Linear-Input Subset Analysis

Geoff Sutcliffe
Dep't of Computer Science, The University of Western Australia
Nedlands, Western Australia, 6009. Email : geoff@cs.uwa.oz.au

Abstract. There are syntactically identifiable situations in which reduction does not occur in chain format linear deduction systems, i.e. situations in which linear-input subdeductions are performed. Three methods of detecting these situations are described in this paper. The first method (Horn subset analysis) focuses on Horn input chains while the second (LISS analysis) and third (LISL analysis) are successive generalisations of the first method. A significant benefit that may be derived from detecting linear-input subdeductions is the applicability of a truth value deletion strategy in such subdeductions. The completeness of the deletion strategy is proved, and its efficacy indicated.

1. Introduction

The exponential size of the search space of the resolution procedure necessitates the use of refinements which restrict the search space. Many refinements of the resolution procedure have been developed, one category of which is the linear refinements. The earliest linear deduction systems were the R 3 refinement [Luckham, 1970], s-linear resolution [Loveland, 1970], and the strategy of preference of a 'new' conjunction [Zamov, 1969]. After these initial three systems, two streams of development are evident. One stream developed refinements based on the merging restriction [Andrews, 1968], whilst the other developed the chain format systems. The results presented in this paper are for chain format linear deduction systems.

The first chain format systems developed were the Model Elimination procedure [Loveland, 1969] and Linear resolution with Selection function (SL-resolution) [Kowalski, 1971]. Subsequent chain format systems include Ordered Linear deduction [Chang, 1973], the Graph Construction procedure [Shostak, 1976], Selective Linear Model deduction [Brown, 1974], and Linear resolution with Unrestricted Selection based on Trees (LUST)-resolution [Minker 1982]. There have also been many implementations of chain format systems, e.g. [Fleisig, 1974], [Stickel, 1986], [Sutcliffe, 1990], [Tarver, 1990]. The results presented in this paper are not immediately applicable to the Graph Construction procedure and Selective Linear Model deduction, due to their mechanisms for reusing portions of deductions. However, a simple modification to these deduction systems (as discussed in the conclusion) makes the results applicable.

Linear-input deduction is a refinement of linear deduction which does not permit any form of ancestor resolution, i.e. linear deduction using only the extension operation. Linear-input deduction is complete for sets of Horn clauses, but is not complete for sets of non-Horn clauses. [Ringwood, 1988] provides an interesting synopsis and references for the history of linear-input deduction systems. The use of reduction makes linear deduction complete for sets of non-Horn clauses. There are syntactically identifiable situations in which reduction does not occur in linear deduction systems, i.e. situations in which linear-input subdeductions are performed, and three methods of analysing sets of input chains have been developed for detecting these situations. The first method focuses on Horn input chains while the second and third are successive generalisations of the first method. Other work [Wakayama, 1990] has also noted that ancestor resolution and factoring are not always necessary for obtaining a refutation when the input set is non-Horn. The methods of analysis presented here are, however, more general than Wakayama's, which is restricted to entire input sets.

The detection of linear-input subdeductions is useful for (and was largely motivated by) the imposition of truth value deletion in linear deduction systems. The use of semantic information appears to have the potential to significantly improve the performance of deduction systems. This has been noted in the literature, e.g. "An emphasis on semantics rather than on syntax has far greater potential for producing a dramatic impact on the power of automated reasoning programs" [Wos, 1988, p. 257] (Research towards a "semantically orientated strategy" is Problem 5 in [Wos, 1988]), and "... if searches in symbolic computation are not to fall prey to combinatorial explosion, they must incorporate domain-specific knowledge in such a way so as to give direction to the search." [McRobbie, 1988, p. 198]. Up to this point, truth value deletion has been considered incompatible with linear deduction. The isolation of linear-input subdeductions, through linear-input analysis, makes it possible to use truth value deletion in linear deductions. This is described in section 3 of this paper.

2. Linear-Input Analysis

2.1 Horn Subset Analysis

It has been noted that "...in many proofs, most of the input clauses are Horn clauses..." [Plaisted, 1982, p. 231]. Examination of linear refutations of some input sets reveals that once the positive B-literal of a Horn input chain has been extended against, no reductions are performed until that B-literal (in the guise of an A-literal) is truncated. Horn subset analysis detects these subdeductions.

