«LE MATEMATICHE»
Vol. XLIII (1988) - Fasc. I, pp. 225-246

AUTOMATED DEDUCTION IN TOPOLOGY: TWO DIFFERENT APPROACHES

VINCENZO CUTELLO (Catania, New York)

Two approaches to theorem proving in Topology are described and some research problems in the field are given .

1. Introduction.

In this paper we will survey some results (and related problems) concerning the mechanization of proofs of theorems in General Topology.

In particular, we will briefly describe two approaches to the problem that have been recently proposed. Both of them, though, in principle, very valuable, suffer from feasibility and theoretical problems, and might (and should) be considerably improved. To this end, we will propose some research problems.

In the appendix, we will describe a theorem proving system that has been used in one of the methods we are going to introduce.

For all the basic notions of set theory and mechanical theorem proving we will refer to [11] and [8], respectively.

2. Basic Topology Definitions.

Let us introduce now the basic concepts of General Topology. For a complete discussion of the subject see for instance [12].

A *Topology* $\mathcal{T}$ on a set $\mathcal{X}$ can be defined in three equivalent ways. Namely, in terms of

- neighborhood sets;
- open sets;
- closed sets.

We follow the second option.

DEFINITION 2.1. *The pair* $(\mathcal{X}, \mathcal{T})$, *where* $\mathcal{X}$ *is a nonempty set, is a topological space if* $\mathcal{T}$ *is a family of subsets of* $\mathcal{X}$, *which will be called open sets, such that*

- *the intersection of any two members of* $\mathcal{T}$ *is a member of* $\mathcal{T}$;
- *the union of the members of each subfamily of* $\mathcal{T}$ *is a member of* $\mathcal{T}$;
- $\mathcal{X} = \bigcup_{O \in \mathcal{T}} O.$

It is easily seen that $\mathcal{X}$ and $\emptyset$ are open sets.

DEFINITION 2.2. *A set* $C \subseteq \mathcal{X}$ *is said to be closed if there exists an open set* O *such that* $C = \mathcal{X} \setminus O$.

A set $U \subseteq \mathcal{X}$ *is said to be a neighborhood of a point* $x \in \mathcal{X}$ *if there exists an open set* O *such that* $x \in O \subseteq U$. ■

EXAMPLE 2.1. A trivial topology for a nonempty set $\mathcal{X}$ is the one given by $\mathcal{T} = \{\emptyset, \mathcal{X}\}$. Such a topology is called *indiscrete*.

At the other extreme is the topology given by $\mathcal{T}$ =*powerset*(X). This is called the *discrete* topology. ■

As we already mentioned, the topology of a topological space can be described in terms of neighborhoods of points. In order to give a definition of open and closed sets in terms of neighborhoods we need the following definition.

DEFINITION 2.3. *A point x is an accumulation point of a subset A of a topological space $(X, \mathcal{T})$ if and only if every neighborhood of x contains points of A other than x.* ■

The following theorems, then, hold.

THEOREM 2.1. *A subset of a topological space is closed if and only if it contains the set of its accumulation points.* ■

THEOREM 2.2. *The union of a set and the set of its accumulation points (closure of the set) is a closed set.* ■

The closure of a set, introduced in the last theorem, satisfies the following definition.

DEFINITION 2.4. (Kuratowski closure axioms) *A closure operator $^-$ is an operator which assigns to each subset A of X a subset $\bar{A}$ of X in such a way that the following statements are satisfied.*

(a) $\bar{\emptyset} = \emptyset$.

(b) For each $A, A \subseteq \bar{A}$.

(c) For each $A, \overline{(\bar{A})} = \bar{A}$.

(d) For each $A, B, \overline{A \cup B} = \bar{A} \cup \bar{B}$. ■

Several basic notions of general topology can be formulated in terms of the closure operator.

Namely, if we write $A' = X \backslash A$, we have that:

- $Int(A)$ $\equiv_{\text{Def}}$ $(\overline{A'})'$ (the *interior* of A);

- *Ext*(A) $\equiv_{\text{Def}}$ (A') (the *exterior* of A);
- $\partial(A)$ $\equiv_{\text{Def}}$ $\bar{A}\setminus$ *Int*(A) (the *boundary* of A).

