TR-583

An Abductive Procedure for the CMS/ATMS

by
K. Inoue

August, 1990

© 1990, ICOT

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

An Abductive Procedure for the CMS/ATMS

Katsumi Inoue

ICOT Research Center
Institute for New Generation Computer Technology
Mita Kokusai Bldg. 21F
1-4-28 Mita, Minato-ku, Tokyo 108, Japan
inoue@icot.or.jp

Abstract

This paper concerns procedural semantics for a variety of ATMSs. Reiter & de Kleer view an ATMS as a kind of abduction in which the best explanation of a formula is defined as a minimal conjunction of hypotheses that explain the formula. However, they do not give any algorithm to compute such minimal explanations of a formula in their CMS that is a generalization of de Kleer's basic ATMS. In this paper, we use the notion of characteristic clauses to make precise definitions of the CMS and the ATMS and to produce a sound and complete abductive procedure based on an extension of linear resolution. By means of this abductive procedure, we give the CMS algorithms for computing minimal explanations in the interpreted approach and for updating them in the compiled approach. We then present algorithms for generating and updating labels of nodes in an extended ATMS that accepts non-Horn justifications and literal assumptions. Finally, how a variation of the abductive procedure can be used to answer queries for circumscription of ground theories is presented.

Keywords: ATMS, CMS, abduction, circumscription, linear resolution

1 Introduction

An assumption-based truth maintenance system (ATMS) [4] has been widely used when problems require reasoning in multiple contexts. However, this basic ATMS can only handle the restricted form of formulas, and is described algorithmically rather than declaratively or model-theoretically, and no proof of its correctness is given, so it is not obvious how to generalize or refine it. The motivation for this research was the desire to formalize generalizations of the ATMS within simple model and proof theories.

Recent investigations such as those of Reiter & de Kleer [22] and Levesque [16] show that there are strong connections between an ATMS and a logical account of *abduction* or *hypothesis generation* [20, 3, 9, 19]. An ATMS can be characterized by the following type of abduction:

Definition 1.1 Let W be a set of formulas, A a set of ground literals (called the *assumptions*), and G a closed formula. A conjunction H of elements of A is an *explanation of G from (W, A)* if (i) $W \cup \{H\} \models G$ and (ii) $W \cup \{H\}$ is satisfiable.

An explanation H of G from (W, A) is *minimal* if no proper sub-conjunct of H is an explanation of G from (W, A), that is, if no sub-conjunct H' of H satisfies $W \cup \{H'\} \models G$.

The ATMS is precisely intended to generate *all and only* minimal explanations [11]. In the ATMS terminology, the set of minimal explanations of a *node* G from the *justifications* W and the *assumptions* A is called the *label* of G, which is *consistent, sound, complete* and *minimal*. The *basic* ATMS [4] is restricted to accepting only Horn clause justifications and atomic assumptions. In the above declarative conditions for an ATMS, justifications can contain non-Horn clauses, and assumptions are allowed to be literals, so that this generalization covers de Kleer's various extended versions of the ATMS [5, 6, 7], Dressler's extended basic ATMS [8], and Reiter & de Kleer's clause management system (CMS) [22].

In spite of its usefulness in a wide range of applications, the algorithms for the ATMS in [4, 5, 6] have not yet been proved to be correct with respect to the declarative semantics. Although the CMS is well defined and the basic connection between resolution and the CMS processing is given in [22], there has not yet been any complete algorithm for computing labels of a formula for non-Horn theories in terms of popular and useful resolution methods. One of the problems is that although linear resolution is widely used and contains several restriction strategies, it is incomplete for consequence-finding [18] so that it cannot be directly used as an ATMS procedure.

The goal of this paper is to provide a sound and complete abductive procedure which solves the above problems for the CMS and ATMSs. In the remaining sections, we describe abduction as the problem of finding the *characteristic clauses* [1, 24] that are theorems of a given set of clauses and that belong to a distinguished sub-vocabulary of the language. We will give an extension of propositional linear resolution procedures which is complete for characteristic-clause-finding, then show ways in which to implement the CMS and the extended ATMS described above for both label generating (the interpreted approach) and label updating (the compiled approach). Since this extended ATMS can accept literal assumptions and non-Horn clauses, the methods described in this paper can also be applied to better implementations of theorem provers for closed world assumptions [1] and circumscription [21, 10] of ground theories, based on abductive procedures [13]. Unless otherwise specified, proofs for theorems and propositions are shown in Appendix.

2 Characteristic Clauses

We begin with some definitions and notations that will be used throughout this paper. We shall assume a propositional language with finitely many propositional symbols $\mathcal{A}$ and with logical connectives. The set of *literals* is defined as: $\mathcal{A}^{\pm} = \mathcal{A} \cup \neg \cdot \mathcal{A}$, where $\neg \cdot S$ means the set formed by taking the negation of each element in S. A *clause* is a finite set of literals, understood disjunctively; the empty clause is denoted by $\Box$. A *conjunctive normal form (CNF) formula* is a conjunction of clauses. Let C and C' be two clauses. $C - C'$ denotes a clause whose literals are those in the difference of C and C'. C is said to *subsume* C' if every literal in C occurs in C' ($C \subseteq C'$). In logical notation, C subsumes C' if $\models C \supset C'$. For a set of clauses Σ, by $\mu\Sigma$ or $\mu[\Sigma]$ we mean the set of clauses of Σ not subsumed by any other clause of Σ.

Definition 2.1 Let Σ be a set of clauses.
(1) A clause C is an *implicate* of Σ if $\Sigma \models C$. The set of implicates of Σ is denoted by $Th(\Sigma)$.
(2) The *prime implicates* of Σ are: $PI(\Sigma) = \mu Th(\Sigma)$.

We use the notion of *characteristic clauses*, which helps to analyze the computational aspect of ATMSs. While the idea of characteristic clauses was introduced by Bossu & Siegel [1] to evaluate a form of closed-world reasoning and was later generalized by Siegel [24], neither research focused on abductive reasoning or the ATMS. Informally speaking, characteristic clauses are intended to represent "interesting" clauses to solve a certain problem, and are constructed over a sub-vocabulary of the representation language called a *production field*.

Definition 2.2 (1) A *production field* $\mathcal{P}$ is a pair, $\langle L_{\mathcal{P}}, Cond \rangle$, where $L_{\mathcal{P}}$ (called the *characteristic literals*) is a subset of $\mathcal{A}^{\pm}$, and $Cond$ is a condition to be satisfied. When $Cond$ is not specified, $\mathcal{P}$ is just denoted as $\langle L_{\mathcal{P}} \rangle$. A production field $\langle \mathcal{A}^{\pm} \rangle$ is denoted $\mathcal{P}_{\pi}$.
(2) A clause C *belongs to a production field* $\mathcal{P} = \langle L_{\mathcal{P}}, Cond \rangle$ if every literal in C belongs to $L_{\mathcal{P}}$ and C satisfies $Cond$. The set of implicates of Σ belonging to $\mathcal{P}$ is denoted by $Th_{\mathcal{P}}(\Sigma)$.
(3) A production field $\mathcal{P}$ is *stable* if $\mathcal{P}$ satisfies the condition: for two clauses C and C' where C subsumes C', if C' belongs to $\mathcal{P}$, then C also belongs to $\mathcal{P}$.

Example 2.3 The following are examples of implicates belonging to stable production fields.
(1) $\mathcal{P} = \mathcal{P}_{\pi}$: $Th_{\mathcal{P}}(\Sigma)$ is equivalent to $Th(\Sigma)$.
(2) $\mathcal{P} = \langle \mathcal{A} \rangle$: $Th_{\mathcal{P}}(\Sigma)$ is the set of positive clauses implied by Σ.
(3) $\mathcal{P} = \langle \neg \cdot A,$ *below size* $k \rangle$ where $A \subseteq \mathcal{A}$: $Th_{\mathcal{P}}(\Sigma)$ is the set of negative clauses implied by Σ containing less than k literals all of which belong to $\neg \cdot A$.

Definition 2.4 Let Σ be a set of clauses.
(1) The *characteristic clauses of* Σ *with respect to* $\mathcal{P}$ are:

$$Carc(\Sigma, \mathcal{P}) = \mu Th_{\mathcal{P}}(\Sigma).$$

In other words, a characteristic clause of Σ is a prime implicate of Σ belonging to $\mathcal{P}$.
(2) Let F be a formula. The *new characteristic clauses of* F *with respect to* Σ *and* $\mathcal{P}$ are:

$$Newcarc(\Sigma, F, \mathcal{P}) = Carc(\Sigma \cup \{F\}, \mathcal{P}) - Carc(\Sigma, \mathcal{P}),$$

that is, those characteristic clauses of $\Sigma \cup \{F\}$ that are not characteristic clauses of Σ.

$Carc(\Sigma, \mathcal{P})$ represents *saturation*: all the unsubsumed implicates of Σ that belong to a production field $\mathcal{P}$ must be contained in it. For example, $Carc(\Sigma, \mathcal{P}_\pi) = PI(\Sigma)$. Note that the empty clause $\square$ belongs to every stable production field, and that if Σ is unsatisfiable, then $Carc(\Sigma, \mathcal{P})$ contains only $\square$. On the contrary, the next theorem shows that $Newcarc(\Sigma, F, \mathcal{P})$ represents ***abduction***, that is, the set of minimal explanations of $\neg F$ from $(\Sigma, \neg \cdot \mathcal{P})$.