The *input predicate set* of a set of input chains contains the predicate structures[1] that appear in the input set. To detect situations in which reduction does not occur in a linear deduction from a negative top chain in an input set (lemma 1.1 relies on a negative top chain), the *Horn subset* of the input predicate set is extracted. A predicate structure is in the Horn subset iff (i) it does not occur positively in a non-Horn input chain, and (ii) for every Horn input chain in which the predicate structure occurs positively, every predicate structure in the chain is in the Horn subset.

Example

The Horn subset of {`~r~p~q, ~pq, p~q, pq, r~t~s, t~u, u, s`}, with `~r~p~q` as the top chain, is {`r, t, s, u`}.

[1] A predicate structure is the predicate symbol and arity of a literal.

The Horn subset divides the input chains into two groups, based on whether or not all literals in the chain have predicate structures that are in the Horn subset. Any predicate structures, literals, or input chains which only contain predicate structures that are in the Horn subset, are called *Horn subset objects*.

In a linear deduction from a negative top chain no reduction against Horn subset literals is performed, and once a Horn subset B-literal is selected, no reduction against literals rightwards from the selected B-literal is performed until that B-literal (in the guise of an A-literal) is truncated. Further, only the positive B-literal of a Horn subset input chain is ever resolved against in an extension operation. These properties are now proved. (Concepts similar to those used here were informally introduced in [Sutcliffe, 1989].)

Lemma 1.1
In a linear deduction from a negative top chain : (i) no positive Horn subset A- or B-literal occurs in a centre chain. (ii) negative B-literals in Horn subset input chains are never resolved against in extension operations. (iii) no reduction against Horn subset A- or B-literals is performed.

The proof of part (i) is by contradiction. If a positive Horn subset A- or B-literal occurs in a centre chain then the A-literal immediately to its left must be a Horn subset A-literal, as its complement originates from the same input chain as the first literal. Further, the Horn subset A-literal to the left must be positive, for otherwise the first literal occurs positively in a non-Horn input chain. Iteratively, all the A-literals to the left of a positive Horn subset A- or B-literal must be positive. However, the leftmost A-literal in the centre chain must be negative as the top chain is negative. Contradiction. Hence : (i) no positive Horn subset A- or B-literal occurs in a centre chain. (ii) as there can only be negative Horn subset B-literals in a centre chain, negative B-literals in Horn subset input chains can never be resolved against in extension operations. (iii) as complementary Horn subset A- and B-literals cannot occur in any centre chain, no reduction against such literals is performed. **QED**

Theorem 1.2
In a linear deduction from a negative top chain : (i) every A- and B-literal to the right of a Horn subset A-literal in a center chain, is also a Horn subset literal. (ii) no reduction against A- and B-literals rightwards from a Horn subset A-literal is performed. (iii) once a Horn subset B-literal is selected, no reduction against literals rightwards from the selected B-literal is performed until that B-literal (in the guise of an A-literal) is truncated.

By lemma 1.1, once a Horn subset B-literal in a centre chain has been selected, it is necessarily extended against. From the definition of the Horn subset, the B-literals added to the centre chain in the extension are Horn subset B-literals. Therefore : (i) iteratively, every A- and B-literal to the right of the selected Horn subset B-literal (now an A-literal) is a Horn subset literal. (ii) by lemma 1.1, no reduction against A- and B-literals rightwards from a Horn subset A-literal is performed. (iii) the structure of chain format linear deductions and part (ii) ensure that no reduction against literals rightwards from the selected B-literal is performed until that B-literal (in the guise of an A-literal) is truncated. **QED**

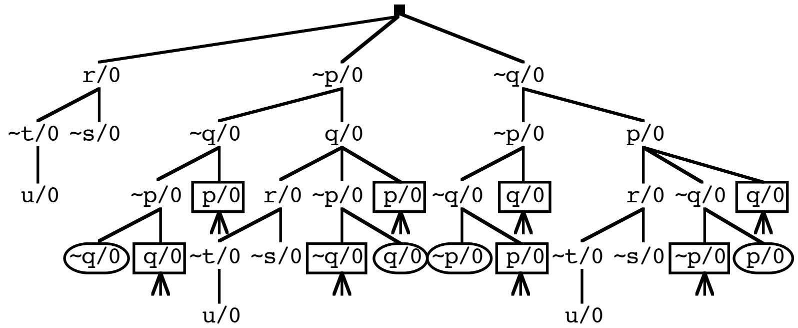
2.2 Linear-Input Subsets for Literal Structures

