Rough Resolution: A Refinement of Resolution to Remove Large Literals*

Heng Chu and **David A. Plaisted**
Department of Computer Science
University of North Carolina
Chapel Hill, NC 27599-3175, USA
{chu|plaisted}@cs.unc.edu

Abstract

Semantic hyper-linking [Plaisted *et al.*, 1992, Chu and Plaisted, 1993, Chu and Plaisted, 1992] has been proposed recently to use semantics with hyper-linking [Lee and Plaisted, 1992], an instance-based theorem proving technique. Ground instances are generated until an unsatisfiable ground set is obtained; semantics is used to greatly reduce the search space. One disadvantage of semantic hyper-linking is that large ground literals, if needed in the proofs, sometimes are hard to generate. In this paper we propose *rough resolution*, a refinement of resolution [Robinson, 1965], to only resolve upon *maximum literals*, that are potentially large in ground instances, and obtain *rough resolvents*. Rough resolvents can be used by semantic hyper-linking to avoid generating large ground literals since maximum literals have been deleted. As an example, we will show how rough resolution helps to prove LIM3 [Bledsoe, 1990], which cannot be proved using semantic hyper-linking only. We will also show other results in which rough resolution helps to find the proofs faster. Though incomplete, rough resolution can be used with other complete methods that prefer small clauses.

Introduction

Semantic hyper-linking has been recently proposed [Plaisted *et al.*, 1992, Chu and Plaisted, 1992, Chu and Plaisted, 1993] to use semantics with hyper-linking [Lee and Plaisted, 1992]. Some hard theorems like IMV [Bledsoe, 1983] and ExQ [Wang, 1965] problems have been proved with user-provided semantics only.

Semantic hyper-linking is a complete, instance-based refutational theorem proving technique. Ground instances of the input clauses are generated and a satisfiability check is applied to the ground instance set. Semantics is used to reduce the search space and keep relevant ground instances. Size is measured based on the largest literal in the clause and smaller ground instances are generated first. Such an instance generation strategy, however, imposes difficulties generating large ground instances of a literal since there might be many smaller (yet irrelevant) ones that need to be generated first. For example, for some LIM+ problems [Bledsoe, 1990], a correct ground instance of the goal clause

$$(D \leq 0) \vee \sim(|f(xs(D)) - f(a)| + |g(xs(D)) - g(a)| \leq e0)$$

has to be generated by substituting variable D with $min(d1(ha(e0)), d2(ha(e0)))$ to obtain the proof. The ground instance is large and, with normal semantics, semantic hyper-linking generates many irrelevant smaller instances and diverts the search. Similar problems happen when large ground literals need to be generated in axiom clauses.

Rough resolution addresses this problem by resolving upon (and deleting) literals that are potentially large. Thus large ground literals are not needed and proofs are easier to obtain by semantic hyper-linking. However, rough resolution is against the philosophy of hyper-linking, namely, not to combine literals from different clauses. We set restrictions to reduce the number of retained resolvents so duplication of search space is not a serious problem as in ordinary resolution.

In this paper we first describe semantic hyper-linking in brief. Then we discuss in detail the rough resolution technique. An example (LIM3 problem) is given to help illustrate the ideas. Finally we give some more test results and then conclude the paper.

Semantic Hyper-Linking

In this section we briefly describe semantic hyper-linking. For more detailed discussion, please refer to [Chu and Plaisted, 1992, Chu and Plaisted, 1993].

A refutational theorem prover, instead of showing a theorem H logically follows from a set of axiom clauses A, proves A and $\sim H$ is unsatisfiable by deriving contradiction from the input clauses set. Usually we can find *semantics* for the theorem (and the axioms) to be proved. Such semantics can be represented by a

*This research was partially supported by the National Science Foundation under grant CCR-9108904

structure which contains a *domain* of the objects and *interpretations* for constants, functions and predicates. A structure can be viewed as a (possibly infinite) set of all ground literals true in the semantics. A structure I is *decidable* if for a first-order formula F it is decidable whether F is satisfied by I.

According to Herbrand theorem, in the input clauses for a refutational theorem prover, there must be ground instances of some clauses (usually the negation of the theorem) that are false in the semantics; otherwise the input clauses are satisfiable. In general, no matter what semantics is chosen, such false ground instances exist for the input set. This observation is the base of semantic hyper-linking.