Theorem 2.5 Let Σ be a set of clauses, $A \subseteq \mathcal{A}^{\pm}$, G a formula. The set of all minimal explanations of G from (Σ, A) is $\neg \cdot Newcarc(\Sigma, \neg G, \mathcal{P})$, where $\mathcal{P} = \langle \neg \cdot A \rangle$.

3 Linear Abductive Procedure

In this section, given a set of clauses Σ, a stable production field $\mathcal{P}$ and a formula F, we show how the characteristic clauses $Carc(\Sigma, \mathcal{P})$ and the new characteristic clauses $Newcarc(\Sigma, F, \mathcal{P})$ can be computed by extending linear resolution. Before describing this matter in detail, it is worth noting that, the proof procedure has the following difficulties for dealing with abduction:

1. It should be complete for *consequence-finding*, that is, every relevant theorem can be produced, instead of just *refutation-complete* (producing $\square$ if the theory is unsatisfiable).

2. It should focus on producing only those theorems that belong to $\mathcal{P}$.

3. It should be able to check produced clauses from $\Sigma \cup \{F\}$ and $\mathcal{P}$ with the condition "not belonging to $Th_{\mathcal{P}}(\Sigma)$", which corresponds to consistency checking in abduction.

The completeness for consequence-finding was investigated by Slagle, Chang & Lee [25] and Minicozzi & Reiter [18]. The second property requires that such consequences belong to $\mathcal{P}$. Bossu & Siegel [1] give an *incremental* resolution procedure to overcome the above three difficulties, which should first deduce all the $Carc(\Sigma, \mathcal{P})$ prior to giving $Carc(\Sigma \cup \{F\}, \mathcal{P})$.

A better approach to compute $Newcarc(\Sigma, C, \mathcal{P})$ does not construct the whole of each saturated set. It is possible by using an extension of linear resolution, given Σ, $\mathcal{P}$, and a newly added single clause C as the *top clause* of a deduction. Siegel [24] proposes such a resolution method by extending SL-resolution [15]. In this paper, we use the basic idea of [24] but introduce a more simplified procedure which is enough to explain our goals. The resolution method, which we call *m.c.l.s. resolution*, is based on *m.c.l.* (merge, C-ordered, linear) *resolution* [18] [1], and is augmented by the *skipping* operation. The following procedure is based on the description of OL-deduction in [2], but the result is not restricted to it. An *ordered* clause is a sequence of literals possibly containing *framed literals* which represents literals that have been resolved upon: from a clause C an ordered clause $\vec{C}$ is obtained just by ordering the elements of C; conversely, from an ordered clause $\vec{C}$ a clause C is obtained by removing the framed literals and converting the remainder to the set. A *structured* clause $\langle P, \vec{Q} \rangle$ is a pair of a clause P and an ordered clause $\vec{Q}$, whose clausal meaning is $P \cup Q$.

Definition 3.1 Given a set of clauses Σ, a clause C, and a production field $\mathcal{P} = \langle L_{\mathcal{P}}, Cond \rangle$, an *m.c.l.s. deduction of a clause S from $\Sigma + C$ and $\mathcal{P}$* consists of a sequence of structured clauses $D_0, D_1, \ldots, D_n$, such that:

[1] By the term m.c.l. resolution, we mean the family of linear resolution using ordered clauses and the information of literals resolved upon. Examples of m.c.l. resolution are OL-resolution [2], SL-resolution [15], the model elimination procedure [17], and the graph construction procedure [23]. This family is recognized to be one of the most familiar and efficient classes of resolution because of containing several restriction strategies.

1. $D_0 = \langle \Box, \vec{C} \rangle$.
2. $D_n = \langle S, \Box \rangle$.
3. For each $D_i = \langle P_i, \vec{Q}_i \rangle$, $P_i \cup Q_i$ is not a tautology.
4. For each $D_i = \langle P_i, \vec{Q}_i \rangle$, $P_i \cup Q_i$ is not subsumed by any $P_j \cup Q_j$, where $D_j = \langle P_j, \vec{Q_j} \rangle$ is a previous structured clause, $j < i$.
5. $D_{i+1} = \langle P_{i+1}, \vec{Q_{i+1}} \rangle$ is generated from $D_i = \langle P_i, \vec{Q}_i \rangle$ according to the following steps:
   (a) Let l be the first literal of $\vec{Q}_i$. P_{i+1} and $\vec{R_{i+1}}$ are obtained by applying either of the rules:
      i. **(Skip)** If $l \in L_{\mathcal{P}}$ and $P_i \cup \{l\}$ satisfies *Cond*, then $P_{i+1} = P_i \cup \{l\}$ and $\vec{R_{i+1}}$ is the ordered clause obtained by removing l from $\vec{Q}_i$.
      ii. **(Resolve)** $P_{i+1} = P_i$ and $\vec{R_{i+1}}$ is an ordered resolvent of $\vec{Q}_i$ with a clause B_i in Σ, where the literal resolved upon in $\vec{Q}_i$ is l.
   (b) $\vec{Q_{i+1}}$ is the reduced ordered clause of the ordered factor of $\vec{R_{i+1}}$.

Remarks. (1) Rules 1, 3, 5(a)ii and 5b form an OL-deduction for the non-production part (the right side) of structured clauses. By the *ordered factor* of $\vec{R}_i$, it implies the ordered clause obtained by merging right for any identical literals in $\vec{R}_i$ and by deleting every framed literal not followed by an unframed literal in the remainder (truncation). The *reduction* (or ancestry) of $\vec{R}_i$ deletes any unframed literal k in $\vec{R}_i$ for which there exists a framed literal $\boxed{\neg k}$ in $\vec{R}_i$.

(2) Rule 4 is included for efficiency. It does not affect the completeness described below [2].

(3) Rules 5(a)i and 5(a)ii are not exclusive; for $l \in L_{\mathcal{P}}$ either rule may be applied.

The **Skip** rule (5(a)i) reflects the following operational interpretation of a *stable* production field $\mathcal{P}$: by Definition 2.2 (3), if a clause C does not belong to $\mathcal{P}$ and a clause C' is subsumed by C, then C' does not belong to $\mathcal{P}$ either. That is why we can prune a deduction sequence if no rule can be applied for a structured clause D_i; if **Skip** was applied nevertheless, any resultant sequence would not succeed, thus making unnecessary computation.

For m.c.l.s. resolution, the following theorem can be shown to hold.

Theorem 3.2 (1) *Soundness*: If a clause S is derived using an m.c.l.s. deduction from $\Sigma + C$ and $\mathcal{P}$, then S belongs to $Th_{\mathcal{P}}(\Sigma \cup \{C\})$.
(2) *Completeness*: If a clause T *does not belong to* $Th_{\mathcal{P}}(\Sigma)$, *but* belongs to $Th_{\mathcal{P}}(\Sigma \cup \{C\})$, then there is an m.c.l.s. deduction of a clause S from $\Sigma + C$ and $\mathcal{P}$ such that S subsumes T.

Note that m.c.l. resolution is refutation-complete [17, 15, 2], but is incomplete for consequence-finding [18]. The procedure of m.c.l.s. resolution is complete for characteristic-clause-finding (Theorem 3.2 (2)), and thus complete for consequence-finding if $\mathcal{P} = \mathcal{P}_*$, because it includes the additional skipping operation.

Definition 3.3 Given a set of clauses Σ, a clause C, and a stable production field $\mathcal{P}$, the *production from* $\Sigma + C$ *and* $\mathcal{P}$ is defined as:
$Prod(\Sigma, C, \mathcal{P}) = \mu\{ S \mid S$ is a clause derived using an m.c.l.s. deduction from $\Sigma + C$ and $\mathcal{P}\}$.

[2] In fact, in Chang & Lee's version of OL-deduction [2] this rule is overlooked. The deletion rule is clearly present in the model elimination procedure [17]. These two observations were pointed out by Mark Stickel.

In [24], there is no precise statement about computing $Newcarc(\Sigma, C, \mathcal{P})$ and $Carc(\Sigma, \mathcal{P})$ by using $Prod(\Sigma, C, \mathcal{P})$. Here we show the connections between them. Firstly, the next theorem shows that we can compute $Newcarc(\Sigma, C, \mathcal{P})$ for a single clause C, without a naive implementation of Definition 2.4 (2) that computes the saturated sets, $Carc(\Sigma, \mathcal{P})$ and $Carc(\Sigma \cup \{C\}, \mathcal{P})$, and that we need check for each clause $S \in Prod(\Sigma, C, \mathcal{P})$, only whether $\Sigma \models S$ or not.

Theorem 3.4 Let C be a clause. $Newcarc(\Sigma, C, \mathcal{P}) = Prod(\Sigma, C, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma)$.

For a CNF formula G, $Newcarc(\Sigma, G, \mathcal{P})$ can be computed incrementally as follows:

Theorem 3.5 Let $G = C_1 \wedge \cdots \wedge C_m$ be a CNF formula. Then

$$\begin{aligned} Newcarc(\Sigma, G, \mathcal{P}) &= \mu[\bigcup_{i=1}^{m} Newcarc(\Sigma_i, C_i, \mathcal{P})] \\ &= \mu[\bigcup_{i=1}^{m} Prod(\Sigma_i, C_i, \mathcal{P})] - Th_{\mathcal{P}}(\Sigma), \end{aligned}$$

where $\Sigma_1 = \Sigma$, and $\Sigma_{i+1} = \Sigma_i \cup \{C_i\}$, for $i = 1, \ldots, m-1$.

