RESOLUTION WITH MERGING

Peter B. Andrews[1]

Report 67-33

October, 1967

1 This research was supported by NSF grants GP-4494 and GP-7660.

Resolution with Merging
by Peter B. Andrews[1]

§1. Introduction.

In [2] it was shown that one can associate with any wff of first order predicate calculus a certain set S of clauses such that the given wff is unsatisfiable if and only if there is a refutation of S, i.e. a deduction of the empty clause $\square$ from S. (A sequence $\sigma_1,\ldots,\sigma_n$ of clauses such that each clause in the sequence is either a member of S or a resolvent of earlier clauses in the sequence is called a deduction of σ_n from S (by resolution).) We shall be concerned with a method of increasing the efficiency of the search for a refutation of a given set of clauses.

For convenience we shall broaden slightly the definition of resolvent as given in [2] so as to permit one to combine a substitution with the operation of forming a resolvent as in [2].

Definition. The clause γ is a resolvent of the clauses α and β if there are substitutions $\hat{A}$ and $\hat{B}$, clauses δ and ϵ, and an atom p such that[2] $\hat{A}\alpha = \{p\} \dot{\cup} \delta$, $\hat{B}\beta = \{\sim p\} \dot{\cup} \epsilon$, and $\gamma = \delta \cup \epsilon$. We call p (or its antecedent(s) in α) and $\sim p$ (or its antecedent(s) in β) the literals resolved upon. In addition, if δ and ϵ contain a common literal, we say that γ is a merge of α and β; any literal of γ which occurs in both δ and ϵ is called a merge literal.

Clearly, a merge is an especially important sort of resolvent, for in order to derive $\square$ from a given set of clauses one must obtain successively shorter clauses, and merges provide one of the

[1] This research was supported by NSF grants GP-4494 and GP-7660.

[2] We shall write $\tau = \rho \dot{\cup} \sigma$ as an abbreviation for 'τ is the union of the disjoint sets ρ and σ.' Thus $\dot{\cup}$, unlike $\cup$, is not to be regarded as an operator which can be applied to an arbitrary pair of sets. Nevertheless we find this notation very convenient.

principal means of progressing toward shorter clauses.

It is natural to ask whether one can require that all resolvents in a refutation actually be merges or resolvents involving a one-literal clause. Unfortunately, the answer is negative, as the following example shows. Let S be the set of ground clauses with the following members:

(1) $\{p,q\}$	(2) $\{p,r\}$	(3) $\{q,r\}$
(4) $\{\sim p,\sim q\}$	(5) $\{\sim p,\sim r\}$	(6) $\{\sim q,\sim r\}$.

Resolving (1) with (5), and the result with (3), we obtain $\{q\}$. Similarly we can derive $\{\sim q\}$, and hence $\square$. Hence S has a refutation. However, no resolvent of a pair of clauses in S is a merge.

Nevertheless, we shall discriminate against clauses which are resolvents but not merges by requiring that no two such clauses may be resolved with one another. A deduction satisfying this condition will be called a deduction by resolution with merging. We shall show that if S is any set of clauses which has a refutation, then S has a refutation by resolution with merging. When one seeks to construct a refutation by resolution with merging, one has fewer choices of possible resolvents than in an ordinary refutation, so the 'search tree' grows more slowly. Moreover, one requires that merges occur frequently, and this tends to make the refutation more efficient than it might otherwise be.

Of course, strategies which tend to increase the efficiency of searches for refutations are of most value when they can be combined with other such strategies. Therefore we shall show that resolution with merging can be combined with the set of support strategy [5].

Definition. Given sets K and S of clauses with $K \subseteq S$, we define $\hat{s}(K,S)$, the set of clauses derived from S with K-support, as the smallest set of clauses satisfying the following conditions:

(i) $K \subseteq \hat{s}(K,S)$

(ii) if $\alpha \in \hat{s}(K,S)$ and $\beta \in S \cup \hat{s}(K,S)$ and γ is a resolvent of α and β, then $\gamma \in \hat{s}(K,S)$.

Informally, a deduction from S has K-support if every clause in the deduction is in $S \cup \hat{s}(K,S)$.

We shall show that if K and S are sets of clauses such that S has a refutation with K-support, then S has a refutation by resolution with merging with K-support. To avoid any possible confusion, we shall now define a deduction from S by resolution with merging with K-support as a sequence of triples $<\gamma,\hat{p},\hat{r}>$, where γ is a clause, $\hat{p}$ is $\hat{m}$ or $\hat{o}$ (to indicate whether or not γ is either a merge or a member of S) and $\hat{r}$ is $\hat{s}$ or $\hat{o}$ (to indicate whether or not γ has been shown to have K-support).

Definition. Let K and S be sets of clauses with $K \subseteq S$. A deduction from S by resolution with merging with K-support is a finite sequence of triples $<\gamma,\hat{p},\hat{r}>$, where γ is a clause, $\hat{p}$ is $\hat{m}$ or $\hat{o}$, and $\hat{r}$ is $\hat{s}$ or $\hat{o}$, such that each triple satisfies at least one of the following conditions:

(i) $\gamma \in S$, $\hat{p}$ is $\hat{m}$, and $\hat{r}$ is $\hat{s}$ if and only if $\gamma \in K$.

(ii) $<\gamma,\hat{p},\hat{r}>$ is preceded in the sequence by triples $<\alpha,\hat{p}_1,\hat{r}_1>$ and $<\beta,\hat{p}_2,\hat{r}_2>$ such that γ is a resolvent of α and β and either $\hat{p}_1$ or $\hat{p}_2$ is $\hat{m}$ and either $\hat{r}_1$ or $\hat{r}_2$ is $\hat{s}$; moreover $\hat{r}$ is $\hat{s}$, and $\hat{p}$ is $\hat{m}$ if and only if γ is a merge of α and β.

Of course, if one wishes to use resolution with merging without the set of support strategy, one can trivialize the set of support by taking $K = S$ in the definitions above.

§2. *Resolution Trees*.

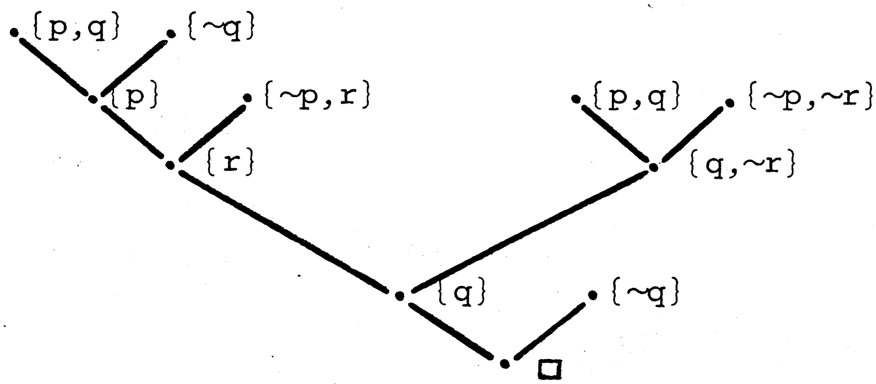
Since we shall be concerned with transforming deductions into deductions by resolution with merging, it will be convenient to arrange our deductions in the form of binary trees, as in Figure 1. This of course requires that a clause occur as many times in the tree as it is used. Thus the nodes of our tree should be regarded as occurrences of clauses, rather than clauses. It is convenient to write $\Phi < \Psi$ if the node Φ is (strictly) below Ψ on some branch of the tree, and the relation $<$ is irreflexive and transitive, so we may regard the tree as a partially ordered set. We take this as the starting point for our formal definitions.