Also, the following topological predicates are expressible:

- *open*(A) $\equiv_{\text{Def}}$ A =*Int*(A) (A is an *open* set);
- *closed*(A) $\equiv_{\text{Def}}$ $A = \bar{A}$ (A is a *closed* set);
- *open_domain*(A) $\equiv_{\text{Def}}$ A =*Int*$(\bar{A})$ (A is an *open domain*);
- *neighborhood*(A, a) $\equiv_{\text{Def}}$ $a \in$ *Int*(A) (A is a *neighborhood* of a);
- *dense*(A) $\equiv_{\text{Def}}$ $\bar{A} = 1$ (A is *dense* in the space 1);
- *co_dense*(A) $\equiv_{\text{Def}}$ *dense*(A') (A is *co_dense* in the space 1);
- *nowhere_dense*(A) $\equiv_{\text{Def}}$ *co_dense*$(\bar{A})$ (A is *nowhere_dense* in the space 1).

It should be noted that since the definition of open or closed set can be given in terms of the closure operator, the notion of topology can be also given in terms of the closure operator.

We conclude this section with the following definitions.

DEFINITION 2.5. *If* $(X, \mathcal{T})$ *is a topological space and* $\mathcal{B}$ *is a subfamily of* $\mathcal{T}$ *then* $\mathcal{B}$ *is a base for the topology* $\mathcal{T}$ *if and only if for each* $x \in X$ *and for each neighborhood* U *of* x *there exists a member* V *of* $\mathcal{B}$ *such that* $x \in V \subseteq U$. ■

If $(X, \mathcal{T})$ is a topological space and $Y \subseteq X$ we may construct a topology $\mathcal{V}$ for Y, which is called *relative topology*, by defining $\mathcal{V}$ as the family of all intersection of members of $\mathcal{T}$ with Y.

DEFINITION 2.6. *A topological space* $(X, \mathcal{T})$ *is connected if and only if* X *is not the union of two nonempty, disjoint and closed subsets of* X.

Analogously, if $Y \subseteq X$, *then* Y *is connected if the topological space with the relative topology is connected, or equivalently, if do not exist two nonempty, disjoint and closed sets in the topology* $(X, \mathcal{T})$

whose union contains Y. ■

DEFINITION 2.7. *A topological space* $(X, \mathcal{T})$ *is compact if and only if each open cover*(1) *has a finite subcover.* ■

Using the De Morgan formulas on complements, we have the following characterization of compactness in terms of closed sets.

THEOREM 2.3. *A topological space* $(X, \mathcal{T})$ *is compact if and only if each family of closed sets which has the finite intersection property*(2) *has a nonempty intersection.* ■

DEFINITION 2.8. *Let* $(X, \mathcal{T})$ *and* $(\mathcal{Y}, \mathcal{V})$ *be two topological spaces. Let* f *be a function from* X *to* $\mathcal{Y}$.

(a) f *is said to be continuous if and only if the inverse image of an open set in* $\mathcal{Y}$ *is an open set in* X.

(b) f *is said to be closed [resp. open] if and only if the image of a closed set [reps. open set] in* X *is a closed set [resp. open set] in* $\mathcal{Y}$. ■

3. The Resolution-Based Method.

In this section we will review the work done to present a first-order formulation of point-set topology, in order to automatically prove theorems using a resolution-paramodulation based theorem prover system. For a full discussion see [15].

Let us first show a sorted first-order formulation of some of the definitions given in the previous section.

---

(1) An open cover is a family of open sets $\mathcal{A}$ such that $X = \bigcup_{O \in \mathcal{A}} O$. The cover is finite if the family $\mathcal{A}$ is finite.

(2) The finite intersection property requires that each finite subfamily of a family $\mathcal{A}$ has a nonempty intersection.

The implicit rule that we will use is that lower case letter symbols, e.g. u, v, w, x, y, z, represent points, whereas symbols starting with $A, B, C, U, V, W, X, Y, Z$, represent sets and symbols starting with F, G, H, S, T represent family of sets.

- The union of the members of a family F (which we will indicate with $sigma(F)$) is given by the formula

$$\forall F \forall u \left(u \in sigma(F) \leftrightarrow \exists A \left\{ \begin{array}{c} u \in A \wedge \\ A \in F \end{array} \right\} \right)$$

- The intersection of the members of a family F (which we will indicate with $pi(F)$ is given by the formula

$$\forall F \forall u (u \in pi(F) \leftrightarrow \forall A (A \in F \rightarrow u \in A))$$

- So the definition of topological space can be given as

$$\forall X \forall T \left(top\_space(X,T) \leftrightarrow \left\{ \begin{array}{c} sigma(T) \subseteq X \wedge \\ \emptyset \in T \wedge X \in T \wedge \\ \forall Y \forall Z (Y \in T \wedge Z \in T \rightarrow (Y \cap Z) \in T) \wedge \\ \forall F (F \subseteq T \rightarrow sigma(F) \in T) \end{array} \right\} \right)$$