The idea of semantic hyper-linking is this: Initially a decidable input structure is given by the user, then the prover systematically *changes* the structure (semantics) by generating ground instances of input clauses that are false in the structure. We have a set U of ground instances, initially empty. Ground instances that are false in the current structure, are generated. User-provided semantics is used to help generate the new ground instances. These ground instances are added to U. Then the structure is changed, if possible, to satisfy U. This procedure is repeated with the new structure until U is unsatisfiable. For details, please see [Plaisted *et al.*, 1992, Chu and Plaisted, 1992, Chu and Plaisted, 1993].

Semantic hyper-linking has been shown to have great potential. Hard theorems like IMV (the intermediate value theorem in analysis) [Bledsoe, 1983, Ballantyne and Bledsoe, 1982] and ExQ (three examples from quantification theory) [Wang, 1965] are proved with the user-provided semantics only. No other human control is needed. However, if large ground literals are needed for the proof, they often are difficult to generate because of the way the ground instances are generated (which is basically an enumeration of the Herbrand base). Many small irrelevant ground literals need to be generated before large ones are generated. This generates a lot of useless smaller ground instances to complicate the proof search and makes many theorems unable to be proved. Rough resolution is designed to address this problem.

Rough Resolution

The basic idea of rough resolution is simple: we only resolve on those literals that are potentially large in ground instances. Those literals resolved upon are deleted in the resolvents and their ground instances need not be generated. If such large ground literals are used in the proof, rough resolution can avoid generating them and help semantic hyper-linking find the proof faster. For example, consider the following clauses (x, y and z are variables):

$C_1 = \{ \sim g(x, y, f(z)), d(z, x) \}$
$C_2 = \{ g(f(x), g(y), z), \sim f(x, y) \}$

If the following two ground instances are needed in the proof which needs no other instances of $g(f(a), g(h(b)), f(c))$:

$C_1' = \{ \sim g(f(a), g(h(b)), f(c)), d(c, f(a)) \}$
$C_2' = \{ g(f(a), g(h(b)), f(c)), \sim f(a, h(b)) \}$

We can resolve C_1 and C_2 upon the first literals and get

$C = \{ d(z, f(x)), \sim f(x, y) \}$

Then, instead of C_1' and C_2', a smaller ground instance

$C' = \{ d(c, f(a)), \sim f(a, h(b)) \}$

can be used in the proof and avoids the use of larger $g(f(a), g(h(b)), f(c))$ which might be difficult to generate by semantic hyper-linking.

Maximum Literals

Binary resolution on two clauses $C1$ and $C2$ chooses one literal L in $C1$ and one literal M in $C2$ such that $L\theta = \sim M\theta$, where θ is a most general unifier of L and $\sim M$. A *resolvent* $R = (C1 - L)\theta \cup (C2 - M)\theta$ is generated. R is *smooth* if $L\theta\sigma$ is never larger than any literal in $R\sigma$ for any σ; R is *rough* if for some σ, $L\theta\sigma$ is larger than any literal in $R\sigma$. Smooth resolvents are not kept because they only remove small literals (the "smooth" parts of clauses); large literals (the "rough" parts of clauses) remain as a difficulty. Thus we are particularly interested in rough resolvents because they might remove large literals needed for the proof. Rough resolvents can be obtained by resolving upon *maximum literals*.

Definition 1 A literal L in a clause C is an *absolute maximum literal* if, for *all* σ, the size of $L\sigma$ is larger than or equal to that of any literal in $C\sigma$; L is a *likely maximum literal* if for *some* σ, $L\sigma$ is larger than or equal to any literal in $C\sigma$.

A clause can only have one of those two kinds of maximum literals. For example, in clause

$$\{ d(v, k(w, u)), \sim d(v, u), \sim d(v, w) \}$$

where u, v and w are variables, $d(v, k(w, u))$ is the only absolute maximum literal; in clause

$$\{ d(u, q(w, v)), \sim d(f(u, v), w), \sim d(v, w) \}$$

both $d(u, q(w, v))$ and $\sim d(f(u, v), w)$ are absolute maximum literals; clause

$$\{ d(u, v), \sim d(v, u) \}$$

has two absolute maximum literals; in clause

$$\{ d(u, w), \sim d(u, v), \sim d(v, w) \}$$

all three literals are likely maximum literals.