Definitions. Let $\mathfrak{T}$ be a partially ordered set (whose members we call *nodes*) under the irreflexive and transitive relation $<$. If $\Phi < \Psi$ we say that Φ is *below* Ψ, and Ψ is *above* Φ. If $\Phi < \Psi$ and there is no node Ω such that $\Phi < \Omega < \Psi$, we say that Ψ is *immediately* above Φ. Φ is *maximal* [minimal] if there is no node Ω such that $\Phi < \Omega$ $[\Omega < \Phi]$. $\mathfrak{T}$ is a *finite binary tree* if $\mathfrak{T}$ is a finite set, $\mathfrak{T}$ has a unique minimal node, every non-maximal node of $\mathfrak{T}$ has exactly two nodes immediately above it, and every non-minimal node has exactly one node immediately below it. The maximal nodes will be called *leaves*, and the minimal node will be called the *root* of $\mathfrak{T}$, and denoted *root*$(\mathfrak{T})$.

Definition. A *vine* is a finite binary tree in which each node is either a leaf or is immediately below some leaf. A node Φ of a vine is a *top-leaf* if Φ is above every node of the vine which is not a leaf.

It should be noted that a finite binary tree always has an odd number of nodes. Also, a vine which has more than one node has exactly two top-leaves.

Figure 1.



<u>Definition</u>. A <u>resolution tree</u> is a finite binary tree together with a mapping $\hat{c}$ which associates with each node Φ of the tree a clause $\hat{c}\Phi$ (called <u>the clause of</u> Φ) in such a way that if Φ, Ψ, and Ω are any distinct nodes of the tree with Φ and Ψ immediately above Ω, then $\hat{c}\Omega$ is a resolvent of $\hat{c}\Phi$ and $\hat{c}\Psi$.

A node of a resolution tree will be called a <u>merge</u> if it is not a leaf and its clause is a merge of the clauses of the two nodes immediately above it. We shall let $\hat{m}(\mathfrak{T})$ denote the number of merges in a resolution tree $\mathfrak{T}$.

We remark that if Φ is any node in a resolution tree $\mathfrak{T}$, then the set of nodes Ψ such that $\Phi \leq \Psi$ (i.e. $\Phi < \Psi$ or $\Phi = \Psi$) may be regarded in a natural way as a resolution tree; we call this the <u>sub-tree of</u> $\mathfrak{T}$ <u>rooted in</u> Φ.

<u>Definition</u>. Given a set K of clauses, a node of a resolution tree <u>has</u> K-<u>support</u> if it is $\leq$ to a node whose clause is in K. A resolution tree <u>has</u> K-<u>support</u> if each of its nodes which is not a leaf has K-support.

Clearly each deduction from a set S of clauses can be represented by a resolution tree whose leaves have clauses in S, and the deduction has K-support if and only if the tree has K-support. Conversely, each resolution tree (with K-support) represents a deduction (with K-support).

It is easy to see that a vine represents a deduction by resolution with merging. Also, any resolution tree which represents a deduction by resolution with merging has the property that the sub-tree rooted in any maximal merge must be a vine. Moreover, if one prunes away all nodes above a maximal merge in such a tree (so that what was a maximal merge becomes a leaf of the pruned tree), the pruned tree has the same property. Thus a resolution tree representing a deduction by resolution with merging is in a certain sense constructed from vines, with merges (as well as true leaves) serving as leaves of the internal vines.

A mapping f from the nodes of a tree $\mathfrak{T}$ to the nodes of a tree $\mathfrak{T}'$ will be called an <u>order-isomorphism of</u> $\mathfrak{T}$ <u>onto</u> $\mathfrak{T}'$ if f is one-one with domain $\mathfrak{T}$ and range $\mathfrak{T}'$, and for all nodes Φ and Ψ of $\mathfrak{T}$, $f\Phi < f\Psi$ in $\mathfrak{T}'$ iff $\Phi < \Psi$ in $\mathfrak{T}$. A mapping f from a resolution tree $\mathfrak{T}$ to a resolution tree $\mathfrak{T}'$ is <u>clause-preserving</u> if $\hat{c}f\Phi = \hat{c}\Phi$ for each node Φ of $\mathfrak{T}$. If $\mathfrak{T}$ and $\mathfrak{T}'$ are resolution trees such that there is a clause-preserving order isomorphism from $\mathfrak{T}$ onto $\mathfrak{T}'$, we may write $\mathfrak{T} = \mathfrak{T}'$.

We shall establish certain facts about resolution trees. Our task is simplified by the circumstance that for the most part it suffices to consider <u>ground</u> <u>resolution</u> <u>trees</u>, i.e. resolution trees whose clauses are all ground clauses (i.e., contain no individual variables). Of course, no substitutions are involved in forming a resolvent of ground clauses. Thus theorems about ground resolution trees can be regarded as theorems about propositional calculus. We remark that for our present purposes, there is complete symmetry in the roles played by atoms and negated atoms.

Clearly a literal may occur many times in a ground resolution tree, i.e. in (clauses of) different nodes of the tree. It is appropriate to regard certain occurrences of literals as <u>descended</u> from other occurrences of that same literal.

<u>Definition</u>. Given a ground resolution tree, the <u>descendance</u> relation is the smallest binary relation between occurrences of literals in the tree satisfying the following conditions(where a,b, and c are occurrences of literals):

(1) a is a descendant of a.

(2) If a is a descendant of b, and b is a descendant of c, then a is a descendant of c.

(3) If Φ, Ψ, and Ω are distinct nodes of the tree with Φ and Ψ immediately above Ω, with $\hat{c}\Phi = \{p\} \dot{\cup} \delta$, $\hat{c}\Psi = \{\sim p\} \dot{\cup} \epsilon$, and $\hat{c}\Omega = \delta \cup \epsilon$, and if t is any literal in δ, then the occurrence of t in Ω is a descendant of the occurrence of t in Φ; if t is any literal in ϵ, then the occurrence of t in Ω is a descendant of the occurence of t in Ψ.

Definition. a is an ancestor of b iff b is a descendant of a.

§3. Theorems.

Definition. Let K and S be sets of clauses with $K \subseteq S$. $\hat{m}\hat{s}(K,S)$ is the set of all clauses γ such that there is a deduction from S by resolution with merging with K-support, the final triple of which has the form $<\gamma,\hat{m},\hat{s}>$.