- Predicates for open and closed sets have a very simple definition

$$\forall U \forall X \forall T \left(open(U,X,T) \leftrightarrow \left\{ \begin{array}{c} top\_space(X,T) \wedge \\ U \in T \end{array} \right\} \right)$$

and

$$\forall U \forall X \forall T \left(closed(U,X,T) \leftrightarrow \left\{ \begin{array}{c} top\_space(X,T) \wedge \\ open(X-U,X,T \end{array} \right\} \right)$$

- The definition of neighborhood is the following(3)

(3) In [15] the neighborhood of y is defined as an open set that contains y.

$$\forall U \forall y \forall X \forall T \left(neighborhood(U,y,X,T) \leftrightarrow \exists A \left\{ \begin{array}{c} top\_space(X,T) \wedge \\ open(A,X,T) \wedge A \subseteq U \wedge \\ y \in A \end{array} \right\} \right)$$

- The interior of a set

$$\forall Y \forall X \forall T \forall u$$
$$\left(u \in interior\ (Y,X,T) \leftrightarrow \left\{ \begin{array}{c} top\_space(X,T) \wedge Y \subseteq X \wedge \\ \exists V (u \in V \wedge V \subseteq Y \wedge open\ (V,X,T)) \end{array} \right\} \right)$$

- The closure of a set

$$\forall Y \forall X \forall T \forall u$$
$$\left(u \in closure\ (Y,X,T) \leftrightarrow \left\{ \begin{array}{c} top\_space(X,T) \wedge Y \subseteq X \wedge \\ \forall V (Y \subseteq V \wedge closed\ (V,X,T) \rightarrow u \in V) \end{array} \right\} \right)$$

To the list of formulas given above, must be added all the formulas concerning the set theory concepts used, as for instance $\subseteq, \setminus$.

Moreover, since in the language there are three sorts of variables, namely *points, sets and collections of sets*, in any refutation process unification of variables of different types must be avoided.

In [15] three solutions are proposed for this problem.

- Include type literals for the different types. So any formula $\forall x \phi$ where x is, for instance, a set variable, should be rewritten as $\forall x(set(x) \rightarrow \phi)$, whereas, any formula of kind $\exists x \phi$ where x is, for instance, a point variable, should be rewritten as $\exists x(point(x) \wedge \phi)$ (called *relativization of quantifiers*).

- Put type functions around terms. So for instance, literals of kind $open(x)$ should be rewritten as *open* $(set(x))$.

- Use implicit typing, i.e. the position of each argument determines its type. A type is associated with each constant and function symbol, and a type is associated with each argument position of each predicate and function symbol.

Since the first solution does not assure that unwanted unifications do not happen and the second solution appears to be too expensive because of the extra type functions, the last solution is followed.

In order to make things work, the membership relation is partitioned into two relations. Namely, *el_p*(x, y) which says that x is a point that belongs to the set y and *el_s*(x, y) which says that x is a set that belongs to the collection y.

Analogously, the binary predicates *subset_s* and *subset_c* are introduced, meaning respectively set inclusion and collection inclusion.

Here follows a complete list of the set theoretic predicates that are needed for this approach.

$\in$ *el*_p(point,set), el_s(set,collection);

$=$ eq_p(point, point), eq_s(set,set), eq_c(collection,collection);

$\subseteq$ subset_s(set,set), subset_c(collection,collection);

disjoint disjoint_s(set,set), disjoint_c(collection,collection);

$\cup$ Union_s(set,set), Union_c(collection,collection);

$\cap$ inter_s(set,set), inter_c(collection,collection);

$\setminus$ rel_comp_s(set,set), rel_comp_c(collection,collection);

$\emptyset$ 0_s (empty set), 0_c (empty collection).

Once these new predicates are introduced, first order formulas to handle them must be introduced as well. For instance, several properties of the equality relations, such as reflexivity, transitivity, symmetry and substitutivity must be stated. All these first-order formulas must then the transformed into clausal form.

For instance, the clausal representation of the first order formula defining a topological space is the following:

```
1 -top_space(x,vT) | eq_s(sigma(vT),x).
2 -top_space(x,vT) | el_s(0_s,vT).
3 -top_space(x,vT) | el_s(x,vT).
4 -top_space(x,vT) | -el_s(y,vT) | -el_s(z,vT) |
```

```
                el_s(inter_s(y,z),vT).
5 -top_space(x,vT) | -subset_c(vF,vT) |
                -el_s(sigma(vF),vT).
6  top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(f3(x,vT),vT |
                subset_c(f5(x,vT),vT).
7  top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(f3(x,vT),vT |
                -el_s(sigma(f5(x,vT)),vT).
8  top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(f4(x,vT),vT |
                subset_c(f5(x,vT),vT).
9  top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(f4(x,vT),vT |
                -el_s(sigma(f5(x,vT)),vT).
10 top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(inter_s(f3(x,vT),
                f4(x,vT)),vT) | subset_c(f5(x,vT),vT).
11 top_space(x,vT) | -eq_s(sigma(vT),x)  | el_s(0_s,vT) |
                -el_s(x,vT) | el_s(inter_s(f3(x,vT),
                f4(x,vT)),vT) | -el_s(sigma(f5(x,vT),vT)
```