Horn subset analysis focuses on Horn input chains, and does not provide adequate analysis for input chains which are non-Horn but are Horn in a renaming of the input set. Many results based on the polarity of literals can be generalised to be based on a division of the literal structures[2] that appear in the input set, e.g. P_1 resolution [Robinson, 1965] generalises to P_P resolution [Meltzer, 1966], hyper-resolution [Robinson, 1965] generalises to AM-clashes [Slagle, 1967]. Similarly, Horn subset analysis generalises to results for non-Horn chains, in the form of Linear-Input Subset for literal Structures (LISS) analysis. The generalisation from Horn subsets to LISSs comes at the cost of a more complex analysis. Rather than a direct examination of the input set, LISS analysis requires examination of an abstraction of the deduction search tree.

For an input chain, the corresponding *chain structure set* contains the literal structures that occur in the chain. To detect situations in which reduction does not occur in a deduction from a chosen top chain, the *linear-input subset* of the literal *structures* that occur in the input set is extracted. This is done by building an *extension tree* whose nodes are literal structures. The extension tree has a mythical root whose offspring are the elements of the chain structure set corresponding to the chosen top chain. For each non-root literal structure in an extension tree, its offspring are those literal structures that (i) are in chain structure sets that contain a literal structure complementary to the parent literal structure, (ii) are not the complementary literal structure, and (iii) do not have themselves as ancestors in the extension tree unless, between the offspring and the ancestor, there exists a literal structure which does not have itself as an ancestor above the offspring's ancestor. A literal structure is in the LISS iff for every occurrence in the extension tree (i) it is not complementary to an ancestor, and (ii) all of its descendants are in the LISS.

Example

The first few level of the LISS tree for {`r~p~q, ~pq, p~q, pq, ~r~t~s, tu, ~u, s`}, with `r~p~q` as the top chain, are :



[2] A literal structure is the sign, predicate symbol and arity of a literal.

> Circled literal structures are leaves of the tree, as dictated by item (iii) in the definition of these trees. Boxes literal structures are complementary to an ancestor. The lower levels of the tree reveal no new information. The LISS is thus `{r, ~t, u, ~s}`. No Horn subset exists for this top chain, as it is non-negative. With `~r~t~s` as the top chain, the Horn subset is `{s}` and the LISS is `{~t, u, ~s}`.

Any literal structures or literals which only contain literal structures that are in the LISS, are called *LISS objects*.

In a linear deduction from a chosen top chain, no reduction against LISS B-literals is performed, and once a LISS B-literal is selected, no reduction against literals rightwards from the selected B-literal is performed until that literal (in the guise of an A-literal) is truncated. These properties are now proved.

Lemma 2.1
In a linear deduction from a chosen top chain no reduction against LISS A- or B-literals is performed.

The root to tip sequence of literal structures in a branch of the extension tree corresponds to possible left to right sequences of A- and B-literal structures in centre chains of a deduction from the chosen top chain. Each node corresponds to a possible A-literal in a centre chain, and literal structures that are lower in the branch correspond to possible B-literals to the left of that A-literal in a centre chain. Therefore : (i) no LISS B-literal in a centre chain has a structure complementary to an A-literal to its left (LISS definition part (i)), and no reduction against LISS B-literals is performed. (ii) no LISS A-literal has a B-literal with a complementary structure to its right (LISS definition part (ii)), and no reduction against LISS A-literals is performed. **QED**

Theorem 2.2
In a linear deduction from a chosen top chain : (i) every A- and B-literal to the right of a LISS A-literal in a center chain, is also a LISS literal. (ii) no reduction against A- or B-literals rightwards from a LISS A-literal is performed. (iii) once a LISS B-literal is selected, no reduction against literals rightwards from the selected B-literal is performed until that B-literal (in the guise of an A-literal) is truncated.