Finally, the characteristic clauses $Carc(\Sigma, \mathcal{P})$ can be generated by the following incremental method. This will be used for the compiled approaches to the CMS and an ATMS. Notice that for some propositional symbol p, if $\Sigma \not\models p$, $\Sigma \not\models \neg p$, and $p \vee \neg p$ belongs to some stable production field $\mathcal{P}$, then $p \vee \neg p$ belongs to $Carc(\Sigma, \mathcal{P})$.

Theorem 3.6 The characteristic clauses with respect to $\mathcal{P}$ can be generated incrementally [3]:

$$\begin{aligned} Carc(\phi, \mathcal{P}) &= \{\, p \vee \neg p \mid p \in \mathcal{A} \text{ and } p \vee \neg p \text{ belongs to } \mathcal{P} \,\}, \text{ and} \\ Carc(\Sigma \cup \{C\}, \mathcal{P}) &= \mu[Carc(\Sigma, \mathcal{P}) \cup Newcarc(\Sigma, C, \mathcal{P})] \\ &= \mu[Carc(\Sigma, \mathcal{P}) \cup Prod(\Sigma, C, \mathcal{P})]. \end{aligned}$$

4 The CMS

Reiter & de Kleer [22] propose a generalization of the basic ATMS [4] called the *clause management system* (CMS) and show its applications to abductive reasoning. A CMS is intended to work together with a reasoner, which issues queries that take the form of clauses. The CMS is then responsible for finding *minimal supports* for the queries:

Definition 4.1 [22] Let Σ be a set of clauses and C a clause. A clause S is a *support for C with respect to* Σ if $\Sigma \models S \cup C$, and $\Sigma \not\models S$.
A support for C with respect to Σ is *minimal* if there is no other support S' for C which subsumes S. The set of minimal supports for C with respect to Σ is written $MS(\Sigma, C)$.

Comparing minimal supports with minimal explanations described in Definition 1.1, a minimal support S for C with respect to Σ is exactly a minimal explanation $\neg S$ of C from $(\Sigma, \mathcal{A}^{\pm})$. Therefore, the above definition can be easily extended to handle any formula instead of a clause as a query. Setting the production field to $\mathcal{P}_* = (\mathcal{A}^{\pm})$, we see that:

[3] In practice, no tautology will take part in any deduction; tautologies decrease monotonically.

Proposition 4.2 Let F be any formula. $MS(\Sigma, F) = Newcarc(\Sigma, \neg F, \mathcal{P}_\pi)$.

This formulation can solve one of the limitations of the CMS. In [22], the CMS is defined to handle only the queries of the clause form, so that it cannot compute minimal explanations of a conjunctive query. For example, $\mu\{\neg e \mid \Sigma \models e \supset g_1 \wedge g_2 \text{ and } \Sigma \not\models \neg e\}$ can be computed straightforwardly in our formulation as $Newcarc(\Sigma, \neg g_1 \vee \neg g_2, \mathcal{P}_\pi)$. And for a disjunctive normal form query F, we can compute $MS(\Sigma, \neg F)$ by using Theorem 3.5.

We thus see that our algorithm can compute minimal supports. However, Reiter & de Kleer [22] consider the two ways the CMS manages the knowledge base: keeping the set of clauses Σ transmitted by the reasoner as it is (the *interpreted* approach), or computing $PI(\Sigma)$ (the *compiled* approach). Theorem 3.4 shows that we can generate the new characteristic clauses $Newcarc(\Sigma, C, \mathcal{P}_\pi)$ without knowing the saturated sets, $PI(\Sigma)$ and $PI(\Sigma \cup \{C\})$. Therefore, computation using Theorem 3.4 and Proposition 4.2 represents the interpreted approach [4].

When we are faced with a situation where we want to know explanations for many different queries, we must run the algorithm each time a query is issued. Instead of keeping the initial theory Σ as it is and doing the same deductions over and over for different top clauses, some of these inferences can be made once and for all. That is the motivation for the compiled approach: the set Σ is compiled into the saturated set, $PI(\Sigma) = Carc(\Sigma, \mathcal{P}_\pi)$.

Given $PI(\Sigma)$, to find $MS(\Sigma, G)$ for each query G in the compiled approach, again we do not need to compute the saturated set $PI(\Sigma \cup \{\neg G\})$, as Reiter & de Kleer show some relationships between prime implicates and minimal supports.

Proposition 4.3 [22] Let C be a clause. $MS(\Sigma, C) = \mu\{ P - C \mid P \in PI(\Sigma) \text{ and } P \cap C \neq \phi \}$.

Corollary 4.4 [22] Let $n \in \mathcal{A}^{\pm}$ be a literal. $MS(\Sigma, \{n\}) = \{ P - \{n\} \mid P \in PI(\Sigma) \text{ and } n \in P \}$.

One of the disadvantages of the compiled approach is the high cost of updating the knowledge base. When the reasoner adds a clause C to Σ, we must compute all the $PI(\Sigma \cup \{C\})$. However, for both purposes, that is, constructing the prime implicates and updating them, Theorem 3.6 can be used by setting the production field to $\mathcal{P}_\pi$.

Proposition 4.5 Given $PI(\Sigma)$ and a clause C, $PI(\Sigma \cup \{C\})$ can be found incrementally:

$$\begin{aligned} PI(\phi) &= \{ p \vee \neg p \mid p \in \mathcal{A} \}, \text{ and} \\ PI(\Sigma \cup \{C\}) &= \mu[\, PI(\Sigma) \cup Prod(PI(\Sigma), C, \mathcal{P}_\pi) \,]. \end{aligned}$$

By Proposition 4.5, the prime implicates can be incrementally constructed using every clause as a top clause. Thus the transmitted clauses Σ can be substituted for $PI(\Sigma)$. When a clause C is newly added, we just need to add the theorems deduced from $PI(\Sigma)$ with top clause C and to remove the subsumed clauses. The computation of all prime implicates of Σ by Proposition 4.5 is much more efficient than the brute-force way of resolution proposed briefly by Reiter & de Kleer [22], which makes every possible resolution until no more unsubsumed clauses are produced.

Note that either computing supports with an uncompiled theory or compiling a theory is an enumeration problem of prime implicates and each computational complexity is exponential [5]. The computational superiority of the proposed technique as compared with a brute-force algorithm comes from the restriction of resolution, as the key problem here is to generate as few as possible subsumed clauses together with making as few as possible subsumption tests.

[4] Note that in [22] there is no description of an algorithm for the interpreted approach.

[5] In [12], another interesting and empirically efficient way to manage the knowledge base that offers an intermediate alternative to the compiled and interpreted disjunctive is shown.

5 An ATMS

In de Kleer's versions of ATMSs [4, 5, 6, 7], there is a distinguished set of assumptions $A \subseteq \mathcal{A}^{\pm}$. One of the most generalized versions of the ATMS can be considered as a CMS with assumptions as described in Definition 1.1. Therefore, based on Theorem 2.5, an ATMS can be defined as a system responsible for finding all the minimal explanations (called the labels) for the queries:

Definition 5.1 An *ATMS* is a triple $\langle N, A, \Sigma \rangle$, where $N \subseteq \mathcal{A}^{\pm}$ is a set of literals, *nodes*; $A \subseteq N$ is a set of literals, *assumptions*; and Σ is a set of clauses all of whose literals belong to $N \cup \neg \cdot N$, *justifications*. The *label of* $n \in N$ *with respect to* $\langle N, A, \Sigma \rangle$ is defined as:

$$L(n, A, \Sigma) = \neg \cdot Newcarc(\Sigma, \neg n, \mathcal{P}), \quad \text{where } \mathcal{P} = \langle \neg \cdot A \rangle.$$

The following properties [4, 6] hold for the label of each node $n \in N$ with respect to an ATMS $\langle N, A, \Sigma \rangle$ given by Definition 5.1:

Proposition 5.2 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$ a literal, and $\mathcal{P} = \langle \neg \cdot A \rangle$.
(1) *Label consistency*: for each $E_i \in L(n, A, \Sigma)$, $\Sigma \cup \{E_i\}$ is satisfiable.
(2) *Label soundness*: for each $E_i \in L(n, A, \Sigma)$, $\Sigma \cup \{E_i\} \vdash n$.
(3) *Label completeness*: for every conjunct E of assumptions in A, if $\Sigma \cup \{E\} \models n$, then there exists $E_i \in L(n, A, \Sigma)$ such that E_i is a sub-conjunct of E.
(4) *Label minimality*: every $E_i \in L(n, A, \Sigma)$ is not a super-conjunct of any other element.

In the same way as the CMS, we will consider the following two problems, that is, abduction and saturation, concerning the computation of the labels of the nodes with respect to an ATMS:

1. *Generating labels.* Given an ATMS $\langle N, A, \Sigma \rangle$, compute $L(n, A, \Sigma)$ for some node $n \in N$ from the original set Σ. This corresponds to the interpreted approach of the CMS.

2. *Updating labels.* Given an ATMS $\langle N, A, \Sigma \rangle$, the current label $L(n, A, \Sigma)$ of each $n \in N$, and a newly added clause C, compute the new label $L(n, A, \Sigma \cup \{C\})$ of every $n \in N$ with respect to $\langle N, A, \Sigma \cup \{C\} \rangle$. This corresponds to the compiled approach of the CMS.

Generating the label $L(n, A, \Sigma)$ of a node n is straightforward by Theorem 3.4 and Definition 5.1. Moreover, a query is not restricted to being a literal of N in this case: for a general formula, Theorem 3.5 can be applied by converting it to CNF.