Definition 2 A *rough resolution step* involves simultaneously resolving upon maximum literals $L_1, \ldots, L_n$ of a clause C (called *nuclei*) with some *absolute* maximum literals in other clauses $C_1, \ldots, C_n$ (called *electrons*). A *rough resolvent* is obtained from a rough resolution step.

Nuclei can have absolute or likely maximum literals; electrons can only have absolute maximum literals. Absolute maximum literals in a nucleus are all resolved upon at the same time in one single rough resolution step. This is based on the observation that absolute maximum literals should be eventually all removed since they are always the largest in any instances, and intermediate resolvents might not be saved due to non-negative growths (to be discussed in next section).

Resolving on likely maximum literals in a clause is difficult to handle because it might not delete large literals and, at the same time, could generate too many useless resolvents. For example, the transitivity axiom

$$\sim(x \leq y) \vee \sim(y \leq z) \vee (x \leq z)$$

contains three likely maximum literals and often generates too many clauses during a resolution proof. Their role is obscure in rough resolution. We have used two strategies to do rough resolution on likely maximum literals.

The first is to apply the following heuristics to simultaneously resolve on more than one likely maximum literal: if there are two likely maximum literals in a clause, we resolve upon them one at a time; if there are more than two likely maximum literals, we also resolve upon each possible pair of two of them at the same time. This is based on the observation that usually only few likely maximum literals will become the largest in the ground instances.

Another important strategy is to require that any likely maximum literal resolved upon should still be a likely (or absolute) maximum literal after the proper substitution is applied. Otherwise the resolvent is discarded because it introduces larger literals from those not resolved upon.

For example, consider the clause

$$C = \{ \sim p(x, y), \sim p(y, z), p(x, z) \}$$

where all literals are likely maximum literals. Suppose first two literals are resolved with $p(x, a)$ and $p(a, z)$ respectively with substitution $\theta = \{ y \rightarrow a \}$. Such resolution is not allowed since none of $p(x, a)$ and $p(a, z)$ are maximum literals in $C\theta$, and the literal not resolved upon, $p(x, z)$, becomes an absolute maximum literal in $C\theta$.

Retaining Resolvents

Rough resolution only resolves upon maximum literals. Since usually there are not many absolute maximum literals in a clause, the number of resolvents are greatly reduced. However there are still too many resolvents if no further restriction is applied. In this section we discuss one strategy that we use to retain resolvents more selectively.

From the rough resolvents, we prefer those *smaller* than the parents clauses. Ordering on clauses is needed here and we use ordering of the multisets of all literal sizes in a clause.

Definition 3 $|C|$ is the multiset of sizes of all literals in C. Difference $|C_1| - |C_2|$ is $c1 - c2$ where $c1$ and $c2$ are the largest elements (0 if the multiset is empty) in $|C_1|$ and $|C_2|$ respectively after common elements are deleted.

For example, for clause $C = \{ p(x), p(y), \sim q(x, y) \}$, $|C| = \{ 3, 2, 2 \}$; $\{ 3, 2, 2 \} - \{ 3, 2 \} = \{ 2 \} - \{ \} = 2$ and $\{ 4 \}$-$\{ 5, 2 \} = -1$.

Definition 4 Suppose in a rough resolution step, resolvent R is obtained from clauses $C_1, \ldots, C_n$. The *growth* of R is $|R|-$ maximum of $(|C_1|, |C_2|, \ldots, |C_n|)$.

Such multiset idea is only used to compute *growth* of a rough resolvent. In other situations the size of a clause is still the largest literal size.

Growth is a useful measurement to retain resolvents from resolving upon likely maximum literals. Intuitively growth indicates the size growth of a rough resolvent relative to the parent clauses before substitution is applied. If growth is negative, the resolvent is smaller than the largest parents clause and "progress" has been made to reduce the number of large literals. On the other hand, if the growth is zero or positive, the resolvent is of the same length or larger than the largest parent clause. There is no progress from this rough resolution step and it is not useful to keep the resolvent.

The algorithm of rough resolution is described in Fig. 1. Resolvents of non-negative growth are retained only when there are no resolvents with negative growth. The procedure repeats until a proof is found. The resolvents are used in semantic hyper-linking in a limited way because many resolvents could be generated. Reasonable time bound is set on the use of rough resolvents in semantic hyper-linking.