Lemma 1. Let K and S be sets of clauses such that $K \subseteq S$. Then $\hat{m}\hat{s}(\hat{m}\hat{s}(K,S),\ S \cup \hat{m}\hat{s}(K,S)) = \hat{m}\hat{s}(K,S)$.

Proof: It is easy to see that if $K_1 \subseteq K \subseteq S \subseteq S_2$ then $\hat{m}\hat{s}(K_1,S) \subseteq \hat{m}\hat{s}(K,S) \subseteq \hat{m}\hat{s}(K,S_2)$. Also $K \subseteq \hat{m}\hat{s}(K,S)$ so $\hat{m}\hat{s}(K,S) \subseteq \hat{m}\hat{s}(K,S \cup \hat{m}\hat{s}(K,S)) \subseteq \hat{m}\hat{s}(\hat{m}\hat{s}(K,S),\ S \cup \hat{m}\hat{s}(K,S))$.

To prove containment in the opposite direction, let $K_o = \hat{m}\hat{s}(K,S)$ and $S_o = S \cup \hat{m}\hat{s}(K,S)$ and suppose $\alpha \in \hat{m}\hat{s}(K_o,S_o)$. Let a particular deduction of α satisfying the conditions of the definition of $\hat{m}\hat{s}(K_o,S_o)$ be given. Let $\beta_1,\ldots,\beta_n$ be the clauses of this deduction which are not resolvents of earlier clauses in the deduction. Then the triple $<\beta_i,\hat{p},\hat{r}>$ in this deduction has $\hat{p}=\hat{m}$, and $\hat{r}$ is $\hat{s}$ if and only if $\beta_i \in K_o$. Since $\beta_i \in S_o = S \cup \hat{m}\hat{s}(K,S)$, there is a deduction of β_i from S by resolution with merging with K-support, whose final triple $<\beta_i,\hat{m},\hat{r}>$ has $\hat{r} = \hat{s}$ if $\beta_i \in \hat{m}\hat{s}(K,S)$, but whose final (and only) triple has $\hat{r} = \hat{0}$ if $\beta_i \notin \hat{m}\hat{s}(K,S)$. It is now easy to see that the deductions of $\beta_i,\ldots,\beta_n$, and α mentioned above can be fitted together in the obvious way to obtain a deduction of α from S by resolution with merging with K-support, whose final triple is $<\alpha,\hat{m},\hat{s}>$. Thus $\alpha \in \hat{m}\hat{s}(K,S)$.

The reader will recall that under the usual interpretation of a clause as the disjunction of its literals, if $\alpha \subseteq \beta$, then α implies β. Our next lemma states, roughly , that if we have a deduction of δ from clauses $\beta_1,\ldots,\beta_n$, and if α_i implies β_i for $1 \leq i \leq n$, then from $\alpha_1,\ldots,\alpha_n$ there is a deduction of a clause γ which implies δ. Moreover this new deduction is in certain respects at least as simple as the given one.

Lemma 2. Let $\mathfrak{J}$ be a ground resolution tree and let d be a mapping which associates with every leaf Λ of $\mathfrak{J}$ a clause $d\Lambda$ such that $d\Lambda \subseteq \hat{c}\Lambda$. Let $\mathcal{K}$ be a subset of the leaves of $\mathfrak{J}$ such that $\{\Lambda | d\Lambda \neq \hat{c}\Lambda\} \subseteq \mathcal{K}$ and every non-leaf of $\mathfrak{J}$ is below some leaf in $\mathcal{K}$. Then there is a ground resolution tree $\mathfrak{J}'$ and a one-one map f from the set of leaves of $\mathfrak{J}'$ into the set of leaves of $\mathfrak{J}$ such that:

(1) $\hat{c}\Lambda = df\Lambda$ for each leaf Λ of $\mathfrak{J}'$;
(2) $\hat{c}(\text{root } \mathfrak{J}') \subseteq \hat{c}(\text{root } \mathfrak{J})$;
(3) $\hat{m}(\mathfrak{J}') \leq \hat{m}(\mathfrak{J})$;
(4) each non-leaf of $\mathfrak{J}'$ is below some leaf in $f^{-1}(\mathcal{K} \cap \text{range } f)$;
(5) either $\mathfrak{J}' = \mathfrak{J}$ or root $(\mathfrak{J}')$ is $\leq$ some leaf in $f^{-1}(\mathcal{K} \cap \text{range } f)$.

Remark: Clearly (5) follows from (4) unless $\mathfrak{J}'$ consists of a single node, so that its root is a leaf.

Proof: by induction on the number of nodes in $\mathfrak{J}$.

If $\mathfrak{J}$ has just one node Ψ, let $\mathfrak{J}'$ be the tree with a single node whose clause is $d\Psi$. This makes $\mathfrak{J}'$ the same as $\mathfrak{J}$ unless $d\Psi \neq \hat{c}\Psi$; in the latter case $\Psi \in \mathcal{K}$, so condition (5) is satisfied. Verification of conditions (1)-(4) is trivial.

Suppose $\mathfrak{J}$ has three or more nodes. Let Φ_1 and Φ_2 be the nodes immediately above the root of $\mathfrak{J}$, and let $\mathfrak{J}_1$ and $\mathfrak{J}_2$ be the sub-trees of $\mathfrak{J}$ rooted in Φ_1 and Φ_2, respectively. We may assume $\hat{c}\Phi_1 = \{p\} \dot{\cup} \alpha_1$, $\hat{c}\Phi_2 = \{\sim p\} \dot{\cup} \alpha_2$, and $\hat{c}(\text{root } \mathfrak{J}) = \alpha_1 \cup \alpha_2$, where p is some atom, and α_1 and α_2 are clauses.

For $i = 1, 2,$ we may apply the inductive hypothesis to $\mathfrak{J}_i$

to obtain a tree $\mathfrak{T}_i'$ satisfying the conditions of the lemma; we specify that $\mathfrak{T}_i'$ is to be $\mathfrak{T}_i$ and f is to be the identity map if $d\Lambda = \hat{c}\Lambda$ for every leaf Λ of $\mathfrak{T}_i$.

If p does not occur in $\hat{c}(\text{root } \mathfrak{T}_1')$, let $\mathfrak{T}'$ be $\mathfrak{T}_1'$. Similarly, if $\sim p$ does not occur in $\hat{c}(\text{root } \mathfrak{T}_2')$, we may let $\mathfrak{T}'$ be $\mathfrak{T}_2'$.