Example 5.3 Let an ATMS be $\langle \{a, b, c, x, \neg y\}, \{x, \neg y\}, \{\neg a \vee \neg b \vee c,\ \neg x \vee \neg b \vee a,\ y \vee b \vee c\} \rangle$. Then the following deduction finds c's label $\{x \wedge \neg y\}$:
$\langle \square, \underline{\neg c} \rangle$, $\langle \square, \underline{\neg a} \vee \neg b \vee \boxed{\neg c} \rangle$, $\langle \square, \underline{\neg x} \vee \not{\neg b} \vee \boxed{\neg a} \vee \neg b \vee \boxed{\neg c} \rangle$, $\langle \neg x, \boxed{\not{\neg a}} \vee \underline{\neg b} \vee \boxed{\neg c} \rangle$,
$\langle \neg x, \underline{y} \vee \not{b} \vee \boxed{\neg b} \vee \boxed{\neg c} \rangle$, $\langle \neg x \vee y, \boxed{\not{\neg b}} \vee \boxed{\not{\neg a}} \rangle$.

The question is how effectively consistency can be checked by testing whether a clause S, produced from $\Sigma + \neg n$ and $\mathcal{P} = \langle \neg \cdot A \rangle$, belongs to $Th_{\mathcal{P}}(\Sigma)$ or not. A direct implementation is to use a theorem prover, as we already know that S belongs to $\mathcal{P}$, but theorem proving is also possible in m.c.l.s. resolution: $\Sigma \models S$ iff $Prod(\Sigma, \neg S, \mathcal{P}) = \{\square\}$. In this case, since we are not interested in any produced clause from $\Sigma + \neg S$ other than $\square$, the production field $\mathcal{P}$ can be replaced with $\langle \phi \rangle$ and **Skip** (Rule 5(a)i) will never be applied. Thus, there is an m.c.l. refutation from $\Sigma \cup \{\neg S\}$ iff there is an m.c.l.s. deduction from $\Sigma + \neg S$ and $\langle \phi \rangle$.

However, there is another way for consistency checking that offers an intermediate approach between the interpreted and compiled approaches. Unlike with the CMS, the computation of $Carc(\Sigma, \mathcal{P})$ can be performed better as the search focuses on the restricted vocabulary $\mathcal{P}$ if it is small compared with the whole literals $\mathcal{A}^{\pm}$. Having $Carc(\Sigma, \mathcal{P})$, consistency checking is much easier; $S \in Th_{\mathcal{P}}(\Sigma)$ iff there is a clause $T \in Carc(\Sigma, \mathcal{P})$ such that T subsumes S. The characteristic clauses $Carc(\Sigma, (\neg \cdot A))$ are called unsubsumed *nogoods* in the ATMS terminology. This checking can be embedded into an m.c.l.s. deduction: **Skip** (Rule 5(a)i) of Definition 3.1 can be replaced with the following rule:

5(a)i'. **(Skip & Check)** If $P_i \cup \{l\}$ belongs to $\mathcal{P}$ and is not subsumed by any clause of $Carc(\Sigma, \mathcal{P})$, then the same as **Skip**.

Proposition 5.4 If **Skip & Check** is used as Rule 5(a)i of an m.c.l.s. deduction instead of the original **Skip** rule, then $Prod(\Sigma, C, \mathcal{P}) = Newcarc(\Sigma, C, \mathcal{P})$.

In the compiled approach to an ATMS, the following result corresponding to Corollary 4.4 for the CMS and to a generalization of [22, Theorem 7] holds:

Theorem 5.5 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$ a literal, and $\mathcal{P} = (\neg \cdot A)$.

$$Newcarc(\Sigma, \neg n, \mathcal{P}) = \{ P - \{n\} \mid P \in PI(\Sigma),\ n \in P \text{ and } P - \{n\} \text{ belongs to } \mathcal{P} \}.$$

Theorem 5.5 shows that we can compute the label of a node from the prime implicates of Σ. Therefore an approach may keep $PI(\Sigma)$ and when a new clause C is added we compute $PI(\Sigma \cup \{C\})$ by Proposition 4.5 for updating labels of nodes. However, compared with the CMS, many of the prime implicates are not significant for the task of an ATMS when the assumptions A are relatively small, although their computation is extremely high. In such a case, we do not want to compute all the prime implicates. Fortunately, we can compute a subset of $PI(\Sigma)$ enough to give labels by using the following stable production field:

Definition 5.6 Given an ATMS $\langle N, A, \Sigma \rangle$ and a production field $\mathcal{P} = \langle \neg \cdot A \rangle$, a production field $\mathcal{P}^*$ is defined as:

$$\mathcal{P}^* = \langle \neg \cdot A \cup N,\ \textit{the number of literals in } N - \neg \cdot A \textit{ is at most one} \rangle.$$

Since $\mathcal{P}^*$ is stable, $Carc(\Sigma, \mathcal{P}^*)$ can be constructed incrementally by using Theorem 3.6:

$$Carc(\Sigma \cup \{C\}, \mathcal{P}^*) = \mu[\, Carc(\Sigma, \mathcal{P}^*) \cup Prod(\Sigma, C, \mathcal{P}^*) \,].$$

Here we only need to keep Σ and $Carc(\Sigma, \mathcal{P}^*)$. Looking further at Definition 5.6, the relationship between $Carc(\Sigma, \mathcal{P}^*)$ and $Carc(\Sigma, \mathcal{P})$ can be shown exactly in the next lemma:

Lemma 5.7 $Carc(\Sigma, \mathcal{P}^*) = Carc(\Sigma, \mathcal{P}) \cup \{ S \cup \{n\} \mid n \in N - \neg \cdot A \text{ and } S \in Newcarc(\Sigma, \neg n, \mathcal{P}) \}$.

Therefore, the knowledge base can consist of the justifications Σ, unsubsumed nogoods $Carc(\Sigma, \mathcal{P})$, and prime implicates mentioning one node with the negation of an element of its label. No other prime implicates are necessary. Having $Carc(\Sigma, \mathcal{P}^*)$, we can find the label of each node easily as follows:

Theorem 5.8 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$, $\mathcal{P} = \langle \neg A \rangle$, and $\mathcal{P}^*$ be the same as in Definition 5.6.

$$Newcarc(\Sigma, \neg n, \mathcal{P}) = \begin{cases} \{ S - \{n\} \mid S \in Carc(\Sigma, \mathcal{P}^*), \text{ and } n \in S \} & \text{if } n \in N - \neg \cdot A \\ \{ S - \{n\} \mid S \in Carc(\Sigma, \mathcal{P}), \text{ and } n \in S \} & \text{if } n \in N \cap \neg \cdot A \end{cases}$$

For updating the knowledge base when a new clause C is added, again we just compute $Carc(\Sigma \cup \{C\}, \mathcal{P}^*)$ from the previous $Carc(\Sigma, \mathcal{P}^*)$ incrementally by using Theorem 3.6. Since this computation guarantees the completeness of characteristic-clause-finding, the four properties of the ATMS labels in Proposition 5.2 are also satisfied in this case. Note that the μ operation removes all the previous prime implicates that are subsumed by some newly added prime implicates. This operation is also crucial to guarantee the label consistency because implicates subsumed by some nogood must be removed.

Example 5.9 Suppose that an ATMS is $\langle \{a, b, x, y\}, \{x, y\}, \Sigma \rangle$ where $\Sigma = \{ a \vee b, \neg y \vee a \}$. In this case $Carc(\Sigma, \mathcal{P}^*) = \Sigma \cup \{ \neg x \vee x, \neg y \vee y \}$. Now suppose that a new clause $\neg x \vee \neg a$ is added to Σ. Then the updating algorithm will find b's new label x, as well as a new unsubsumed nogood $\neg x \vee \neg y$:

$$\langle \Box, \underline{\neg x} \vee \neg a \rangle, \quad \langle \neg x, \underline{\neg a} \rangle, \quad \langle \neg x, \underline{b} \vee \boxed{\neg a} \rangle, \quad \langle \neg x \vee b, \boxed{\neg q} \rangle.$$
$$\hookrightarrow \quad \langle \neg x, \underline{\neg y} \vee \boxed{\neg a} \rangle, \quad \langle \neg x \vee \neg y, \boxed{\neg q} \rangle.$$

We thus see that $Carc(\Sigma, \mathcal{P}^*)$ can be used for giving labels for nodes. To maximize efficiency, however, it can also be used for caching the result of the production as lemmas; it can be utilized later as the bypass of steps of resolution in the previous computation. In [12], we describe how the updating algorithm can be modified for this purpose and still establish the label completeness for various ATMSs [4, 5, 6, 8], and the correspondence of the modified algorithm with de Kleer's label updating algorithms [4, 6].

6 Related Works

In this section, we compare our characteristic-clause-finding procedure to proof procedures of various abductive and nonmonotonic reasoning systems. The notions of production fields and (new) characteristic clauses are very helpful in understanding the relationships between them and in reconstructing them in our simple and general formalism.

6.1 Saturation and Abduction

Bossu & Siegel [1] define a closed-world reasoning called sub-implication, in which all ground atoms are to be minimized. Their saturation procedure finds $Carc(\Sigma, \mathcal{P})$ where the characteristic literals $L_{\mathcal{P}}$ are fixed to positive ground literals (see Example 2.3 (2)). However, it does not use C-ordering, and their method to compute $Newcarc(\Sigma, C, \mathcal{P})$ is a naive implementation of Definition 2.4. Those versions of closed-world assumptions can be generalized to allow for variable and fixed predicates as well as minimized predicates, that is, circumscription of ground theories (see Section 6.2).

Kean & Tsiknis [14] extend Tison's [26] consensus method of producing prime implicates to generate them incrementally. In our framework, the corresponding result is illustrated in Proposition 4.5. The difference is that their method is based on a set-of-support strategy, where subsumption checking is performed at each resolution step, while ours uses linear resolution and thus naturally has more restriction strategies.

