}}$, then $f_{\mathcal{A}}(a_1, \dots, a_n) \in U_{\mathcal{A}}$. (Otherwise, $\mathcal{A}$ would not be a Σ-algebra.)

Requirement 2:

$f_{\mathcal{A}}$ must be monotone (w. r. t. $\succ$).

From now on:

$U_{\mathcal{A}} = \{ n \in \mathbb{N} \mid n \geq 1 \}$.

If arity$(f) = 0$, then P_f is a constant ≥ 1.

If arity$(f) = n \geq 1$, then P_f is a polynomial $P(X_1, \dots, X_n)$, such that every X_i occurs in some monomial $m \cdot X_1^{j_1} \cdots X_k^{j_k}$ with exponent at least 1 and non-zero coefficient $m \in \mathbb{N}$.

$\Rightarrow$ Requirements 1 and 2 are satisfied.