The only remaining case is that in which $\hat{c}(\text{root } \mathfrak{T}_1') = \{p\} \dot{\cup} \alpha_1'$ and $\hat{c}(\text{root } \mathfrak{T}_2') = \{\sim p\} \dot{\cup} \alpha_2'$ for some clauses α_1' and α_2' such that $\alpha_1' \subseteq \alpha_1$ and $\alpha_2' \subseteq \alpha_2$. We construct the tree $\mathfrak{T}'$ from $\mathfrak{T}_1', \mathfrak{T}_2'$, and a node Ψ with $\hat{c}\Psi = \alpha_1' \cup \alpha_2'$, by letting Ψ be the root of $\mathfrak{T}'$, and letting $\text{root}(\mathfrak{T}_1')$ and $\text{root}(\mathfrak{T}_2')$ be the nodes immediately above Ψ in $\mathfrak{T}'$. We define the map f for $\mathfrak{T}'$ to agree with the maps for $\mathfrak{T}_1'$ and $\mathfrak{T}_2'$, and see that condition (1) is satisfied.

To check condition (4), note that if $\mathfrak{T}_i' \neq \mathfrak{T}_i$ for either $i = 1$ or $i = 2$, then $\text{root}(\mathfrak{T}_i')$ is $\leq$ some leaf in $f^{-1}(\mathcal{K} \cap \text{range } f)$, so Ψ is below some such leaf in $\mathfrak{T}'$. On the other hand if $\mathfrak{T}_1' = \mathfrak{T}_1$ and $\mathfrak{T}_2' = \mathfrak{T}_2$, then since $\text{root}(\mathfrak{T})$ is below some leaf $\Lambda \in \mathcal{K}$, $\text{root}(\mathfrak{T}')$ is below $f^{-1}\Lambda$. (Here we use the fact that a resolution tree is finite, so Λ must be in range f since f is one-one.) Thus (4) and (5) are satisfied. $c(\text{root } \mathfrak{T}') = \alpha_1' \cup \alpha_2' \subseteq \alpha_1 \cup \alpha_2 = c(\text{root } \mathfrak{T})$. Now if Ψ is a merge of $\mathfrak{T}'$, then α_1' and α_2' contain a common literal, so α_1 and α_2 do also, and $\text{root}(\mathfrak{T})$ is a merge of $\mathfrak{T}$. Since $\hat{m}(\mathfrak{T}_1') + \hat{m}(\mathfrak{T}_2') \leq \hat{m}(\mathfrak{T}_1) + \hat{m}(\mathfrak{T}_2)$ we see that $\hat{m}(\mathfrak{T}') \leq \hat{m}(\mathfrak{T})$. Thus $\mathfrak{T}'$ is the desired tree.

Lemma 3. Let $\mathfrak{T}$ be a ground resolution tree with $\hat{c}(\text{root } \mathfrak{T}) = \{q\} \dot{\cup} \alpha$, where q is a literal. Suppose the occurrence of q in $\text{root}(\mathfrak{T})$ has exactly one ancestor which is in a leaf of $\mathfrak{T}$. Call this leaf Ψ. Then there is a ground resolution tree $\mathfrak{T}'$ and an order-isomorphism f from $\mathfrak{T}$ onto $\mathfrak{T}'$ such that:

(1) $\hat{c}f\Phi = \hat{c}\Phi - \{q\}$ if Φ is any node of $\mathfrak{T}$ such that $\Phi \leq \Psi$;

(2) $\hat{c}f\Phi = \hat{c}\Phi$ if it is not the case that $\Phi \leq \Psi$ in $\mathfrak{T}$.

(3) If Φ is any node of $\mathfrak{T}$ which is not a leaf, the atom resolved upon in obtaining $\hat{c}\Phi$ from the clauses of the nodes immediately above Φ is the same as the atom resolved upon in obtaining $\hat{c}f\Phi$ in $\mathfrak{T}'$. Moreover Φ is a merge if and only if $f\Phi$ is a merge, and, if so, the merge literals are the same.

Proof: by induction on the number of nodes in $\mathfrak{T}$. If there is just one node in $\mathfrak{T}$, the lemma is obvious.

Suppose that $\mathfrak{T}$ has at least three nodes. Let Φ_1 and Φ_2 be the nodes immediately above $\text{root}(\mathfrak{T})$; let $\mathfrak{T}_1$ and $\mathfrak{T}_2$ be the sub-trees of $\mathfrak{T}$ rooted in Φ_1 and Φ_2, respectively. Without loss of generality we may suppose that Ψ is a leaf of $\mathfrak{T}_1$, $\hat{c}\Phi_1 = \{p,q\} \dot{\cup} \alpha_1$, and $\hat{c}\Phi_2 = \{\sim p\} \dot{\cup} \alpha_2$, where p is an atom, and α_1 and α_2 are clauses such that $\alpha_1 \cup \alpha_2 = \alpha$. q must be distinct from p, but q may be $\sim p$. Also q cannot occur in α_2, since the occurrence of q in $\text{root}(\mathfrak{T})$ has no ancestors in clauses of nodes in $\mathfrak{T}_2$.

Applying the inductive hypothesis to $\mathfrak{T}_1$, we obtain a tree $\mathfrak{T}_1'$ and an order isomorphism f from $\mathfrak{T}_1$ onto $\mathfrak{T}_1'$ satisfying conditions (1)-(3). Let $\mathfrak{T}'$ be the tree constructed from $\mathfrak{T}_1'$, $\mathfrak{T}_2$, and a clause Γ with $\hat{c}\Gamma = \alpha$, by letting $\Gamma = \text{root}(\mathfrak{T}')$ and letting root $\mathfrak{J}_1'$ and Φ_2 be the nodes immediately above Γ in $\mathfrak{J}'$. Since $\hat{c}(\text{root } \mathfrak{J}_1') = \{p\} \dot{\cup} \alpha_1$ we see that $\mathfrak{J}_1'$ is a resolution tree. Extend f to be an order isomorphism from $\mathfrak{J}$ onto $\mathfrak{J}'$ by letting f be the identity function on nodes of $\mathfrak{J}_2$, and $f(\text{root } \mathfrak{J}) = \Gamma$. Clearly conditions (1) and (2) are satisfied, and to check (3) it suffices to consider the node $\text{root}(\mathfrak{J})$. But the atom resolved upon in obtaining both $\hat{c}(\text{root } \mathfrak{J})$ and $\hat{c}(\text{root } \mathfrak{J}')$ is p, and in each case the clause is a merge if and only if α_1 and α_2 contain a common literal. Thus $\mathfrak{J}'$ is the desired tree.

<u>Lemma</u> 4. Let $\mathfrak{J}$ be a ground resolution tree, and let Λ be a leaf of $\mathfrak{J}$. Let α be a clause disjoint from $\hat{c}\Lambda$. Then there is a ground resolution tree $\mathfrak{J}'$ and an order-isomorphism g from $\mathfrak{J}$ onto $\mathfrak{J}'$ such that

(1) $\hat{c}g\Lambda = \alpha \dot{\cup} \hat{c}\Lambda$;

(2) if Φ is any node of $\mathfrak{J}$ which is below Λ, there is a (unique) clause $\gamma \subseteq \alpha$ such that $\hat{c}g\Phi = \gamma \dot{\cup} \hat{c}\Phi$; moreover, either $\gamma = \alpha$ or $\mathfrak{J}'$ contains a merge;

(3) if Φ is any node of $\mathfrak{J}$ which is not $\leq \Lambda$, then $\hat{c}g\Phi = \hat{c}\Phi$;

(4) if Φ is any node of $\mathfrak{J}$ which is a merge, then $g\Phi$ is a merge of $\mathfrak{J}'$.