where, variables start with a symbol in $[u, v, w, x, y, z]$ and the symbols $f1, f2, \ldots, g1, g2 \ldots$ are Skolem functions.

The final result is a huge number of clauses that the system must handle and which make this approach inadequate. To confirm what we have just stated, in [15] an example is shown of a proof of the theorem:

`The topology generated by a basis gives rise to a topological space.`

Such a theorem, very simple from a conceptual point of view, is proved in five steps, and for each step hundreds of clauses are generated [4]

3.1. What should be done.

Let us now remark what we think should be done in order to make the above described approach more effective.

- Much of the work that any theorem prover system must do in its refutation process, is concerned with set theoretic constructs. So, if the problem of finding an inference rule that plays for set theoretic constructs, the same role that paramodulation plays for equality was solved (see [16]), this approach will greatly increase its effectiveness.
- Since paramodulation cannot be used in this case, because three different equality operators, namely $=_{\text{point}}$, $=_{\text{set}}$, $=_{\text{collection}}$, are to be considered, a modified version of the inference rule that allows one to deal with different sorts of variables might be very useful.
- Finally, a *local* search strategy is strongly needed. Indeed, the number of clauses in the set of axioms is very large, nevertheless, in the majority of the cases only a few of them are needed to prove a given theorem.([5]) So, given a theorem, the system should be able to weigh the clauses in such a way to involve in the refutation process first the clauses that are more likely to be effective.

4. A Decision Procedure.

In this section, we describe a different approach to the problem of

[4] We recall here that the system used by the authors of the described work is OTTER, see the last section for a brief description.

([5]) For instance, to prove that the intersection of two neighborhood of a point x is also a neighborhood of x, one only needs to consider that the intersection of two open sets is an open set and that if $A \subseteq B$ and $C \subseteq D$ then $A \cap C \subseteq B \cap D$.

automated deduction in Topology. We will refer to the work described in [6]. The result in this paper is strongly related to a large amount of work, done in the last few years, in the field of *Computable Set Theory*, (see, for instance, [9], [1], [5], [14], [2], [7], [4], [3]).

As we will see in more details, the work to be described is based on the result obtained in [10], of which the above mentioned papers, are extensions.

4.1. Two and Multi Level Syllogistic and the satisfiability problem.

By *Two Level Syllogistic* ($2LS$) we mean the unquantified languages, obtained as propositional closure of the atoms:

- $X = Y \cup Z$;
- $X = Y \backslash Z$;
- $X = Y \cap Z$;
- $x \in X$;
- $X = \{x\}$;

where, capital letters indicate set variables and small letters indicate element variables.

By *Multi Level Syllogistic (MLS)* we mean the generalization of the above theory, where all the variables simply represent set variables, i.e. the elements of a set are themselves set. The decidability of the first theory was studied in [10] while in [9] the decidability of the Multi Level Syllogistic is shown.

By decidable theory we mean that there exists an algorithm that, given any formula $\phi(x_1, x_2, \dots, x_n)$ of the theory, checks in a finite amount of time, whether or not there exists a model that satisfies the formula(6). In this case, we will also say that the given theory has a solvable satisfiability problem.

(6) In other words, both the sets of satisfiable and unsatisfiable formulas of the given theory are recursively enumerable.

Notice that for each formula $\phi(x_1, x_2, \ldots, x_n)$, with free variables $x_1, \ldots, x_n$, the following are equivalent:

- $\phi(x_1, \ldots, x_n)$ is satisfiable;
- $(\exists x_1) \ldots (\exists x_n)\phi(x_1, \ldots, x_n)$ is valid;
- $\neg(\forall x_1) \ldots (\forall x_n)\neg\phi(x_1, \ldots, x_n)$ is valid.

Therefore, any satisfiability test for an unquantified theory, provides a validity test for the universal and the existential closures of the formulas in the theory.

4.2. The theory $\mathcal{L}^{2,\infty}$.

The solvability of the satisfiability problem for the two theories cited above, helps to solve the satisfiability problem for a syllogistic extended with the Kuratowski topological closure operator $^-$, continuous and closed maps between topological spaces, along with the operations of point evaluation, set image, and inverse set image, in the case of a syntactic non circularity property, that we will describe in what follows.