The proof is analogous to that of theorem 1.2.

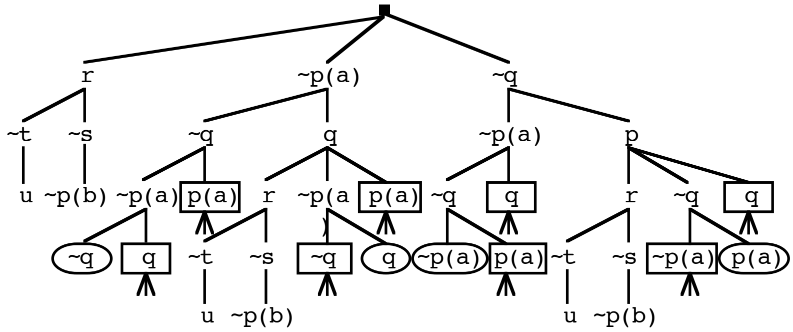
2.3 Linear-Input Subsets for Literals

In building the extension tree, LISS analysis makes the assumption that every pair of literals with complementary literal structures can unify. A more accurate analysis is possible, by working directly with the literals in the input set. Linear-Input Subset for Literals (LISL) analysis does this.

To detect situations in which reduction does not occur in a deduction from a chosen top chain, the *linear-input subset* of the *literals* in the input set is extracted. This is done by building an extension tree whose nodes are literals from the input set. The method used is similar to that for LISS analysis. For each non-root parent literal in a LISL extension tree, its offspring are those literals that (i) are in chains that contain a literal complementarily unifiable with the parent literal, (ii) are not the complementarily unifiable literal, and (iii) do not have themselves as ancestors in the extension tree unless, between the offspring and the ancestor, there exists a literal which does not have itself as an ancestor above the offspring's ancestor. A literal structure is in the LISL iff for every occurrence in the extension tree (i) it is not complementarily unifiable with an ancestor, and (ii) all of its descendants are in the LISL. Note that although the extension tree uses unifiability, unification is never consummated.

Example

The first few levels of the LISL tree for {`r~p(a)~q, ~p(a)q, p(a)~q, p(a)q, ~r~t~s, tu, ~u, s~p(b), p(b)`}, with `r~p(a)~q` as the top chain, are :



Circled literal structures are leaves of the tree, as dictated by item (iii) in the definition of these trees. Boxes literal structures are complementarily unifiable with an ancestor. The lower levels of the tree produce no new information. The LISL is thus {`r`^{1}`, ~t`^{5}`, u`^{6}`, ~s`^{5}`, ~p(b)`^{8}} (where the superscripts indicate the chain number that the literal is in). The LISS is {`~t, u`}.

The results and theorem proofs for LISL analysis are analogous to those for LISS analysis. The results are simply stated here.

Lemma 3.1

In a linear deduction from a chosen top chain no reduction against LISL A- or B-literals is performed.

Theorem 3.2
In a linear deduction from a chosen top chain : (i) every A- and B-literal to the right of a LISL A-literal in a center chain, is also a LISL literal. (ii) No reduction against A- or B-literals rightwards from a LISL A-literal is performed. (iii) Once a LISL B-literal is selected, no reduction against A- and B-literals rightwards from the selected B-literal is performed until that B-literal (in the guise of an A-literal) is truncated.

2.4 Taking Advantage of Linear-Input Subsets

The above results show that once a Horn subset/LISS/LISL B-literal (henceforth, Horn subset/LISS/LISL objects will be referred to generically as *linear-input objects*) has been selected, a linear deduction system starts a linear-input subdeduction. The subdeduction end when that B-literal (in the guise of an A-literal) is truncated. The selected B-literal is called the *top literal* of the subdeduction.

In a linear-input subdeduction the reduction operation can be explicitly ignored, so that no effort is expended attempting to find A-literals to reduce against. If Horn subset analysis is used then only the positive literals of Horn subset input chains need ever be considered when searching for a suitable input chain in an extension operation. A more significant benefit that may be derived from linear-input analysis is the completeness of a truth value deletion strategy in linear-input subdeductions. Truth value deletion strategies reject clauses/chains in a deduction, based upon their interpretation value in a given interpretation. In linear-input deductions, deletion of center chains which are not FALSE in all models of the side chains used, is a complete strategy (see [Bundy, 1983, p 147] for example). This truth value deletion system can be transferred to linear-input subdeductions.