Proof: by induction on the number of nodes in $\mathfrak{J}$. If $\mathfrak{J}$ has just one node, let $\mathfrak{J}'$ be the tree with a single node whose clause is $\alpha \cup \hat{c}\Lambda$.

Suppose $\mathfrak{J}$ has three or more nodes. Let Φ_1 and Φ_2 be the nodes immediately above $\text{root}(\mathfrak{J})$, and let $\mathfrak{J}_1$ and $\mathfrak{J}_2$ be the trees rooted in Φ_1 and Φ_2, respectively. Without loss of generality we may assume that Λ is a leaf of $\mathfrak{J}_1$, $\hat{c}\Phi_1 = \{p\} \dot{\cup} \beta_1$, $\hat{c}\Phi_2 = \{\sim p\} \dot{\cup} \beta_2$, and $\hat{c}(\text{root } \mathfrak{J}) = \beta_1 \cup \beta_2$, where p is an atom, and β_1 and β_2 are clauses.

Applying the inductive hypothesis to $\mathfrak{J}_1$, we obtain a tree $\mathfrak{J}_1'$ and an order isomorphism g_1 from $\mathfrak{J}_1$ onto $\mathfrak{J}_1'$ satisfying the conditions of the lemma. There is a clause $\gamma \subseteq \alpha$ such that $\hat{c}(\text{root } \mathfrak{J}_1') = \hat{c}g_1\Phi_1 = \gamma \dot{\cup} \hat{c}\Phi_1 = \gamma \dot{\cup}(\{p\} \cup \beta_1)$. Let $\mathfrak{J}'$ be the tree obtained from $\mathfrak{J}_1'$ and $\mathfrak{J}_2$ by putting (as root $(\mathfrak{J}')$) a node immediately below root$(\mathfrak{J}_1')$ and root$(\mathfrak{J}_2)$, with clause $\gamma \cup \beta_1 \cup \beta_2$. Clearly there is an order isomorphism g from $\mathfrak{J}$ onto $\mathfrak{J}'$: $g\Phi = g_1\Phi$ if $\Phi \in \mathfrak{J}_1$; $g\Phi = \Phi$ if $\Phi \in \mathfrak{J}_2$; $g(\text{root}(\mathfrak{J})) = \text{root}(\mathfrak{J}')$. Hence $\hat{c}g\Lambda = \alpha \dot{\cup} \hat{c}\Lambda$. Clearly if Φ is a node of $\mathfrak{J}$ which is not $\leq \Lambda$, then $\Phi \in \mathfrak{J}_2$, or $\Phi \in \mathfrak{J}_1$ but not $\leq \Lambda$; therefore it is easy to see that $\hat{c}g\Phi = \hat{c}\Phi$.

The clauses of $\mathfrak{J}$ below Λ are root$(\mathfrak{J})$ and clauses below Λ in $\mathfrak{J}_1$. Hence we need check condition (2) only for root$(\mathfrak{J})$. Let $\delta = \gamma - \beta_2$. Then $\delta \subseteq \gamma \subseteq \alpha$, and δ is disjoint from $\beta_1 \cup \beta_2$ (since we already have γ disjoint from $\{p\} \cup \beta_1$). Hence $\hat{c}g(\text{root } \mathfrak{J}) = \hat{c}(\text{root } \mathfrak{J}') = \gamma \cup \beta_1 \cup \beta_2 = \delta \dot{\cup} (\beta_1 \cup \beta_2)$. We must show that if $\delta \neq \alpha$ then $\mathfrak{J}'$ contains a merge. If $\gamma \neq \alpha$ we already know that $\mathfrak{J}_1'$ contains a merge, so $\mathfrak{J}'$ does also. Hence we need only consider the case where $\gamma = \alpha$. Thus $\alpha \neq \delta = \alpha-\beta_2$, so α and β_2 contain a common literal q, which is distinct from p (since $\gamma = \alpha$ and p is not in γ) and from $\sim$p (since $\sim$p is not in β_2). Write $\alpha = \{q\} \dot{\cup} \alpha'$ and $\beta_2 = \{q\} \dot{\cup} \beta_2'$. Then $\hat{c}(\text{root } \mathfrak{J}_1') = \{p\} \dot{\cup} (\{q\} \cup \alpha' \cup \beta_1)$ and $\hat{c}\Phi_2 = \{\sim p\}\dot{\cup}(\{q\} \cup \beta_2')$, so it follows from the definition that root$(\mathfrak{J}')$ is a merge.

Clearly g preserves merges of $\mathfrak{J}_1$ (since g_1 does) and of $\mathfrak{J}_2$. So we need only check that if root$(\mathfrak{J})$ is a merge, then root$(\mathfrak{J}')$ is. But this is clear, since if β_1 and β_2 contain a common literal, then $\gamma \cup \beta_1$ and β_2 do also.

Thus $\mathfrak{J}'$ satisfies the conditions of the lemma.

Our next lemma is the crucial one. Actually we shall not need condition (5) of Lemma 5, but we include it for the sake of its own interest.

Lemma 5. Let $\mathfrak{I}$ be a ground resolution tree in which no merge literal has a descendant which is resolved upon. (More precisely, suppose there is no merge Φ of $\mathfrak{I}$ with a merge literal q in $\hat{c}\Phi$ such that some descendant of the occurrence of q in Φ is resolved upon.) Let Γ be any leaf of $\mathfrak{I}$. Then there is a ground resolution tree $\mathfrak{I}'$ such that:

(1) $\mathfrak{I}'$ is a vine;

(2) there is a one-one mapping h from the leaves of $\mathfrak{I}'$ into the leaves of $\mathfrak{I}$ such that $\hat{c}h\Lambda = \hat{c}\Lambda$ for each leaf Λ of $\mathfrak{I}'$;

(3) there is a top-leaf Γ' of $\mathfrak{I}'$ such that $h\Gamma' = \Gamma$;

(4) $\hat{c}(\text{root } \mathfrak{I}') \subseteq \hat{c}(\text{root } \mathfrak{I})$;

(5) either $\hat{c}(\text{root } \mathfrak{I}') = \hat{c}(\text{root } \mathfrak{I})$ or $\mathfrak{I}'$ contains at least one merge.

Proof: by induction on the number of nodes in $\mathfrak{I}$. If the number of nodes is 1 or 3, let $\mathfrak{I}'$ be $\mathfrak{I}$, and h the identity map.