De Kleer [7] introduces hyperresolution rules to pre compile a set of clauses Σ, all of which are either positive or negative. This technique is also given in [5] in more general form. An interesting approach in [7] is to use a rule limiting the inference only to negative clauses below a size k. The negative clauses of the resulting set closed under these rules and subsumption are the characteristic clauses $Carc(\Sigma, \mathcal{P})$ where $\mathcal{P} = \langle \neg \cdot A, \textit{ below size } k \rangle$ (see Example 2.3 (3)). In our formulation, instead of using hyperresolution, linear resolution can be used to produce such characteristic clauses for any

clause set Σ and any characteristic literals $L_{\mathcal{P}} \subseteq \mathcal{A}^{\pm}$. In practice, this size-restriction is very useful for minimizing the computational effort, because it causes earlier pruning in m.c.l.s. deduction sequences.

There are many systems for logic-based abductive reasoning. However, many systems [19, 9, 21] other than [20] do not require minimality of explanation. Pople [20] proposed the mechanization of abduction via deduction based on SL-resolution [15], with "synthesis" operation which corresponds to our skipping operation. However, his system does not distinguish literals, that is, the production field is fixed to $\mathcal{P}_*$, and "hypothesizes whatever cannot be proven". This criterion is also used by Cox & Pietrzykowski [3]. It can be implemented if **Skip** (Rule 5(a)i) is preceded by **Resolve** (Rule 5(a)ii) and is applied only if **Resolve** cannot be applied in Step 5a of an m.c.l.s. deduction (Definition 3.1).

Finger [9] gives residue procedures for abductive reasoning where assumptions are restricted to only atoms, but his "resolution residue" uses set-of-support resolution.

6.2 Query Answering for Circumscription

Let $\langle N, A_1 \cup \neg \cdot A_2, \Sigma \rangle$ be an ATMS such that $A_1 \subseteq \mathcal{A}$ and $A_2 \subseteq \mathcal{A}$. We wish to know whether or not a formula is satisfied by every preferred model of Σ. This problem is equivalent to *circumscription* of propositional theories: A_2 is to be *minimized* and A_1 is to be maximized (that is, for each $a \in A_1$, $\neg a$ is to be minimized); $A_1 \cap A_2$ represents the *fixed* propositional symbols and $N - (A_1 \cup A_2)$ represents *variables*. In this case, the production field is $\langle \neg \cdot A_1 \cup A_2 \rangle$.

Przymusinski [21] defines MILO-resolution, a variant of OL-resolution [2], which is used in his circumscriptive theorem prover. Inoue & Helft [13] characterize MILO-resolution as m.c.l.s. resolution where the characteristic literals $L_{\mathcal{P}}$ are set to the positive occurrence of minimized predicates and any occurrence of fixed predicates in the circumscription policy.

Proposition 6.1 [21, 10, 13] Suppose that $L_{\mathcal{P}}$ is the same as in the above description and that $\mathcal{P} = \langle L_{\mathcal{P}} \rangle$. Every circumscriptive minimal model satisfies a formula F if and only if there is a conjunct G of clauses of $[Th_{\mathcal{P}}(\Sigma \cup \{\neg F\}) - Th_{\mathcal{P}}(\Sigma)]$ such that $[Th_{\mathcal{P}}(\Sigma \cup \{\neg G\}) - Th_{\mathcal{P}}(\Sigma)] = \phi$.

There is a big difference between MILO-resolution and the ATMS [13]. In Proposition 6.1, to get theorems in $[Th_{\mathcal{P}}(\Sigma \cup \{C\}) - Th_{\mathcal{P}}(\Sigma)]$ for some clause C, MILO-resolution does not actually compute $Newcarc(\Sigma, C, \mathcal{P})$, while the ATMS does. Let us divide the produced clauses from $\Sigma + C$ and $\mathcal{P}$ possibly containing subsumed clauses into two sets, say $S1$ and $S2$, such that $\Sigma \cup S1 \models S2$. Then adding $S2$ to $S1$ does not change the models of the production. Thus only $S1$ needs to be computed model-theoretically [6]. We call a set $S1$ verifying this condition a *precursor* of the production. Note that a clause in a precursor is not always a prime implicate of Σ. MILO-resolution computes such a precursor, because when the first literal belongs to $L_{\mathcal{P}}$ in Step 5a of an m.c.l.s. deduction (Definition 3.1), only **Skip** (Rule 5(a)i) is applied. On the contrary, since the CMS and the ATMS are used for computing *all and only minimal* supports for a query, if the literal resolved upon belongs to $L_{\mathcal{P}}$, they apply either **Skip** or **Resolve**. Thus a precursor-finding algorithm [13] can be written by ordering two rules as:

5(a)i'. **(Skip & Cut)** If $l'_1 \cup \{l\}$ belongs to $\mathcal{P}$, then the same as **Skip** (Rule 5(a)i).
5(a)ii'. **(Resolve')** Otherwise, the same as **Resolve** (Rule 5(a)ii).

Theorem 6.2 If a clause T is derived by an m.c.l.s. deduction from $\Sigma + C$ and $\mathcal{P}$, then there is a deduction with the **Skip & Cut** rule of a clause S from $\Sigma + C$ and $\mathcal{P}$ such that $\Sigma \cup \{S\} \models T$.

[6] Note that a clause in $S1$ is the weakest in the sense that for any clause $A_2 \subset S_2$ there exists a clause $A_1 \in S_1$ such that $\Sigma \cup \{\neg A_2\} \models \neg A_1$ holds (recall that for $A \in S_1 \cup S_2$, $\neg A$ is an explanation of $\neg C$ from $(\Sigma, \neg \cdot \mathcal{P})$ if $\Sigma \not\models A$).

7 Conclusion

We have shown a procedural interpretation of the CMS and the ATMS based on an extension of linear resolution. The **Skip** rule can be safely embedded in linear resolution strategies making characteristic-clause-finding complete, due to the stability of production fields. While we used the description of OL-resolution as the definition of our linear resolution procedure, **Skip** can be applied to other, superior versions of propositional linear resolution, such as Shostak's graph construction procedure [23], and further improvements on these methods can be used to improve efficiency still more. We should also note that the control of inference can be made to the production in various ways as breadth-first or best-first search [2], integration of top-down and bottom-up strategies [9], reordering subgoal trees [24], and others.

Using the methods described in this paper, many AI techniques such as preferential-models approaches to nonmonotonic reasoning and constraint satisfaction problems, as well as direct applications of abduction or the ATMS, may be helped on the way to better implementation.

Acknowledgment

I would like to thank Koichi Furukawa, Nicolas Helft, Ken Satoh, Yoshihiko Ohta, David Poole and Wolfgang Bibel for helpful discussions on this work.

References

[1] Bossu, G. and Siegel, P., "Saturation, Nonmonotonic Reasoning, and the Closed-World Assumption", *Artificial Intelligence* **25** (1985), pp.23–67.

[2] Chang, C. L., and Lee, R. C. T., *Symbolic Logic and Mechanical Theorem Proving* (Academic Press, 1973).

[3] Cox, P. T. and Pietrzykowski, T., "Causes for Events: Their Computation and Applications", *Proc. 8th Conf. on Automated Deduction*, Lecture Notes in Computer Science 230, Springer-Verlag (1986), pp.608–621.

[4] de Kleer, J., "An Assumption based TMS", *Artificial Intelligence* **28** (1986), pp.127 162.

[5] de Kleer, J., "Extending the ATMS", *Artificial Intelligence* **28** (1986), pp.163–196.

[6] de Kleer, J., "A General Labeling Algorithm for Assumption-based Truth Maintenance", *Proc. AAAI-88* (1988), pp.188–192.

[7] de Kleer, J., *Propositional Inference in CSP and ATMS Techniques*, SSL Paper P89-00023, Xerox Palo Alto Research Center, 1989.

[8] Dressler, O., "An Extended Basic ATMS", *Proc. 2nd Int'l Workshop on Non-Monotonic Reasoning*, Lecture Notes in Artificial Intelligence 346, Springer-Verlag (1989), pp.143–163.

[9] Finger, J. J., *Exploiting Constraints in Design Synthesis*, Department of Computer Science, STAN-CS-88-1204, Stanford University, 1987.

[10] Ginsberg, M. L., "A Circumscriptive Theorem Prover", *Artificial Intelligence* **39** (1989), pp.209–230.

[11] Inoue, K., "Generalizing the ATMS: A Model-based Approach (Preliminary Report)", *IPSJ SIG Reports*, SIG AI 63-3, pp.21-28, Information Processing Society of Japan, March 1989.

[12] Inoue, K., *Procedural Interpretation for an Extended ATMS*, ICOT Technical Report TR-547, ICOT, March 1990.

[13] Inoue, K. and Helft, N., "On Theorem Provers for Circumscription", *Proc. CSCSI-90*, Ottawa (May 1990), pp.212-219.

[14] Kean, A. and Tsiknis, G., *An Incremental Method for Generating Prime Implicants/Implicates*, Technical Report 88-16, Department of Computer Science, The University of British Columbia, 1988.

[15] Kowalski, R. A. and Kuhner, D. G., "Linear Resolution with Selection Function", *Artificial Intelligence* **2** (1971), pp.227-260.

[16] Levesque, H. J., "A Knowledge-level Account of Abduction (preliminary version)", *Proc. IJCAI-89* (1989), pp.1061-1067.

[17] Loveland, D., *Automated Theorem Proving: A Logical Basis*, (North-Holland, 1978).