The theory $\mathcal{L}^{2,\infty}$ is a two level multi-sorted language with functions. For each positive integer n (whose intented meaning is to denote a particular topological space) we have

- two denumerable sequence of variables $(x_0^{(n)}, \ldots, x_h^{(n)}, \ldots, y_0^{(n)}, \ldots, y_h^{(n)}, \ldots)$ (denoting individual variables);
- two denumerable sequence of variables $(X_0^{(n)}, \ldots, X_h^{(n)}, \ldots, Y_0^{(n)}, \ldots, Y_h^{(n)}, \ldots)$ (denoting set variables);
- the constants $0^{(n)}$ and $1^{(n)}$ denoting respectively the empty set and the universe of the topological space denoted by n;
- the unary operators $'^{(n)}$ and $^{-(n)}$ denoting respectively the set complementation (with respect to the universe $1^{(n)}$) and the Kuratowski closure operator;

The usual binary operators $\cup, \cap, \setminus$ and the binary predicates $=, \subseteq, \in$ are also available, along with the propositional connectives $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$. Finally, for each pair of integers (n, m) two denumerable sequences of variables are available. They are of type $f_h^{(n,m)}$ (denoting a continuous function from the topological space n to the topological space m) and $g_h^{(n,m)}$ (denoting a closed function from the topological space n to the topological space m).

DEFINITION 4.1. *For each* n

- *each individual variable of sort* n *is an individual term of sort* n;
- *for any* m *and any individual term* $t^{(n)}$ *of sort* n $f^{(n,m)}(t^{(n)})$, $g^{(n,m)}(t^{(n)})$ *are individual terms of sort* m;
- *each set variable of sort* n *and* $1^{(n)}$, $0^{(n)}$ *are set terms of sort* n;
- *if* $T_1^{(n)}$, $T_2^{(n)}$ *are set terms of sort* n *then* $T_1^{(n)} \cup T_2^{(n)}$, $T_1^{(n)} \cap T_2^{(n)}$, $T_1^{(n)} \setminus T_2^{(n)}$, $(T_1^{(n)})'$, $\overline{T_1^{(n)}}$ *are set terms of sort* n;
- *for each* m $(f^{(m,n)})^{-1}[T_1^{(n)}]$, $g^{(n,m)}[T_1^{(n)}]$ *are set terms of sort* n. ■

Formulae of $\mathcal{L}^{2,\infty}$ are propositional combinations of atoms

$$t_1^{(n)} = t_2^{(n)}, \qquad t_1^{(n)} \in T_1^{(n)},$$

$$T_1^{(n)} = T_2^{(n)}, \qquad T_1^{(n)} \subseteq T_2^{(n)},$$

where $t_1^{(n)}, t_2^{(n)}$ are individual terms of sort n and $T_1^{(n)}$, $T_2^{(n)}$ are set terms of sort n, for some natural number n.

By using a simple normalization process, it is immediate to see that the satisfiability problem for formulas of $\mathcal{L}^{2,\infty}$ can be reduced to the satisfiability problem for conjunctions of literals of type

$$(1) \qquad \begin{array}{lll} x = y, & x \neq y, & y = f(x), \\ y = g(x), & x \in X, & X = 0, \\ X = 1, & X = f^{-1}[Y], & Y = g[X], \\ X = \bar{Y}, & X = Y \cup Z, & X = Y \setminus Z, \end{array}$$

where x, y stand for individual variables, f and g for a continuous and a closed map variable respectively, and X, Y for set variables.

Let us see in more detail, what we mean by satisfiability of formulas of $\mathcal{L}^{2,\infty}$.

DEFINITION 4.2. *A topological assignment M is any interpretation of the constants and variables of $\mathcal{L}^{2,\infty}$ such that*

- *$M1^{(n)}$ is a non empty set endowed with a topology $\tau^{(n)}$ and $M0^{(n)}$ is the empty set.*
- *For each individual variable $x^{(n)}, Mx^{(n)} \in M1^{(n)}$.*
- *For each set variable $X^{(n)}, MX^{(n)} \subseteq M1^{(n)}$.*
- *${}'^{(n)}$ is interpreted as the set complementation in $M1^{(n)}$.*
- *${}^{-(n)}$ is interpreted as the Kuratowski closure for the topological space $(M1^{(n)}, \tau^{(n)})$.*
- *$\cup, \cap, \setminus$ are interpreted as the usual set binary operators.*
- *$Mf^{(n,m)}$ [resp. $Mg^{(n,m)}$] is a continuous [resp. closed] map from the topological space $(M1^{(n)}, \tau^{(n)})$ into the topological space $(M1^{(m)}, \tau^{(m)})$.* ■