3. Truth Value Deletion in Linear Deduction Systems

Def'n : A *side chain model* of a deduction is an interpretation that is a model of the side parent chains used in the deduction.

Def'n : The *rightwards subchain* of a literal in a center chain consists of the literal and all literals to its right.

A truth value deletion system which requires all rightwards subchains of the top literal in a linear-input subdeduction to be interpreted as FALSE, in all side chain models of the subdeduction, is complete. The system arises from the following results.

Lemma 4.1
In a linear refutation in which only extension and truncation operations are performed, all centre chains are interpreted as FALSE in all side chain models of the refutation. (In determining the interpretation value of a centre chain, only B-literals are considered.)

In this situation a linear refutation is reduced to a linear-input refutation. The lemma then follows directly from [Bundy, 1983, Thm 2, p 147]. **QED**

Lemma 4.2
In a linear-input subdeduction, all rightwards subchains of the top literal are interpreted as FALSE in all side chain models of the subdeduction. (In determining the interpretation value of a rightwards subchain, only B-literals are considered.)

Let $C_1R_1,...,C_{n-1}R_{n-1},C_n$ be the centre chains of a linear-input subdeduction, so that R_1 is the top literal, each R_i is a rightwards subchain of the top literal, and R_{n-1} is the top literal in the guise of an A-literal. Theorems 1.2, 2.2 and 3.2 show that no literal in any C_i is used in deduction operations that affect the R_i. Therefore there is a refutation from the top chain R_1 using only extension and truncation operations. By lemma 4.1, all the R_i are interpreted as FALSE in all side chain models of the refutation. Thus in the linear-input subdeduction all the R_i are interpreted as FALSE in all side chain models of the subdeduction. **QED**

Lemma 4.3
In a linear-input subdeduction, all rightwards subchains of the top literal are simultaneously interpreted as FALSE in all side chain models of the subdeduction. (In determining the interpretation value of a rightwards subchain, only B-literals are considered.)

Consider a ground universe instance of a linear-input subdeduction. By lemma 4.2 all rightwards subchains of the top literal are (because they are ground) simultaneously interpreted as FALSE in all side chain models of the ground subdeduction. Side chain models of the original non-ground subdeduction are also side chain models of the ground subdeduction. All rightwards subchains of the top literal in the original subdeduction therefore simultaneously have ground universe instances that are interpreted as FALSE in all side chain models of the original subdeduction. Thus, in a linear-input subdeduction, all rightwards subchains of the top literal are simultaneously interpreted as FALSE in all side chain models of the subdeduction. **QED**

Theorem 4.4
In a linear-input subdeduction, all rightwards subchains of the top literal are simultaneously interpreted as FALSE in all side chain models of the subdeduction. (In determining the interpretation value of a rightwards subchain, all literals, including A-literals, are considered.)

In a linear-input subdeduction, all A-literals in rightwards subchains of the top literal also occur as B-literals in ancestor rightwards subchains. The ancestor rightwards subchains are simultaneously subject to the truth value restriction. Therefore lemma 4.3 can be extended to include A-literals when determining interpretation values. **QED**

Theorem 4 establishes the completeness of a truth value deletion system for chain format linear deduction systems when a linear-input subdeduction is being performed. The deletion system is called the *rightwards subchain* truth value deletion system. The imposition of the rightwards subchain system relies on being able to predict which input chains can be used as side chains of each linear-input subdeduction, so that side chain models can be supplied. This property is called *side chain predictability*. If Horn subset analysis is used then side chain predictability is available immediately, as only non-negative Horn subset input chains are used as side parent chains. For LISS analysis, side chain predictability is obtained by inspection of the extension tree. For a linear-input subdeduction from a given top literal, the input chains that may be used as side parent chains are those that were used in building extension subtrees rooted at the LISS element corresponding to the top literal. Thus a set of possible side parent chains is associated with each LISS element. There are then two options for building side chain models :