[18] Minicozzi, E. and Reiter, R., "A Note on Linear Resolution Strategies in Consequence-Finding", *Artificial Intelligence* **3** (1972), pp.175-180.

[19] Poole, D., "A Logical Framework for Default Reasoning", *Artificial Intelligence* **36** (1988), pp.27-47.

[20] Pople, H. E., "On the Mechanization of Abductive Logic", *Proc. IJCAI-73* (1973), pp.147-152.

[21] Przymusinski, T. C., "An Algorithm to Compute Circumscription", *Artificial Intelligence* **38** (1989), pp.49-73.

[22] Reiter, R. and de Kleer, J., "Foundations of Assumption-based Truth Maintenance Systems: Preliminary Report", *Proc. AAAI-87* (1987), pp.183-188.

[23] Shostak, R., "Refutation Graphs", *Artificial Intelligence* **7** (1976), pp.51-64.

[24] Siegel, P., *Représentation et Utilisation de la Connaissance en Calcul Propositionnel*, PhD thesis, University of Aix-Marseille II, 1987.

[25] Slagle, J. R., Chang, C. L., and Lee, R. C. T., "Completeness Theorems for Semantic Resolution in Consequence Finding", *Proc. IJCAI-69* (1969), pp.281-285.

[26] Tison, P., "Generalized Consensus Theory and Application to the Minimization of Boolean Functions", *IEEE transactions on electronic computers* **16** (1967), pp.446-456.

A Appendix: Proofs of Theorems

The next proposition is used to prove Theorem 2.5.

Proposition A.1 $Newcarc(\Sigma, F, \mathcal{P}) = \mu[Th_{\mathcal{P}}(\Sigma \cup \{F\}) - Th_{\mathcal{P}}(\Sigma)]$.

Proof: Let $A = Th_{\mathcal{P}}(\Sigma \cup \{F\})$ and $B = Th_{\mathcal{P}}(\Sigma)$. Notice that $B \subseteq A$. We will prove that $\mu[A - B] = \mu A - \mu B$.

Let $c \in \mu[A - B]$. Then obviously $c \in A - B$ and thus $c \in A$. Now assume that $c \notin \mu A$. Then $\exists d \in \mu A$ such that $d \subset c$. By the minimality of $c \in A - B$, $d \in B$. Since $d \subset c$, $c \in B$, contradiction. Therefore $c \in \mu A$. Clearly, by $c \notin B$, $c \notin \mu B$. Hence, $c \in \mu A - \mu B$.

Conversely, assume that $c \in \mu A - \mu B$. Firstly we must prove that $c \in A - B$. Suppose to the contrary that $c \in B$. Since $c \notin \mu B$, $\exists d \in \mu B$ such that $d \subset c$. However, as $B \subseteq A$, $d \in A$, contradicting the minimality of $c \in A$. Therefore, $c \in A - B$. Now assume that c is not minimal in $A - B$. Then, $\exists e \in A - B$ such that $e \subset c$, again contradicting the minimality of $c \in A$. Hence, $c \in \mu[A - B]$. □

Theorem 2.5 Let Σ be a set of clauses, $A \subseteq \mathcal{A}^{\pm}$, G a formula. The set of all minimal explanations of G from (Σ, A) is $\neg \cdot Newcarc(\Sigma, \neg G, \mathcal{P})$, where $\mathcal{P} = \langle \neg \cdot A \rangle$.

Proof: Suppose that H is an explanation of G from (Σ, A). By Definition 1.1, it is observed that (1) $\Sigma \cup \{H\} \models G$ can be written as $\Sigma \cup \{\neg G\} \models \neg H$, (2) the fact that $\Sigma \cup \{H\}$ is satisfiable means $\Sigma \not\models \neg H$, and (3) $\neg H$ is a clause all of whose literals belong to $\neg \cdot A$. Thus $\neg H \in [Th_{\mathcal{P}}(\Sigma \cup \{\neg G\}) - Th_{\mathcal{P}}(\Sigma)]$. Conversely, it can be easily shown that if $F \in [Th_{\mathcal{P}}(\Sigma \cup \{\neg G\}) - Th_{\mathcal{P}}(\Sigma)]$, then $\neg F$ is an explanation of G from (Σ, A). By Proposition A.1, H is a minimal explanation of G from (Σ, A) if and only if $\neg H \in Newcarc(\Sigma, \neg G, \mathcal{P})$. □

Theorem 3.2 (1) *Soundness*: If a clause S is derived using an m.c.l.s. deduction from $\Sigma + C$ and $\mathcal{P}$, then S belongs to $Th_{\mathcal{P}}(\Sigma \cup \{C\})$.
(2) *Completeness*: If a clause T *does not belong to* $Th_{\mathcal{P}}(\Sigma)$, *but* belongs to $Th_{\mathcal{P}}(\Sigma \cup \{C\})$, then there is an m.c.l.s. deduction of a clause S from $\Sigma + C$ and $\mathcal{P}$ such that S subsumes T.

Proof: The proof for the completeness can be seen as an extension of the result for linear resolution by Minicozzi & Reiter [18]. And these results follow easily using the same method as in the proofs for Siegel's procedure described in [24]. □

The next lemma is a direct consequence of Theorem 3.2, and is used for the proof of Theorem 3.4.

Lemma A.2 Let C be a clause. $Newcarc(\Sigma, C, \mathcal{P}) \subseteq Prod(\Sigma, C, \mathcal{P}) \subseteq Th_{\mathcal{P}}(\Sigma \cup \{C\})$.

Proof: Let $T \in Newcarc(\Sigma, C, \mathcal{P})$. By Proposition A.1, $T \in \mu[Th_{\mathcal{P}}(\Sigma \cup \{C\}) - Th_{\mathcal{P}}(\Sigma)]$. By Theorem 3.2 (2), $\exists S \in Prod(\Sigma, C, \mathcal{P})$ such that $S \subseteq T$. By the minimality of T, $T = S$. Hence, $Newcarc(\Sigma, C, \mathcal{P}) \subseteq Prod(\Sigma, C, \mathcal{P})$. By Theorem 3.2 (1), the second set-inclusion relationship easily follows. □

Theorem 3.4 Let C be a clause. $Newcarc(\Sigma, C, \mathcal{P}) = Prod(\Sigma, C, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma)$.

Proof: By Lemma A.2, $Newcarc(\Sigma, C, \mathcal{P}) \subseteq Prod(\Sigma, C, \mathcal{P})$. It remains to show that $Prod(\Sigma, C, \mathcal{P}) - Newcarc(\Sigma, C, \mathcal{P}) \subseteq Th_{\mathcal{P}}(\Sigma)$. Suppose to the contrary, for $S \in Prod(\Sigma, C, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma)$, that $S \notin Newcarc(\Sigma, C, \mathcal{P})$. Since $S \notin Th_{\mathcal{P}}(\Sigma)$, $S \notin Carc(\Sigma, \mathcal{P})$. Because $S \in Prod(\Sigma, C, \mathcal{P})$, $S \in Th_{\mathcal{P}}(\Sigma \cup \{C\})$ by Lemma A.2. Since S is not minimal by the supposition, $\exists S' \in Carc(\Sigma \cup \{C\}, \mathcal{P})$ such that $S' \subset S$. Then, clearly $S' \notin Th_{\mathcal{P}}(\Sigma)$ as $S' \subset S$. Thus, $S' \in Th_{\mathcal{P}}(\Sigma \cup \{C\}) - Th_{\mathcal{P}}(\Sigma)$. By Theorem 3.2 (2), $\exists S'' \in Prod(\Sigma, C, \mathcal{P})$ such that $S'' \subseteq S' \subset S$. However, by Definition 3.3, $Prod(\Sigma, C, \mathcal{P})$ is μ-closed, that is, does not contain any redundant clauses, contradiction. Hence, $S \in Newcarc(\Sigma, C, \mathcal{P})$. □

Theorem 3.5 Let $G = C_1 \wedge \cdots \wedge C_m$ be a CNF formula. Then

$$\begin{aligned} Newcarc(\Sigma, G, \mathcal{P}) &= \mu[\bigcup_{i=1}^{m} Newcarc(\Sigma_i, C_i, \mathcal{P})] \\ &= \mu[\bigcup_{i=1}^{m} Prod(\Sigma_i, C_i, \mathcal{P})] - Th_{\mathcal{P}}(\Sigma), \end{aligned}$$

where $\Sigma_1 = \Sigma$, and $\Sigma_{i+1} = \Sigma_i \cup \{C_i\}$, for $i = 1, \ldots, m-1$.