DEFINITION 4.3. *A formula ϕ of $\mathcal{L}^{2,\infty}$ is topologically satisfiable, if there exists a topological assignment M such that $M\phi$ is true.*

A formula ϕ of $\mathcal{L}^{2,\infty}$ is topologically valid, if $M\phi$ is true for all the topological assignments M. ■

We have already mentioned a non-circularity property to hold for formulas of $\mathcal{L}^{2,\infty}$. More precisely, let ϕ be a formula of $\mathcal{L}^{2,\infty}$. Let $N_\phi = \{i_0, i_1, \ldots, i_n\}$ be the set of sorts of all individual and set variables. We associate to ϕ the graph $G_\phi = (N_\phi, E_\phi)$ where there is an edge in E_ϕ $i_s \Rightarrow i_t$ if and only if either ϕ contains some occurrences of a continuos map variable of sort (i_t, i_s) or some occurrences of a closed map variables of sort (i_s, i_t).

DEFINITION 4.4. *We say that a formula ϕ of $\mathcal{L}^{2,\infty}$ is acyclic if its associated graph is acyclic.* ■

Notice that the decidability of the theory $\mathcal{L}^{2,\infty}$ is obtained by means of a reduction process, that starting from an acyclic conjunction of type (1), eliminates all the map constructs and the closure operator constructs reducing the satisfiability problem for acyclic formulas of $\mathcal{L}^{2,\infty}$ to the satisfiability problem for formulas of two-level syllogistic (see [6] for all the details).

We have then the following lemma.

LEMMA 4.1. *Given any acyclic formula ϕ of the theory $\mathcal{L}^{2,\infty}$, it is possible to build, in a finite amount of time, a formula ϕ^* of the theory 2LS such that ϕ is satisfiable if and only if ϕ^* is satisfiable.* ∎

The above lemma, combined with the solvability of the satisfiability problem for the 2LS gives the following theorem.

THEOREM 4.1. *The class of acyclic formulas in the language $\mathcal{L}^{2,\infty}$ has a solvable satisfiability problem.* ∎

4.3. Examples.

In section 2 we have seen how many constructs of topology can be expressed in terms of the Kuratowski's closure operator; then they can be expressed in the language $\mathcal{L}^{2,\infty}$. In particular, since the class of acyclic formulas of $\mathcal{L}^{2,\infty}$ is closed under negation, we have that the validity of the following statements can be checked by the satisfiability test for the acyclic formulas of $\mathcal{L}^{2,\infty}$:

(a) the intersection of two open domains is an open domain;

(b) the union of two open domains need not be an open domain;

(c) if A and B are open domains, then

$$A \subseteq B \leftrightarrow \bar{A} \subseteq \bar{B};$$

(d) $\bar{\emptyset} = \emptyset$, and for all A and B

$$A \subseteq \bar{A}, \bar{\bar{A}} = \bar{A}, \overline{A \cup B} = \bar{A} \cup \bar{B}$$

(Kuratowski's closure axioms);

(e) if A is co-dense and B is nowhere dense, then $A \cup B$ is co-dense;

(f) the union of two co-dense sets is not necessarily a co-dense set;

(g) for all subsets A of a topological space U,

$$Int(A) \cup \partial(A) \cup Int(A') = U;$$

(h) for each continuous map f and each set B,

$$f^{-1}[Int(B)] \subseteq Int(f^{-1}[B]);$$

(i) for all A,

$$Int(A) \subseteq Ext(Ext(A));$$

(j) every continuous map is *locally* continuous, i.e., for all x, if B is a neighborhood of $f(x)$, then $f^{-1}[B]$ is a neighborhood of x.

4.4. Complexity and future work.

The satisfiability problem for the two-level syllogistic (or multi-level syllogistic) is $\mathcal{NP}$-complete. It turns out then that the satisfiability problem for the theory $\mathcal{L}^{2,\infty}$ is $\mathcal{NP}$-hard. The algorithm described in [6] has a prohibitive complexity. Namely, if h is the length of the longest path in the associated graph of a formula ϕ, then the cited algorithm decides the satisfiability of the formula ϕ in time

$$O\left(2^{2^{2^{\cdot^{\cdot^{\cdot^{n}}}}}}\right)$$

where n is the total number of variables in the formula and the length of the exponential stack is $h+1$. Nevertheless, we strongly believe that such algorithm can be improved so as to work in polynomial time in most cases.