Suppose $\mathfrak{I}$ has more than 3 nodes. Let Φ_1 and Φ_2 be the nodes immediately above the root of $\mathfrak{I}$, and let $\mathfrak{I}_1$ and $\mathfrak{I}_2$ be the sub-trees of $\mathfrak{I}$ rooted in Φ_1 and Φ_2, respectively. Without loss of generality we may assume that Γ is a leaf of $\mathfrak{I}_1$, $\hat{c}\Phi_1 = \{p\} \dot{\cup} \alpha_1$, $\hat{c}\Phi_2 = \{\sim p\} \cup \alpha_2$, and $\hat{c}(\text{root } \mathfrak{I}) = \alpha_1 \cup \alpha_2$, where p is an atom, and α_1 and α_2 are clauses.

Applying the inductive hypothesis to $\mathfrak{I}_1$, there is a vine $\mathfrak{I}_1'$, a clause-preserving one-one map h_1 from the leaves of $\mathfrak{I}_1'$ into the leaves of $\mathfrak{I}_1$, and a top-leaf Γ' of $\mathfrak{I}_1'$ such that $h_1\Gamma' = \Gamma$; also $\hat{c}(\text{root } \mathfrak{I}_1') \subseteq \{p\} \cup \alpha_1$. If the atom p does not occur in

$\hat{c}(\text{root } \mathfrak{T}_1')$, then $\hat{c}(\text{root } \mathfrak{T}_1') \subseteq \alpha_1 \subseteq \alpha_1 \cup \alpha_2 = \hat{c}(\text{root } \mathfrak{T})$, and $\mathfrak{T}_1'$ contains a merge, so let $\mathfrak{T}'$ be $\mathfrak{T}_1'$.

Next we consider the case where p does occur in $c(\text{root } \mathfrak{T}_1')$. Then $\hat{c}(\text{root } \mathfrak{T}_1') = \{p\} \dot{\cup} \alpha_1'$ for some clause $\alpha_1' \subseteq \alpha_1$. If $\alpha_1' \neq \alpha_1$ then $\mathfrak{T}_1'$ contains a merge. The literal $\sim p$ in Φ_2 is resolved upon in $\mathfrak{T}$ to obtain $\hat{c}(\text{root } \mathfrak{T})$, so this occurrence of $\sim p$ has no ancestor which is a merge literal of $\mathfrak{T}$. Hence there is a unique leaf Ψ_2 of $\mathfrak{T}_2$ which contains an ancestor of the occurrence of $\sim p$ in Φ_2. Write $\hat{c}\Psi_2 = \{\sim p\} \dot{\cup} \beta$. (See Figure 2.) (If $\beta = \square$ then $\Psi_2 = \Phi_2$ and $\mathfrak{T}_2$ has just one node.) Hence we may apply Lemma 3 to obtain from $\mathfrak{T}_2$ a tree $\mathfrak{T}_3$ and an order isomorphism f from $\mathfrak{T}_2$ onto $\mathfrak{T}_3$ satisfying the conditions of Lemma 3, with $\hat{c}f\Psi_2 = \beta$ and $\hat{c}(\text{root } \mathfrak{T}_3) = \hat{c}f\Phi_2 = \alpha_2$.

By condition (3) of Lemma 3 we see that no merge literal of $\mathfrak{T}_3$ has a descendant which is resolved upon (since $\mathfrak{T}_2$ inherited the same property from $\mathfrak{T}$), and $\mathfrak{T}_3$ has fewer nodes than $\mathfrak{T}$, so we may apply our inductive hypothesis to $\mathfrak{T}_3$, with $\Psi_3 = f\Psi_2$ (definition) as the designated leaf of $\mathfrak{T}_3$. Thus we obtain a vine $\mathfrak{T}_4$, a one-one clause-preserving map h_4 from the leaves of $\mathfrak{T}_4$ into the leaves of $\mathfrak{T}_3$, and a top-leaf Ψ_4 of $\mathfrak{T}_4$ such that $h_4\Psi_4 = \Psi_3$ so $\hat{c}\Psi_4 = \hat{c}\Psi_3 = \beta$. Let $\alpha_2' = \hat{c}(\text{root } \mathfrak{T}_4) \subseteq \hat{c}(\text{root } \mathfrak{T}_3) = \alpha_2$. If $\alpha_2' \neq \alpha_2$ then $\mathfrak{T}_4$ contains a merge.

Finally we apply Lemma 4 to $\mathfrak{T}_4$, with Ψ_4 as the designated leaf of $\mathfrak{T}_4$ and $\alpha_1' - \beta$ as the designated clause. Thus we obtain a vine $\mathfrak{T}_5$ and an order-isomorphism g from $\mathfrak{T}_4$ onto $\mathfrak{T}_5$, such that there is a top-leaf $\Psi_5 = g\Psi_4$ of $\mathfrak{T}_5$ with $\hat{c}\Psi_5 = \hat{c}g\Psi_4 = (\alpha_1' - \beta) \cup \hat{c}\Psi_4 = \alpha_1' \cup \beta$, but $\hat{c}g\Omega = \hat{c}\Omega$ if Ω is any leaf of $\mathfrak{T}_4$ other than Ψ_4. Moreover there is a clause $\alpha_1'' \subseteq (\alpha_1' - \beta)$ such

Figure 2: $\mathfrak{T}$

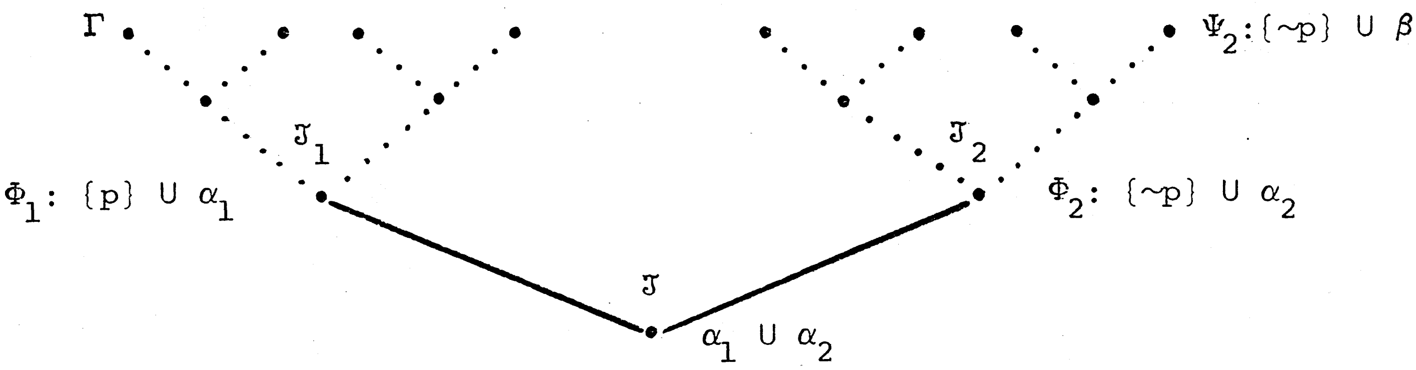
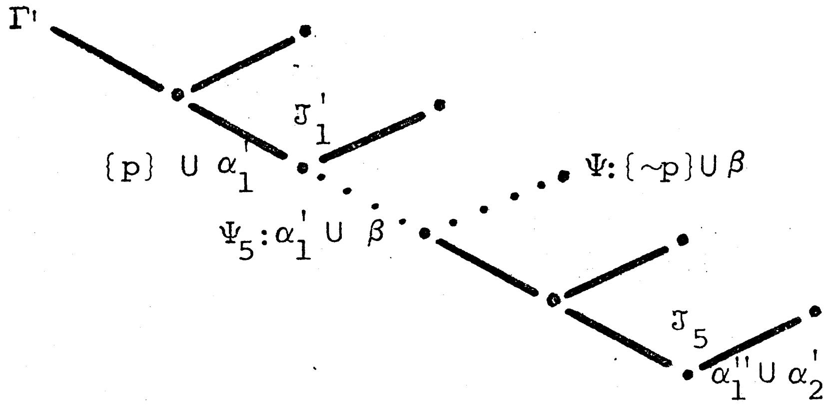