Proof: Notice that in the following proof, for sets, A, B, and C, such that $C \subseteq B \subseteq A$, $A - C = (A - B) \cup (B - C)$ holds.

$$\begin{aligned} & Newcarc(\Sigma, G, \mathcal{P}) \\ = \; & \mu[\, Th_{\mathcal{P}}(\Sigma \cup \{C_1, \cdots, C_m\}) - Th_{\mathcal{P}}(\Sigma)\,] \text{ (by Proposition A.1)} \\ = \; & \mu[\, (Th_{\mathcal{P}}(\Sigma_m \cup \{C_m\}) - Th_{\mathcal{P}}(\Sigma_m)) \cup \cdots \cup (Th_{\mathcal{P}}(\Sigma \cup \{C_1\}) - Th_{\mathcal{P}}(\Sigma))\,] \\ = \; & \mu[\, \mu[Th_{\mathcal{P}}(\Sigma_m \cup \{C_m\}) - Th_{\mathcal{P}}(\Sigma_m)] \cup \cdots \cup \mu[Th_{\mathcal{P}}(\Sigma_2) - Th_{\mathcal{P}}(\Sigma_1)]\,] \\ = \; & \mu[\, Newcarc(\Sigma_m, C_m, \mathcal{P}) \cup \cdots \cup Newcarc(\Sigma_1, C_1, \mathcal{P})\,] \\ = \; & \mu[\bigcup_{i=1}^{m} Newcarc(\Sigma_i, C_i, \mathcal{P})]. \end{aligned}$$

Now, by applying Theorem 3.4, we get the following equation, which can be used successively to prove the last equality:

$$\begin{aligned} & Newcarc(\Sigma_{k+1}, C_{k+1}, \mathcal{P}) \cup Newcarc(\Sigma_k, C_k, \mathcal{P}) \\ = \; & (\, Prod(\Sigma_k \cup \{C_k\}, C_{k+1}, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma_k \cup \{C_k\})) \cup (\, Prod(\Sigma_k, C_k, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma_k)) \\ = \; & (\, Prod(\Sigma_k \cup \{C_k\}, C_{k+1}, \mathcal{P}) \cup Prod(\Sigma_k, C_k, \mathcal{P})) - (\, Th_{\mathcal{P}}(\Sigma_k \cup \{C_k\}) - Prod(\Sigma_k, C_k, \mathcal{P})) \\ & - (\, Th_{\mathcal{P}}(\Sigma_k) - Prod(\Sigma_k \cup \{C_k\}, C_{k+1}, \mathcal{P})) - (\, Th_{\mathcal{P}}(\Sigma_k \cup \{C_k\}) \cap Th_{\mathcal{P}}(\Sigma_k)) \\ = \; & (\, Prod(\Sigma_{k+1}, C_{k+1}, \mathcal{P}) \cup Prod(\Sigma_k, C_k, \mathcal{P})) - \phi \\ & - (\, Th_{\mathcal{P}}(\Sigma_k) - Prod(\Sigma_k \cup \{C_k\}, C_{k+1}, \mathcal{P})) - Th_{\mathcal{P}}(\Sigma_k) \\ = \; & (\, Prod(\Sigma_{k+1}, C_{k+1}, \mathcal{P}) \cup Prod(\Sigma_k, C_k, \mathcal{P})) - Th_{\mathcal{P}}(\Sigma_k). \end{aligned}$$

Hence, $Newcarc(\Sigma, G, \mathcal{P}) = \mu[\bigcup_{i=1}^{m} Prod(\Sigma_i, C_i, \mathcal{P})] - Th_{\mathcal{P}}(\Sigma)$. □

Theorem 3.6 The characteristic clauses with respect to $\mathcal{P}$ can be generated incrementally:

$$\begin{aligned} Carc(\phi, \mathcal{P}) &= \{\, p \vee \neg p \mid p \in \mathcal{A} \text{ and } p \vee \neg p \text{ belongs to } \mathcal{P} \,\}, \text{ and} \\ Carc(\Sigma \cup \{C\}, \mathcal{P}) &= \mu[\, Carc(\Sigma, \mathcal{P}) \cup Newcarc(\Sigma, C, \mathcal{P})] \\ &= \mu[\, Carc(\Sigma, \mathcal{P}) \cup Prod(\Sigma, C, \mathcal{P})]. \end{aligned}$$

Proof: The first equation follows immediately from Definition 2.4 (1). Now,

$$\begin{aligned}
Carc(\Sigma \cup \{C\}, \mathcal{P}) &= \mu Th_{\mathcal{P}}(\Sigma \cup \{C\}) \\
&= \mu[Th_{\mathcal{P}}(\Sigma \cup \{C\}) \cup Th_{\mathcal{P}}(\Sigma)] \\
&= \mu[\mu Th_{\mathcal{P}}(\Sigma \cup \{C\}) \cup \mu Th_{\mathcal{P}}(\Sigma)] \quad (*) \\
&= \mu[Carc(\Sigma, \mathcal{P}) \cup Carc(\Sigma \cup \{C\}, \mathcal{P})] \\
&= \mu[Carc(\Sigma, \mathcal{P}) \cup (Carc(\Sigma \cup \{C\}, \mathcal{P}) - Carc(\Sigma, \mathcal{P}))] \\
&= \mu[Carc(\Sigma, \mathcal{P}) \cup Newcarc(\Sigma, C, \mathcal{P})] \\
&= \mu[Carc(\Sigma, \mathcal{P}) \cup (Prod(\Sigma, C, \mathcal{P}) - Th_{\mathcal{P}}(\Sigma))] \text{ (by Theorem 3.4)} \\
&= \mu[Carc(\Sigma, \mathcal{P}) \cup Prod(\Sigma, C, \mathcal{P})] .
\end{aligned}$$

Notice that at $(*)$, for two sets, A and B, $\mu[A \cup B] = \mu[\mu A \cup \mu B]$ holds. □

Proposition 4.2 Let F be any formula. $MS(\Sigma, F) = Newcarc(\Sigma, \neg F, \mathcal{P}_\pi)$.

Proof: A clause S is a support for F with respect to Σ
$\Leftrightarrow$ $\Sigma \models S \cup F$, and $\Sigma \not\models S$
$\Leftrightarrow$ $\Sigma \cup \{\neg F\} \models S$, and $\Sigma \not\models S$
$\Leftrightarrow$ $S \in [Th(\Sigma \cup \{\neg F\}) - Th(\Sigma)]$.
Therefore, $S \in MS(\Sigma, F) \Leftrightarrow S \subset Newcarc(\Sigma, \neg F, \mathcal{P}_\pi)$ (by Proposition A.1). □

The next lemma says that the set Σ of clauses is logically equivalent to $PI(\Sigma)$ in the sense that both sets can produce the same (new) characteristic clauses with respect to a production field $\mathcal{P}$.

Lemma A.3 For any stable production field $\mathcal{P}$, $Newcarc(\Sigma, C, \mathcal{P}) = Newcarc(PI(\Sigma), C, \mathcal{P})$.

Proof: Firstly, $Carc(PI(\Sigma), \mathcal{P}) = Carc(Carc(\Sigma, \mathcal{P}_\pi), \langle L_{\mathcal{P}} \rangle) = Carc(\Sigma, \langle \mathcal{L}^{\pm} \cap L_{\mathcal{P}} \rangle) = Carc(\Sigma, \mathcal{P})$.
Now, $Carc(PI(\Sigma) \cup \{C\}, \mathcal{P}) = \mu Th_{\mathcal{P}}(\mu Th_{\mathcal{P}_\pi}(\Sigma) \cup \{C\}) = \mu Th_{\mathcal{P}}(Th(\Sigma) \cup \{C\}) = Carc(\Sigma \cup \{C\}, \mathcal{P})$.
The lemma follows immediately by Definition 2.4 (1). □

Proposition 4.5 Given $PI(\Sigma)$ and a clause C, $PI(\Sigma \cup \{C\})$ can be found incrementally:

$$\begin{aligned}
PI(\phi) &= \{\, p \vee \neg p \mid p \in \mathcal{A} \,\}, \text{ and} \\
PI(\Sigma \cup \{C\}) &= \mu[PI(\Sigma) \cup Prod(PI(\Sigma), C, \mathcal{P}_\pi)] .
\end{aligned}$$

Proof: The proposition follows by setting $\mathcal{P}$ to $\mathcal{P}_\pi$ in Theorem 3.6 and by using Theorem 3.4 and Lemma A.3. □

Proposition 5.2 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$ a literal, and $\mathcal{P} = \langle \neg \cdot A \rangle$.
(1) *Label consistency*: for each $E_i \in L(n, A, \Sigma)$, $\Sigma \cup \{E_i\}$ is satisfiable.
(2) *Label soundness*: for each $E_i \in L(n, A, \Sigma)$, $\Sigma \cup \{E_i\} \models n$.
(3) *Label completeness*: for every conjunct E of assumptions in A, if $\Sigma \cup \{E\} \models n$, then there exists $E_i \in L(n, A, \Sigma)$ such that E_i is a sub-conjunct of E.
(4) *Label minimality*: every $E_i \in L(n, A, \Sigma)$ is not a super-conjunct of any other element.

Proof: By Definition 5.1 and Theorem 2.5, $E_i \in L(n, A, \Sigma)$ is a minimal explanation of n from (Σ, A). Therefore, these four properties obviously hold by Definition 1.1. □

Proposition 5.4 If **Skip & Check** is used as Rule 5(a)i of an m.c.l.s. deduction instead of the original **Skip** rule, then $Prod(\Sigma, C, \mathcal{P}) = Newcarc(\Sigma, C, \mathcal{P})$.