Such improvement might be realized by endowing the algorithm with tests that are able to check quickly if some basic set theoretical properties and axioms are falsified by any model that should satisfy a given formula(7).

Along with these *implementation* details, from a theoretical point of view, mush work is still to be done. Here we try to sketch some possible future research problems:

- the acyclicity condition should be eliminated. This implies the search for an algorithm able to recognize the existence (and eventually able to build) topological spaces with an infinite number of closed (and then open) sets.
- In [6], the decidability of the theory $\mathcal{L}^{2,\infty}$ extended with the predicate *accumulation_point*(x, X) (with the intended meaning x *is an accumulation point for the set* X) and the operator *Der*(X) (i.e. the set of all accumulation point of X), is conjectured but not proved.
- Even more interesting is the extension of $\mathcal{L}^{2,\infty}$ with predicates *connected*(X) and *compact*(X) (with obvious intended meanings). Since *connected*(X) and *compact*(X) imply a universal quantification over the open and closed sets, we expect the decidability of such extensions (if decidable) to be harder to obtain than the solutions to the open problems above cited.

5. Appendix: The Otter System.

OTTER (Other Techniques for Theorem-proving and Effective Research) is a resolution-style, non interactive theorem proving system recently designed and implemented at the *Argonne National Laboratory, Mathematics and Computer Science Division* (see [13] for a complete description of the system).

OTTER includes the inference rules binary resolution, hyperreso-

(7) For instance, many of the two-level or multi-level syllogistic formulas are unsatisfiable because the Axiom of Foundation would not be valid in any model.

lution, UR-resolution, and binary paramodulation. Some of its other abilities are conversion from first-order formulas to clauses, forward and back subsumption, factoring, weighting, answer literals, term ordering, forward and back demodulation, and evaluable functions and predicates. OTTER is coded in C, and it is portable to a wide variety of computers.

It is to be noted that the user decides which inference rules should be adopted, which sets options to control the processing of inferred clauses, which input formulas or clauses are to be in the initial set of support, and which equalities are to be demodulators.

A typical input file for OTTER is the following:

```
set(flag_name).                % set a flag, as many flags as wanted
...
assign(parameter_name,         % assign an integer to a parameter
integer).

list(axioms).                  % read axioms in clause form
...
end_of_list.                   % it ends the list of given axioms

list(sos)                      % read set of support in clause form
...
end_of_list.                   % it ends the list of sos

list(demodulators).            % read demodulators in clause form
...
end_of_list.                   % it ends the list of demodulators
```

OTTER can be given also axioms and set of support in formula form.

EXAMPLE 5.1. The following two lists of axioms are equivalent.

The first is in clausal form,

```
list(axioms).
  (x=x).                              % reflexivity
  (f(e,x)=x).                         % left identity
  (f(g(x),x)=e).                      %left inverse
  (f(f(x,y),z)=f(x,f(y,z))).          % associativity
  (f(z,x)!=f(z,y))|(x=y).             % left cancellation
  (f(x,z)!=f(y,z))|(x=y).             % right cancellation
end_of_list
```

whereas the following is in formula form

```
formula_list(axioms).
  (all a (a=a)).                        % reflexivity
  (all a (f(e,a)=a)).                   % left identity
  (all a f(g(a),a)=e)).                 % left inverse
  (all a all b all c                    % associativity
     (f(f(a,b),c)=f(a,f(b,c)))).
  (all a all b all c                    % left cancellation
     ((f(c,a)=f(c,b))->(a=b))).
  (all a all b all c                    % right cancellation
     ((f(a,c) = f(b,c))->(a=b))).
end_of_list
```

The main loop for inferring and processing clauses and searching for a refutation is

```
While (sos is not empty and no refutation has been found)
  1. Let given_clause be the first clause in sos;
  2. Move given_clause from sos to axioms;
  3. Infer and process new clauses using the inference rules
     in effect; each new clause must have the given_clause
     as one of its parents and members of axioms as its
     other parents; new clauses that pass the retention
     tests are appended to sos;
End of while loop.
```

5.1. An example.

We show now how OTTER works on the following input file.

```
% This file contains the axioms of a group where f represents
  the product
% of two elements and g gives the inverse of an element.
% The product satisfies the special property that f(x,x)=x
  for all x.
% The goal is to prove that the product is commutative; i.e.
  the group is
% abelian.

set(para_from).
set(para_into).
set(para_from_left).
set(para_from_right).
set(para_from_vars).
set(para_into_vars).
set(dynamic_demod_all).

    list(axioms).
        (x=x).                        % reflexivity
        (f(e,x)=x).                   % left identity
    %     (f(x,e)=x).                 % right identity
        (f(g(x),x)=e).                % left inverse
    %     (f(x,g(x)=e).               % right inverse
        (f(x,x)=x).                   % special property
    end_of_list

list(sos).
% In order to speed up the refutation process, the associa-
  tivity property
% of f is put in the set of support.
% Also, the right identity and inverse property are forced
  not to be used.
```

```
% In the following a,b (as well as e in the above clauses)
  are skolem
% constants.

(f(x,y),z)=f(x,f(y,z))).  % associativity.
(f(a,b)!=f(b,a)).         % Denial of the commutativity
                                 property of f
end_of_list
```