Figure 3: Proof of Lemma 5

$$\mathfrak{T}_2 \xrightarrow{f} \mathfrak{T}_3 \xleftarrow{h_4} \mathfrak{T}_4\,\text{(vine)} \xrightarrow{g} \mathfrak{T}_5$$

$\hat{c}\Psi_i$:	$\{\sim p\} \cup \beta$	β	β	$\alpha_1' \cup \beta$
root clause:	$\{\sim p\} \cup \alpha_2$	α_2	α_2'	$\alpha_1'' \cup \alpha_2'$

Figure 4: $\mathfrak{T}'$



that $\hat{c}(\text{root } \mathfrak{T}_5) = \hat{c}g(\text{root } \mathfrak{T}_4) = \alpha_1'' \dot{\cup} \hat{c}(\text{root } \mathfrak{T}_4) = \alpha_1'' \dot{\cup} \alpha_2'$, and if $\alpha_1'' \neq \alpha_1' - \beta$ then $\mathfrak{T}_5$ contains a merge.

We summarize the relation between $\mathfrak{T}_2$ and $\mathfrak{T}_5$. (See Figure 3.) $\mathfrak{T}_2$ has a leaf Ψ_2 with $\hat{c}\Psi_2 = \{\sim p\} \dot{\cup} \beta$, and root Φ_2, with $\hat{c}\Phi_2 = \{\sim p\} \dot{\cup} \alpha_2$. $\mathfrak{T}_5$ is a vine with a top-leaf Ψ_5, with $\hat{c}\Psi_5 = \alpha_1' \cup \beta$, and $\hat{c}(\text{root } \mathfrak{T}_5) = \alpha_1'' \dot{\cup} \alpha_2'$, where $\alpha_1'' \subseteq \alpha_1' - \beta \subseteq \alpha_1' \subseteq \alpha_1$ and $\alpha_2' \subseteq \alpha_2$. There is a one-one map $h_5 = f^{-1} \circ h_4 \circ g^{-1}$ from the leaves of $\mathfrak{T}_5$ into the leaves of $\mathfrak{T}_2$ such that $h_5\Psi_5 = \Psi_2$, and $\hat{c}h_5\Omega = \hat{c}\Omega$ if Ω is any leaf of $\mathfrak{T}_5$ other than Ψ_5.

Now we construct the tree $\mathfrak{T}'$ from the trees $\mathfrak{T}_5$, $\mathfrak{T}_1'$, and a node which we call Ψ, with $\hat{c}\Psi = \{\sim p\} \cup \beta$, by putting the root of $\mathfrak{T}_1'$ and Ψ immediately above the leaf Ψ_5 of $\mathfrak{T}_5$. (See Figure 4.) Note that $\hat{c}(\text{root } \mathfrak{T}_1') = \{p\} \cup \alpha_1'$ and $\hat{c}\Psi = \{\sim p\} \cup \beta$ have as a resolvent the clause $\alpha_1' \cup \beta = \hat{c}\Psi_5$, so the tree $\mathfrak{T}'$ is a resolution tree. A node of $\mathfrak{T}'$ is a non-leaf of $\mathfrak{T}'$ if and only if it is Ψ_5 or a non-leaf of $\mathfrak{T}_1'$ or of $\mathfrak{T}_5$; hence it is easy to see that $\mathfrak{T}'$, like $\mathfrak{T}_1'$ and $\mathfrak{T}_5$, is a vine. (A node immediately below Ψ_5 in $\mathfrak{T}_5$ is also immediately below the other top-leaf of $\mathfrak{T}_5$, which is still a leaf in $\mathfrak{T}'$.) The leaves of $\mathfrak{T}'$ are the leaves of $\mathfrak{T}_1'$, Ψ, and the leaves of $\mathfrak{T}_5$ other than Ψ_5; hence we can define a one-one clause-preserving map h from the leaves of $\mathfrak{T}'$ into the leaves of $\mathfrak{T}$ by setting $h\Lambda = h_1\Lambda$ if Λ is a leaf of $\mathfrak{T}_1'$, $h\Psi = \Psi_2$, and $h\Lambda = h_5\Lambda$ if Λ is a leaf of $\mathfrak{T}_5$ other than Ψ_5. Note that Γ' is a top-leaf of $\mathfrak{T}'$ with $h\Gamma' = h_1\Gamma' = \Gamma$, and that $\hat{c}(\text{root } \mathfrak{T}') = \hat{c}(\text{root } \mathfrak{T}_5) = \alpha_1'' \cup \alpha_2' \subseteq \alpha_1 \cup \alpha_2 = \hat{c}(\text{root } \mathfrak{T})$.

Finally we must check that $\mathfrak{T}'$ satisfies condition (5). If $\alpha_1' \neq \alpha_1$ then $\mathfrak{T}_1'$ contains a merge, so $\mathfrak{T}'$ does also. Thus we need consider only the case where $\alpha_1' = \alpha_1$. If α_1' and β

contain a common literal, then Ψ_5 is a merge of $\mathfrak{J}'$. Thus we need consider only the case where α_1' and β are disjoint, so $\alpha_1' - \beta = \alpha_1' = \alpha_1$. If $\alpha_1'' \neq \alpha_1' - \beta$ then $\mathfrak{J}_5$ contains a merge, so $\mathfrak{J}'$ does also. Hence we need consider only the case where $\alpha_1'' = \alpha_1$. If $\alpha_2' = \alpha_2$ then $\mathfrak{J}_4$ contains a merge, so by clause (4) of Lemma 4, $\mathfrak{J}_5$ does also, so $\mathfrak{J}'$ contains a merge. Thus we need consider only the case where $\alpha_2' = \alpha_2$. But then $\hat{c}(\text{root } \mathfrak{J}') = \alpha_1'' \cup \alpha_2' = \alpha_1 \cup \alpha_2 = \hat{c}(\text{root } \mathfrak{J})$, so $\mathfrak{J}'$ satisfies condition (5).