The collaboration of rough resolution and semantic hyper-linking is not explicitly shown in Fig. 1. Basically rough resolution executes for some amount of time then stops, and new resolvents are used in later semantic hyper-linking; when executed again, rough resolution picks up from where it left off and continues.

The rough resolvents with negative growth are always kept; among the rough resolvents with non-negative growth, only the smallest (in terms of the largest literal in the resolvent) are saved, if necessary. This allows the resolvents with non-negative growth to be used in a controlled manner.

With restrictions on how rough resolution is applied (by resolving upon maximum literals simultaneously) and how resolvents are retained (based on growth), much less resolvents are retained than those in other similar resolution strategy. And we have found the algorithm practical and useful when used with semantic hyper-linking.

```
Algorithm Rough Resolution
begin
 loop
   for each clause $C$ with absolute maximum literals
     Obtain a new rough resolvent $R$ using $C$ as nucleus
     if $R$ has a negative growth
     then
       save $R$ permanently as a new clause
     else
       save $R$ temporarily
 until there are no new resolvents with growth $< 0$
 for each clause $C$ having likely maximum literals
   loop
     Obtain a new rough resolvent $R$ using $C$ as nucleus
     if $R$ has negative growth
     then
       save $R$ permanently as a new clause
     else
       save $R$ temporarily
   until no new rough resolvents can be generated
 if in last loop no rough resolvent was generated
     with negative growth
 then
   for each smallest temporarily saved rough
       resolvent $R$
     save $R$ permanently as a new clause
end
```

Figure 1: Algorithm: Rough Resolution

An Example

Bledsoe gave LIM+ problems in [Bledsoe, 1990] as challenge problems for automated theorem provers. Because of the large search space they might generate, LIM+ problems are difficult for most theorem provers. However, they are not difficult for Str+ve [Hines, 1992] which has built in inequality inference rules for densed linear ordering.

In this section we will look at the proof of LIM3 using rough resolution. Intermediate results from semantic hyper-linking are omitted.

As mentioned in the introduction, the correct ground instance of the goal clause has to be generated to obtain the proof. However, that ground instance contains a literal so large that semantic hyper-linking cannot generate it early enough in the search for the proof. As a result, the prover got lost even before correct goal instances are generated. Rough resolution helps to delete large literals by resolving upon them; smaller ground literals are generated by semantic hyper-linking. It is interesting to observe that the proof presented here does not need the term $min(d1(ha(e0)), d2(ha(e0)))$ which is essential to human proofs.

We only list clauses used in the proof; literals in boxes are maximum literals and only clauses 14 and 15 (from the same transitivity axiom) have likely maximum literals. Each rough resolution step is denoted by list of clause numbers, with nuclei in boxes. For example, $(\boxed{14},3,17)$ denotes a rough resolution step using clause 14 as nucleus and clauses 3 and 17 as electrons.

Input clauses:

3: $\{\boxed{lt(ab(pl(X,Y)),pl(ab(X),ab(Y)))}\}$

8: $\{\boxed{lt(ab(pl(f(Z),ng(f(a)))),E)}, lt(E,o), \sim lt(ab(pl(Z,ng(a))),d1(E))\}$

9: $\{\boxed{lt(ab(pl(g(Z),ng(g(a)))),E)}, lt(E,o), \sim lt(ab(pl(Z,ng(a))),d2(E))\}$

10: $\{\boxed{lt(ab(pl(xs(D),ng(a))),D)}, lt(D,o)\}$

13: $\{\boxed{lt(X,Y)}, \boxed{lt(Y,X)}\}$

14: $\{\boxed{\sim lt(X,Y)}, \boxed{\sim lt(Y,Z)}, lt(X,Z)\}$

15: $\{\boxed{\sim lt(X,Y)}, \boxed{lt(X,Z)}, \sim lt(Y,Z)\}$

17: $\{\boxed{lt(pl(X,Y),Z)}, \sim lt(X,ha(Z)), \sim lt(Y,ha(Z))\}$

18: $\{lt(D,o),$
$\boxed{\sim lt(ab(pl(pl(f(xs(D)),ng(f(a))),pl(g(xs(D)),ng(g(a))))),e}$

The proof:

21: $(\boxed{14},3,17)$
$\{\boxed{lt(ab(pl(X,Y)),Z)}, \sim lt(ab(X),ha(Z)), \sim lt(ab(Y),ha(Z))\}$