Here is the refutation produced by OTTER

```
2 (fe,x)=x).
3 (f(g(x),x)=e).
4 (f(x,x)=x).
5 (f(f(x,y),z)=f(x,f(y,z))).
6 (f(a,b)!=f(b,a)).
19 [para_into,2,6] (f(e,f(b,a))!=f(a,b)).
192 [para_into,4,5] (f(f(x,y),y)=f(x,y)).
193 [new_demod,192] (f(f(x,y),y)=f(x,y)).
254 [para_into,3,192] (f(g(x),x)=f(e,x)).
255 [new_demod,254] (f(g(x),x)=f(e,x)).
263 [para_into,3,192,demod,255,193] (f(e,x)=e).
264 [new_demod,263] (f(e,x)=e).
265 [para_into,2,192,demod,193,264] (f(x,y)=e).
266 [new_demod,265] (f(x,y)=e).
267 [para_into,2,192,demod,266,266] (x=e).
268 [para_from,192,19,demod,266,266,266,266,266] (e!=e).
269 [binary,268,267].
```

REFERENCES

[1] Breban M., Ferro A., Omodeo E., Schwartz J.T., *Decision Procedures for Elementary Sublanguages of Set Theory II,* Formulas involving Restricted Quantifiers, together with Ordinal, Integer, Map, and Domain

Notions. Communications on Pure and Applied Mathematics, **34**, (1981), 177-195.

[2] Cantone D., *A Decision Procedure for a Class of Unquantified Formulae of Set Theory Involving the Powerset and Singleton Operators,* PhD thesis, New York University, October 1986.

[3] Cantone D., Cutello V., *Decision procedures for elementary sublanguages of set theory. XV.,* Multilevel syllogistic extended by the predicate *finite* and the operators singleton and $pred_<$. Journal of Automated Reasoning, to appear.

[4] Cantone D., Cutello V., Ferro A., *Decision procedures for elementary sublanguages of set theory. XIV.,* Three languages involving rank related constructs. In International Symposium on Symbolic and Algebraic Computation, to appear.

[5] Cantone D., Ferro A., Schwartz J.T., *Decision Procedures for Elementary Sublanguages of Set Theory,* V. Multilevel Syllogistic Extended by the General Union Operator. Journal of Computer and Systems Sciences., **34** (1), (1987), 1-18.

[6] Cantone D., Omodeo E., *On the decidability of formulae involving continuous and closed functions*. In Proc. of IJCAI, 1989.

[7] Cantone D., Schwartz J.T., *Decision procedures for elementary sublanguages of set theory,* XI. Multilevel syllogistic extended by some elementary map constructs. To appear on Journal of Automated Reasoning.

[8] Chang C.L., Lee R.C.T., *Symbolic Logic and Mechanical Theorem Proving,* Academic Press, New York, 1973.

[9] Ferro A., Omodeo E., Schwartz J.T., *Decision procedure for elementary sublanguages of set theory,* I. Multilevel syllogistic and some extensions. Comm. Pure Appl. Math. 33 (1980) 559-608

[10] Ferro A., Omodeo E.G., *An efficient validity test for formulae in extensional two-lelvel syllogistic,* Le Matematiche (Catania, Italy), **33**, (1978), 130-137.

[11] Jech T.J., *Set theory,* Academic Press, New York, 1978.

[12] John L. Kelley., *General topology,* Springer-Verlag, new York, reprint of the 1955 ed. published by Van Nostrand edition, 1961.

[13] McCune W.W., *OTTER 1.0 Users' Guide,* Technical Report 44, Argonne National Laboratory, 1988.

[14] Omodeo E.G., *Decidability and proof procedures for set theory with a choice operator,* PhD thesis, New York University, 1984.

[15] Wick C.A., McCune W.W., *Automated reasoning about elementary point-set topology,* Journal of Automated Reasoning, (1989) 239-255.

[16] Wos L, *Automated Reasoning: 33 basic research problems,* Prentice Hall, 1988.

Courant Institute of Mathematical Sciences,
New York University,
New York

Università di Catania
Dipartimento di Matematica
Catania