- Different side chain models may be built for each LISS element, based upon the associated input chains. Although this may require significant effort, there may be some benefit in constructing models that are local to linear-input subdeductions, as the models need only reflect truth value information relevant to the subdeductions. As is noted in [Plaisted, 1982, p 238], "This is interesting because it corresponds to the fact that in the human theorem proving process attention is given to various specialized models at various stages of the proof".
- Side chain models of the union of the sets associated with the LISS elements may be built. This approach is only possible if the union is a proper subset of the input set. If all input chains are possible side parent chains, the LISS subset may be reduced by adding a new condition for membership of the LISS subset : (iii) the chosen top chain is not used in forming any descendant of the literal structure. If this condition is added, then the top chain of the deduction cannot be a side parent chain in any linear-input subdeduction, and models of the union are possible. If this latter approach only excludes the top chain from being a side parent chain then an additional truth value restriction, that requires the top chain of the deduction to be interpreted as FALSE in the side chain models, may be imposed. This is possible because at least one instance of one input chain used in a refutation must be interpreted as FALSE in every truth value interpretation of an unsatisfiable input set.

Side chain predictability for LISL analysis is the same as for LISS analysis.

4. Conclusion

LISS and LISL analysis have been found to be significantly more effective than Horn subset analysis for finding linear-input subdeductions. An advantage of the generalisations is that no restrictions are placed on the nature of the top chains of deductions. LISS analysis and the truth value deletion system have been implemented as part of the Semantically Guided Linear Deduction (SGLD) system [Sutcliffe, 1992]. The generation of the LISS is a simple iterative task, and is done before deductions are started. Truth value deletion is then applied as appropriate.

Following are some performance figures, for SGLD, which indicate the efficacy of the linear-input analysis and truth value deletion. Results are given for each of SGLD's search strategies, and for two initial bounds for the consecutively bounded depth first search used. The results are in the form <total number of deduction operations> : <time taken in seconds>. The column labelled LIA indicates whether or not linear-input analysis and truth value deletion have been used.

The first example is a program verification problem, similar to the shortburst and burstall problems from Reboh [1972]. This problem is, however, non-Horn. The truth value interpretation used, for the rows where LIA is 'Yes', knows the structure of the program's state space.

Search Style	LIA	Initial Depth Bound	
		Top chain length	6
Literal Selected	No	177 : 3	84 : 2
	Yes	75 : 1	22 : <1
Literal Ordered	No	199 : 3	100 : 2
	Yes	95 : 2	38 : 1
Cell Selected	No	307 : 5	127 : 2
	Yes	156 : 3	55 : 1
Cell Ordered	No	307 : 5	127 : 2
	Yes	156 : 3	55 : 1

Table 1. Results for the Program Verification Problem

The second example is the second group theory problem in [Chang, 1970] - In an associative system with an identity element, if the square of every element is the identity, then the system is commutative. The truth value interpretation used is an Abelian group of four elements (including the identity), which conforms to the hypotheses of the problem. Here the linear-input analysis finds that the entire deduction must be linear-input, so that truth value deletion is applied throughout. As the input set is Horn, this is not surprising. However, it does indicate that linear-input analysis encompasses detection of Horn-ness.

Search Style	LIA	Initial Depth Bound	
		Top chain length	7
Literal Selected	No	1413 : 23	1227 : 20
	Yes	725 : 13	555 : 10
Literal Ordered	No	1389 : 23	1203 : 21
	Yes	734 : 13	564 : 10
Cell Selected	No	2890 : 50	2487 : 44
	Yes	1623 : 30	1236 : 23
Cell Ordered	No	2890 : 51	2487 : 44
	Yes	1623 : 30	1236 : 23

Table 2. Results for the Group Theory Problem

The final example is Schubert's Steamroller problem. The truth value interpretation used is aware of sorts, knows that only animals eat, and that only two animals can be compared in size.