24: $(\boxed{18},21)$
$\{\boxed{\sim lt(ab(pl(f(xs(D)),ng(f(a)))),ha(e0))}$
$\boxed{\sim lt(ab(pl(g(xs(D)),ng(g(a)))),ha(e0))}, lt(D,o)\}$

29: $(\boxed{24},8,9)$
$\{\boxed{\sim lt(ab(pl(xs(D),ng(a))),d1(ha(e0)))},$
$\boxed{\sim lt(ab(pl(xs(D),ng(a))),d2(ha(e0)))},$
$lt(ha(e0),o), lt(D,o)\}$

97: $(\boxed{15},29,10)$
$\{\boxed{\sim lt(ab(pl(xs(D),ng(a))),d2(ha(e0)))},$
$\sim lt(D,d1(ha(e0))), lt(ha(e0),o), lt(D,o)\}$

99: $(\boxed{15},29,10)$
$\{\boxed{\sim lt(ab(pl(xs(D),ng(a))),d1(ha(e0)))},$
$\sim lt(D,d2(ha(e0))), lt(ha(e0),o), lt(D,o)\}$

139: $(\boxed{10},97)$
$\{\boxed{\sim lt(d2(ha(e0)),d1(ha(e0)))},$
$lt(ha(e0),o), lt(d2(ha(e0)),o)\}$

($lt(ha(e0),o)$ and $lt(d2(ha(e0)),o)$ are then unit deleted)

141: $(\boxed{10},99)$
$\{\boxed{\sim lt(d1(ha(e0)),d2(ha(e0)))},$
$lt(ha(e0),o), lt(d1(ha(e0)),o)\}$

($lt(ha(e0),o)$ and $lt(d1(ha(e0)),o)$ are then unit deleted)

$(\boxed{13}, 139, 141)$
$\{\}$ is obtained and a proof is found.

Unit $\{\sim lt(ha(e0), o)\}$ is generated by model filtering [Chu and Plaisted, 1993] in semantic hyper-linking. Or it can be generated by UR resolution from clause 4 and 7. It is then used with clause 5 to generate $\{\sim lt(d1(ha(e0)), o)\}$; with clause 6 to generate $\{\sim lt(d2(ha(e0)), o)\}$.

4: $\{\sim lt(e0, o)\}$
5: $\{lt(X, o), \sim lt(d1(X), o)\}$
6: $\{lt(X, o), \sim lt(d2(X), o)\}$
7: $\{lt(X, o), \sim lt(ha(X), o)\}$

None of the above three units can be generated by rough resolution. This shows the collaboration of rough resolution with semantic hyper-linking to find the proof.

Results

We have implemented a prover in Prolog. Experiment results show that rough resolution indeed improves the prover and often helps to find proofs faster. Table 1 lists some results (in seconds) that show rough resolution is in general a useful technique used with semantic hyper-linking. AM8 is the attaining maximum (or minimum) value theorem in analysis [Bledsoe, 1983]; LIM1–3 are the first three LIM+ problems proposed by Bledsoe [Bledsoe, 1990]; IMV is the intermediate value theorem in analysis [Bledsoe, 1983]; I1, IP1, P1 and S1 are four problems in implicational propositional calculus [Lukasiewicz, 1948, Pfenning, 1988]; ls37 is the theorem that $\forall x \in$ a ring R, $x * 0 = 0$ where 0 is the additive identity; SAM's lemma is a lemma presented in [Guard *et al.*, 1969]; wos15 proves the closure property of subgroups; wos19 is the theorem that subgroups of index 2 are normal; wos20 is a variant of wos19; wos21 is a variant of ls37; and ExQ problems (including wos31) are three examples from quantification theory [Wang, 1965].

LIM1–3 are considered simpler in [Bledsoe, 1990]. Prover STR+VE, with built in inference rules for densed linear ordering, can easily prove all LIM+ problems. However, few other general-purpose theorem provers can prove LIM1–3 (especially LIM3). METEOR [Astrachan and Loveland, 1991] can prove all three by using special guidance. OTTER [McCune, 1990] and CLIN [Lee and Plaisted, 1992] could not prove any LIM+ problem.