Proof: Because every clause belonging to $Th_{\mathcal{P}}(\Sigma \cup \{C\})$ that is subsumed by some clause in $Carc(\Sigma, \mathcal{P})$ must be pruned in a deduction sequence, every clause produced from $\Sigma + C$ is not a super-clause of any clause in $Carc(\Sigma, \mathcal{P})$ and thus does not belong to $Th_{\mathcal{P}}(\Sigma)$. Hence, by Theorem 3.4, the proposition follows. □

Theorem 5.5 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$ a literal, and $\mathcal{P} = \langle \neg \cdot A \rangle$.

$$Newcarc(\Sigma, \neg n, \mathcal{P}) = \{ P - \{n\} \mid P \in PI(\Sigma),\ n \in P \text{ and } P - \{n\} \text{ belongs to } \mathcal{P} \}.$$

Proof: ($\supseteq$) Let $P \in PI(\Sigma)$ such that $n \in P$ and $P - \{n\}$ belongs to $\mathcal{P}$. Then, since $P - \{n\} \subset P$, $\Sigma \not\models P - \{n\}$. Since $n \in P$ and $\Sigma \models P$, $\Sigma \cup \{\neg n\} \models P - \{n\}$. Therefore, $P - \{n\} \in Th_{\mathcal{P}}(\Sigma \cup \{\neg n\}) - Th_{\mathcal{P}}(\Sigma)$. As $P \in PI(\Sigma)$ and $n \in P$, for any clause $S \subset P - \{n\}$, $\Sigma \not\models S \cup \{n\}$, and thus $\Sigma \cup \{\neg n\} \not\models S$ holds. This implies that $P - \{n\} \in Carc(\Sigma \cup \{\neg n\}, \mathcal{P})$, and thus $P - \{n\} \in Newcarc(\Sigma, \neg n, \mathcal{P})$ (by Proposition A.1)[7].

($\subseteq$) Let $S \in Newcarc(\Sigma, \neg n, \mathcal{P})$. As $\Sigma \cup \{\neg n\} \models S$, $\Sigma \models S \cup \{n\}$ holds. Suppose that $\exists T \in Th(\Sigma)$ such that $T \subset S \cup \{n\}$ and that $T - \{n\}$ belongs to $\mathcal{P}$. Clearly, $\Sigma \models T \cup \{n\}$. Now for any clause S' such that $S' \subset S$, since $\Sigma \cup \{\neg n\} \not\models S'$, $\Sigma \not\models S' \cup \{n\}$ holds. Therefore, $S \subseteq T \subset S \cup \{n\}$. As n is a literal, $T = S$. However, $S \notin Carc(\Sigma, \mathcal{P})$, contradiction. Hence, $S \cup \{n\} \in PI(\Sigma)$. Replacing $S \cup \{n\}$ with P, we get the theorem. □

Lemma 5.7 $Carc(\Sigma, \mathcal{P}^*) = Carc(\Sigma, \mathcal{P}) \cup \{ S \cup \{n\} \mid n \in N - \neg \cdot A \text{ and } S \in Newcarc(\Sigma, \neg n, \mathcal{P}) \}$.

Proof: $Carc(\Sigma, \mathcal{P}^*)$ can be divided into two disjoint sets of clauses: (1) containing no literal in $N - \neg A$, and (2) containing exactly one literal in $N - \neg \cdot A$. The former is exactly $Carc(\Sigma, \mathcal{P})$. Assume that C belongs to the latter set, and that $n \in C$ is a literal in $N - \neg A$. Then, $C - \{n\}$ contains only literals in $\neg \cdot A$ and thus belongs to $\mathcal{P}$. We must show that $C - \{n\} \in Carc(\Sigma \cup \{\neg n\}, \mathcal{P})$. Since $C \in Carc(\Sigma, \mathcal{P}^*)$, $\Sigma \models C$ and thus $\Sigma \cup \{\neg n\} \models C - \{n\}$. Suppose to the contrary that $\exists S \in Th_{\mathcal{P}}(\Sigma \cup \{\neg n\})$ such that $S \subset C - \{n\}$. Then, $S \cup \{n\} \subset C$ and $S \cup \{n\} \notin Th_{\mathcal{P}^*}(\Sigma)$ by the minimality of $C \in Th_{\mathcal{P}^*}(\Sigma)$. Since $\Sigma \not\models S \cup \{n\}$, $\Sigma \cup \{\neg n\} \not\models S$, contradiction. Therefore, $C - \{n\} \in Carc(\Sigma \cup \{\neg n\}, \mathcal{P})$. Since $C \in Carc(\Sigma, \mathcal{P}^*)$, obviously $C - \{n\} \notin Carc(\Sigma, \mathcal{P})$ holds. Hence, the lemma. □

Theorem 5.8 Let $\langle N, A, \Sigma \rangle$ be an ATMS, $n \in N$, $\mathcal{P} = \langle \neg A \rangle$, and $\mathcal{P}^*$ be the same as in Definition 5.6.

$$Newcarc(\Sigma, \neg n, \mathcal{P}) = \begin{cases} \{ S - \{n\} \mid S \in Carc(\Sigma, \mathcal{P}^*), \text{ and } n \in S \} & \text{if } n \in N - \neg \cdot A \\ \{ S - \{n\} \mid S \in Carc(\Sigma, \mathcal{P}), \text{ and } n \in S \} & \text{if } n \in N \cap \neg \cdot A \end{cases}$$

Proof: ($\supseteq$) Obvious from Theorem 5.5 and Lemma 5.7.

($\subseteq$) Let $T \in Newcarc(\Sigma, \neg n, \mathcal{P})$. By Theorem 5.5, $T \cup \{n\} \in PI(\Sigma)$. (1) If $n \in N - \neg \cdot A$, then $T \cup \{n\}$ belongs to $\mathcal{P}^*$ because T belongs to $\mathcal{P}$. Therefore, $T \cup \{n\} \in Carc(\Sigma, \mathcal{P}^*)$. (2) If $n \in N \cap \neg \cdot A$, then $T \cup \{n\}$ belongs to $\mathcal{P}$. Therefore, $T \cup \{n\} \in Carc(\Sigma, \mathcal{P})$. □

Theorem 6.2 is in essence the same as [13, Theorem 4.2].

[7] Note that in this direction n need not be a literal in N; the relation holds for a clause C: if $P \in PI(\Sigma)$, $C \subseteq P$, and $P - C$ belongs to $\mathcal{P}$, then $\neg(P - C)$ is a minimal explanation of C from (Σ, A). This result corresponds to a generalization of [22, Theorem 3] for a general $\mathcal{P}$.

Theorem 6.2 If a clause T is derived by an m.c.l.s. deduction from $\Sigma + C$ and $\mathcal{P}$, then there is a deduction with the **Skip & Cut** rule of a clause S from $\Sigma + C$ and $\mathcal{P}$ such that $\Sigma \cup \{S\} \models T$.

Proof: Let $D_0, D_1, \cdots, D_n$ be an m.c.l.s. deduction of T from $\Sigma + C$ and $\mathcal{P}$. Let l_i be the first literal of $\vec{Q_i}$, where $D_i = \langle P_i, \vec{Q_i} \rangle$ and $0 \leq i \leq n-1$. Firstly, if **Skip** is applied for every l_j $(0 \leq j \leq n-1)$ such that $l_j \in L_{\mathcal{P}}$, then T is actually derived from $\Sigma + C$ and $\mathcal{P}$ by using the **Skip & Cut** rule, and of course $\Sigma \cup \{T\} \models T$ holds.

Next, suppose that $\exists D_j$ in the m.c.l.s. deduction such that $l_j \in L_{\mathcal{P}}$ but that **Resolve** is applied upon l_j in $\vec{Q_j}$ with a clause $B_j \in \Sigma$. Let m $(1 \leq m \leq n)$ be the number of such clauses, and D_k be such a clause where k $(0 \leq k \leq n-1)$ is the largest number. In this case, $D_{k+1} = \langle P_{k+1}, \vec{Q_{k+1}} \rangle$, where $P_{k+1} = P_k$ and $R_{k+1} = (B_k - \{\neg l_k\}) \cup (Q_k - \{l_k\})$. In the following proof, to simplify the discussion, we assume that there are no identical, truncated, or reduced literals in R_{k+1}; if they exist, then we can modify the proof appropriately. Now, let U be a clause m.c.l.s. derived from $\Sigma + (B_k - \{\neg l_k\})$ and $\mathcal{P}$, V a clause m.c.l.s. derived from $\Sigma + (Q_k - \{l_k\})$ and $\mathcal{P}$. Here, we can choose such U and V to satisfy $T = P_k \cup U \cup V$, because T is m.c.l.s. derived from $\Sigma + (P_{k+1} \cup R_{k+1})$ and $\mathcal{P}$.

Now assume that instead of applying **Resolve**, **Skip & Cut** is applied to D_k, deducing $D'_{k+1} = \langle P'_{k+1}, \vec{Q'_{k+1}} \rangle$, where $P'_{k+1} = P_k \cup \{l_k\}$ and $R'_{k+1} = Q_k - \{l_k\}$. Then, $P_k \cup \{l_k\} \cup V$ is m.c.l.s. derived from $\Sigma + (P'_{k+1} \cup R'_{k+1})$ and $\mathcal{P}$, and thus from $\Sigma + C$ and $\mathcal{P}$. Since $\Sigma \cup \{l_k\} \models B_k - \{\neg l_k\}$, $\Sigma \cup \{l_k\} \models U$ holds, and thus $\Sigma \cup \{(P_k \cup \{l_k\} \cup V)\} \models T$ holds.

Now let $T_0 = T$ and $T_1 = (P_k \cup \{l_k\} \cup V)$. In the similar way, we can find an m.c.l.s. deduction of T_2 from $\Sigma + C$ and $\mathcal{P}$ such that $\Sigma \cup \{T_2\} \models T_1$, by resetting k to the second largest number. By using the bottom-up manner, we can successively find clauses T_j $(1 \leq j \leq m)$ m.c.l.s. derived from $\Sigma + C$ and $\mathcal{P}$ such that $\Sigma \cup \{T_j\} \models T_{j-1}$. Therefore, $\Sigma \cup \{T_m\} \models T_{m-1}$, $\Sigma \cup \{T_{m-1}\} \models T_{m-2}$, $\cdots$, $\Sigma \cup \{T_1\} \models T_0$. Hence, $\Sigma \cup \{T_m\} \models T_0$, and we get the theorem. □