Search Style	LIA	Initial Depth Bound	
		Top chain length	12
Literal Selected	No	15669 : 287	1162 : 22
	Yes	3680 : 51	1343 : 22
Literal Ordered	No	15681 : 300	1005 : 19
	Yes	3728 : 52	1358 : 22
Cell Selected	No	60038 : 1128	61247 : 1512
	Yes	11030 : 230	4638 : 105
Cell Ordered	No	59229 : 1112	32634 : 786
	Yes	10973 : 230	5051 : 113

Table 3. Results for Schubert's Steamroller Problem

In all except for two combinations of strategy, the search space of SGLD is reduced (in some cases dramatically) by the use of linear-input analysis and truth value deletion. In most cases, the time taken is also reduced, in the best case by almost an order of magnitude. The user specified initial bounds (the righthand columns of results) are the values for which SGLD gives the best results without linear-input analysis and truth value deletion. This column illustrates that linear-input analysis and truth value deletion are of utility, even with a careful choice of deduction system parameters. The overall utility of the truth value deletion depends mainly on the manner in which the interpretation used is stored and manipulated. The implementation used in SGLD is moderately efficient.

The combination of linear-input analysis and truth value deletion clearly has the potential to be of benefit in chain format linear deduction systems. In some aspects it is analogous to the Simplified Problem Reduction Format's truth value deletion, the success of which "seems to have something to do with the fact that Horn clauses are common in typical problems." [Plaisted, 1982, p 238] The system presented here has, however, the potential for greater success as a more general notion than Horn-ness is used to determine when truth value deletion is applicable.

In the introduction it was noted that linear-input analysis is not immediately applicable in the Graph Construction procedure and Selective Linear Model deduction. The problem in these two systems is that reduction against C-literals (confusingly called A-literals in SLM) can be performed within, what would otherwise be, linear-input subdeductions. This added possibility is easily dealt with, in one of two ways. Firstly, all linear-input C-literals are inserted at the left most end of center chains, indicating that they are logical consequences of the side parent chains that participated in their production. Therefore the C-literals are TRUE in all models of such side parent chains. Lemma 4.1 is easily extended to cover reduction against such C-literals, and the truth value deletion system is still applicable. The second approach is to add such C-literals to the input set as unit chains, rather than inserting them at the left most end of the centre chain. If this option is chosen, then no reduction against linear-input C-literals can be performed, and all the results of this paper apply. The option of adding unit input chains also has other advantages, as the unit chains can be used throughout deductions.

References

Andrews P.B., Resolution with Merging, In *Journal of the ACM* 15(3), ACM Press, New York, NY, (1968), 367-381.

Brown F.M., SLM, Internal Memo #72, Department of Artificial Intelligence, University of Edinburgh, Edinburgh, Scotland, (1974).

Bundy A., *The Computer Modelling of Mathematical Reasoning*, Academic Press, London, England, (1983).

Chang C-L., The Unit Proof and the Input Proof in Theorem Proving, In *Journal of the ACM* 17(4), ACM Press, New York, NY, (1970), 698-707.

Chang C-L., and Lee R.C-T., *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, New York, NY, (1973).

Fleisig S., Loveland D.W., Smiley A.K., and Yarmush D.L., An Implementation of the Model Elimination Proof Procedure, In *Journal of the ACM* 21(1), ACM Press, New York, NY, (1974), 124-139.

Kowalski R., and Kuehner D., Linear Resolution with Selection Function, In *Artificial Intellience* 2, Elsevier Science, Amsterdam, The Netherlands, (1971), 227-260.

Loveland D.W., A Simplified Format for the Model Elimination Theorem-Proving Procedure, In *Journal of the ACM* 16(3), ACM Press, New York, NY, (1969), 349-363.

Loveland D.W., A Linear Format for Resolution, In Laudet M. et al. (Ed.), *Proceedings of the IRIA Symposium on Automatic Demonstration* (Versailles, France, 1968), Springer-Verlag, New York, NY, (1970), 147-162.

Luckham D., Refinement Theorems in Resolution Theory, In Laudet M. et al. (Ed.), *Proceedings of the Symposium on Automatic Demonstration* (Versailles, France, 1968), Springer-Verlag, New York, NY, (1970), 163-190.

McRobbie M.A., Meyer R.K., and Thistlewaite P.B., Towards Efficient "Knowledge-Based" Automated Theorem Proving for Non-Standard Logics, In Lusk E., Overbeek R. (Ed.), *Proceedings of the 9th International Conference on Automated Deduction* (Argonne, IL, 1988), (Goos G., Hartmanis J. (Ed.), *Lecture Notes in Computer Science 310*), Springer-Verlag, New York, NY, (1988), 197-217.