Conclusions

Rough resolution is incomplete but it is useful when used with other complete methods that prefer small clauses. It helps to focus on removing "large" part of the proofs. In particular, we have found that semantic hyper-linking and rough resolution conceptually work well together: semantic hyper-linking solves the "small" and "non-Horn" part of the proof; UR resolution [Chu and Plaisted, 1993] solves the "Horn" part of the proof; and rough resolution solves the "large" part of the proof.

Rough resolution is powerful enough to help semantic hyper-linking prove some hard theorems which cannot be obtained otherwise. However we have observed that likely maximum literals are the source of rapid search space expansion in some hard theorems like LIM4 and LIM5. Future research includes further investigation of the role of likely maximum literals and avoid generating unnecessary resolvents from resolving upon likely maximum literals. One possible direction is applying rough resolution idea on paramodulation since equality axioms often contain likely maximum literals. Also, focusing on relevant resolvents should be another important issue to be addressed.

Problem	*with rough resolution*	*without rough resolution*
AM8	2127.3	—*
LIM1	83.0	—
LIM2	63.3	—
LIM3	534.8	—
IMV	374.5	49.8
IPC I1	1.9	344.6
IPC IP1	29.2	1614.1
IPC P1	65.8	—
IPC S1	2.9	805.4
ls37	138.7	65.2
SAM's Lemma	146.5	95.5
wos15	282.7	—
wos19	407.5	1183.2
wos20	5715.7	—
wos21	783.7	93.4
wos31 (ExQ1)	203.5	39.2
ExQ2	84.0	130.1
ExQ3	140.3	112.0

*"—" indicates the run is aborted after either running over 20,000 seconds or using over 30 Megabyte memory

Table 1: Proof results using rough resolution with semantic hyper-linking

References

Astrachan, O.L. and Loveland, D.W. 1991. METEORs: High performance theorem provers using model elimination. In Boyer, R.S., editor 1991, *Automated Reasoning: Essays in Honor of Woody Bledsoe.* Kluwer Academic Publishers.

Ballantyne, A. M. and Bledsoe, W. W. 1982. On generating and using examples in proof discovery. *Machine Intelligence* 10:3–39.

Bledsoe, W. W. 1983. Using examples to generate instantiations of set variables. In *Proc. of the 8 th IJCAI*, Karlsruhe, FRG. 892–901.

Bledsoe, W. W. 1990. Challenge problems in elementary calculus. *J. Automated Reasoning* 6:341–359.

Chu, Heng and Plaisted, David A. 1992. Semantically guided first order theorem proving using hyperlinking. Manuscript.

Chu, Heng and Plaisted, David A. 1993. Model finding strategies in semantically guided instance-based theorem proving. In Komorowski, Jan and Raś, Zbigniew W., editors 1993, *Proceedings of the 7h International Symposium on Methodologies for Intelligent Systems.* To appear.

Guard, J.; Oglesby, F.; Bennett, J.; and Settle, L. 1969. Semi-automated mathematics. *J. ACM* 16(1):49–62.

Hines, L. M. 1992. The central variable strategy of Str+ve. In Kapur, D., editor 1992, *Proc. of CADE-11*, Saratoga Springs, NY. 35–49.

Lee, Shie-Jue and Plaisted, David. A. 1992. Eliminating duplication with the hyper-linking strategy. *J. Automated Reasoning* 9:25–42.

Lukasiewicz, Jan 1948. The shortest axiom of the implicational calculus of propositions. In *Proceedings of the Royal Irish Academy.* 25–33.

McCune, William W. 1990. *OTTER 2.0 Users Guide.* Argonne National Laboratory, Argonne, Illinois.

Pfenning, Frank 1988. Single axioms in the implicational propositional calculus. In Lusk, E. and Overbeek, R., editors 1988, *Proc. of CADE-9*, Argonne, IL. 710–713.

Plaisted, David. A.; Alexander, Geoffrey D.; Chu, Heng; and Lee, Shie-Jue 1992. Conditional term rewriting and first-order theorem proving. In *Proceedings of the Third International Workshop on Conditional Term-Rewriting Systems*, Pont-à-Mousson, France. Invited Talk.

Robinson, J. 1965. A machine-oriented logic based on the resolution principle. *J. ACM* 12:23–41.

Wang, H. 1965. Formalization and automatic theorem-proving. In *Proc. of IFIP Congress 65*, Washington, D.C. 51–58.