Meltzer B., Theorem-proving for computers: Some results on resolution and renaming, In *The Computer Journal* 8, The Britsh Computer Society, London, England, (1966), 341-343.

Minker J., and Zanon G., An Extension to Linear Resolution with Selection Function, In *Information Processing Letters* 14(4), Elsevier Science, Amsterdam, The Netherlands, (1982), 191-194.

Plaisted D.A., A Simplified Problem Reduction Format, In *Artificial Intelligence* 18, Elsevier Science, Amsterdam, The Netherlands, (1982), 227-261.

Reboh R., Raphael B., Yates R.A., Kling R.E., and Velarde C., Study of automatic theorem proving programs, Technical Note 72, Artificial Intelligence Center, SRI International, Menlo Park, CA, (1972).

Ringwood G.A., SLD: A Folk Acronym, In Moss C. (Ed.), *Logic Programming Newsletter* 2(1), Association for Logic Programming, London, England, (1988), 5-7.

Robinson J.A., Automatic Deduction with Hyper-resolution, In *International Journal of Computer Mathematics* 1, Gordon and Breach, London, England, (1965), 227-234.

Shostak R.E., Refutation Graphs, In *Artificial Intelligence* 7, Elsevier Science, Amsterdam, The Netherlands, (1976), 51-64.

Slagle J.R., Automatic Theorem Proving with Renamable and Semantic Resolution, In *Journal of the ACM* 14(4), ACM Press, New York, NY, (1967), 687-697.

Stickel M.E., A Prolog Technology Theorem Prover: Implementation by an Extended Prolog Compiler, In Siekmann J.H. (Ed.), *Proceedings of the 8th International Conference on Automated Deduction* (Oxford, England, 1986), (Goos G., Hartmanis J. (Ed.), *Lecture Notes in Computer Science 230*), Springer-Verlag, New York, NY, (1986), 573-587.

Sutcliffe G., Complete Linear Derivation Systems for General Clauses, In Wos L. (Ed.), *Association for Automated Reasoning Newsletter* (13), Association for Automated Reasoning, Argonne, Il, (1989), 3-4.

Sutcliffe G., A General Clause Theorem Prover, In Stickel M.E. (Ed.), *Proceedings of the 10th International Conference on Automated Deduction* (Kaiserslautern, Germany, 1990), (Siekmann J.H. (Ed.), *Lecture Notes in Artificial Intelligence 449*), Springer-Verlag, New York, NY, (1990), 675-676.

Sutcliffe G., The Semantically Guided Linear Deduction System, In Kapur, D. (Ed.), *Proceedings of the 11th International Conference on Automated Deduction* (Saratoga Springs, NY, 1992), Springer-Verlag, New York, NY, (1992).

Tarver M., An Examination of the Prolog Technology Theorem Prover, In Stickel M. (Ed.), *Proceedings of the 10th International Conference on Automated Deduction* (Kaiserslautern, Germany, 1990), (Siekmann J.H. (Ed.), *Lecture Notes in Artificial Intelligence 449*), Springer-Verlag, New York, NY, (1990), 322-335.

Wakayama T., and Payne T.H., Case-Free Programs: An Abstraction of Definite Horn Programs, In Stickel M. (Ed.), *Proceedings of the 10th International Conference on Automated Deduction* (Kaiserslautern, Germany, 1990), (Siekmann J.H. (Ed.), *Lecture Notes in Artificial Intelligence 449*), Springer-Verlag, New York, NY, (1990), 87-101.

Wos L., *Automated Reasoning - 33 Basic Research Problems*, Prentice-Hall, Englewood Cliffs, New Jersey, (1988).

Zamov N.K., and Sharonov V.I., On a class of strategies which can be used to prove theorems by the resolution principle, In (In Russian) (Ed.), *Issled, po konstruktivnoye matematikye i matematicheskoie logikye* III(16), National Lending Library Russian Translating Program 5857, Boston Spa, Yorkshire, (1969), 54-64.