Thus $\mathfrak{J}'$ is the desired tree.

<u>Remark</u>: As often happens in inductive proofs, we have essentially given an algorithm for transforming the tree $\mathfrak{J}$ into the vine $\mathfrak{J}'$. The reader may find it enlightening to carry out this transformation on some examples.

<u>Theorem</u> 6. Let K and S be sets of ground clauses with $K \subseteq S$. Let $K_o = \hat{\hat{ms}}(K,S)$ and $S_o = S \cup K_o$. Let $\mathfrak{J}$ be a ground resolution tree with K_o-support such that $\hat{c}(\text{root } \mathfrak{J}) = \square$ and $\hat{c}\Lambda \in S_o$ for each leaf Λ of $\mathfrak{J}$. Then there is a ground resolution tree $\mathfrak{J}'$ such that:

(1) $\mathfrak{J}'$ is a vine;

(2) $\mathfrak{J}'$ has K_o-support;

(3) $\hat{c}\Lambda \in S_o$ for each leaf Λ of $\mathfrak{J}'$;

(4) $\hat{c}(\text{root } \mathfrak{J}') = \square$.

Proof: The proof is by induction on $\hat{m}(\mathfrak{J})$.

If $\mathfrak{J}$ has just one node let $\mathfrak{J}'$ be $\mathfrak{J}$. If $\mathfrak{J}$ has more than one node, then it has a leaf whose clause is in K_o. If $\mathfrak{J}$ has no merges, but Γ be such a leaf and apply Lemma 5 to obtain a vine $\mathfrak{J}'$ with root clause $\square$ and with leaves whose clauses are in S_o, with a top leaf Γ' whose clause is in K_o. Thus $\mathfrak{I}'$ has K_o-support, as desired.

It remains to deal with the case where $\mathfrak{I}$ contains a merge. Let Ψ be a merge of $\mathfrak{I}$ which is below no other merge. Let $\mathfrak{I}_1$ be the sub-tree of $\mathfrak{I}$ with root Ψ. Ψ has K_o-support in $\mathfrak{I}$, so some leaf Γ of $\mathfrak{I}_1$ has its clause in K_o. Clearly $\mathfrak{I}_1$ satisfies the hypothesis of Lemma 5, since its sole merge is its root. Let $\mathfrak{I}_1'$ be a vine obtained as in Lemma 5 from $\mathfrak{I}_1$, with Γ as the designated leaf of $\mathfrak{I}_1$. Note that $\mathfrak{I}_1'$ has a top-leaf Γ' such that $\hat{c}\Gamma' = \hat{c}\Gamma \in K_o$.

If $\mathfrak{I}_1'$ contains a merge, then (since $\mathfrak{I}_1'$ is a vine) it contains a minimal merge Ω. Since $\mathfrak{I}_1'$ is a vine with leaves whose clauses are in S_o and a top leaf Γ' whose clause is in K_o, $\mathfrak{I}_1'$ represents a deduction from S_o by resolution with merging with K_o-support. Since Ω is a merge of $\mathfrak{I}_1'$, we see that $\hat{c}\Omega \in \hat{\hat{ms}}(K_o,S_o)$. But by Lemma 1, $\hat{\hat{ms}}(K_o,S_o) = \hat{\hat{ms}}(\hat{\hat{ms}}(K,S), S \cup \hat{\hat{ms}}(K,S)) = \hat{\hat{ms}}(K,S) = K_o$. Let $\mathfrak{I}_2$ be the tree obtained from $\mathfrak{I}_1'$ by pruning away all nodes above Ω. (Of course root $\mathfrak{I}_2$ may be Ω, in which case $\mathfrak{I}_2$ contains a single node). If, however, $\mathfrak{I}_1'$ does not contain a merge, let $\mathfrak{I}_2$ be $\mathfrak{I}_1'$. Then in either case $\mathfrak{I}_2$ is a vine with a top-leaf (Ω or Γ') whose clause is in K_o, $\mathfrak{I}_2$ contains no merges, the clauses of the leaves of $\mathfrak{I}_2$ are all in S_o, and $\hat{c}(\text{root } \mathfrak{I}_2) = \hat{c}(\text{root } \mathfrak{I}_1') \subseteq \hat{c}(\text{root } \mathfrak{I}_1) = \hat{c}\Psi$.

We wish to replace $\mathfrak{I}_1$ by $\mathfrak{I}_2$ within $\mathfrak{I}$, but we cannot do this directly if $\hat{c}(\text{root } \mathfrak{I}_2) \neq \hat{c}\Psi$. So let $\mathfrak{I}_3$ be the tree obtained from $\mathfrak{I}$ by pruning away all nodes of $\mathfrak{I}$ which are above Ψ; thus Ψ becomes a leaf of $\mathfrak{I}_3$. We now wish to apply Lemma 2 to obtain from $\mathfrak{I}_3$ a tree $\mathfrak{I}_4$ in which the node Ψ is replaced by a node Ψ' whose clause is $\hat{c}(\text{root } \mathfrak{I}_2)$. So define d on the leaves of $\mathfrak{I}_3$

so that $d\Psi = \hat{c}(\text{root } \mathfrak{T}_2)$, and $d\Lambda = \hat{c}\Lambda$ if Λ is a leaf of $\mathfrak{T}_3$ other than Ψ. Let $\mathcal{K} = \{\Lambda | \Lambda$ is a leaf of $\mathfrak{T}_3$, and $\Lambda = \Psi$ or $\hat{c}\Lambda \in K_o\}$. Since $\mathfrak{T}$ has K-support we see that $\mathfrak{T}_3$, d, and $\mathcal{K}$ satisfy the hypotheses of Lemma 2, so we obtain the desired tree $\mathfrak{T}_4$ and map f as in Lemma 2.

Thus if $\mathfrak{T}_4$ has leaf Ψ' such that $f\Psi' = \Psi$, we see that $\hat{c}\Psi' = df\Psi' = d\Psi = \hat{c}(\text{root } \mathfrak{T}_2)$; let $\mathfrak{T}^*$ be the tree obtained by grafting $\mathfrak{T}_2$ onto $\mathfrak{T}_4$ so that root$(\mathfrak{T}_2)$ is identified with Ψ'. If the leaf Ψ of $\mathfrak{T}_3$ is not in the range of f, let $\mathfrak{T}^*$ be $\mathfrak{T}_4$. Note that the leaves of $\mathfrak{T}^*$ are all leaves of $\mathfrak{T}_2$, or leaves of $\mathfrak{T}_4$ other than Ψ', so it is easy to see that $\hat{c}\Lambda \in S_o$ for each leaf Λ of $\mathfrak{T}^*$. By clause (4) of Lemma 2, each non-leaf of $\mathfrak{T}_4$ is below Ψ' or a leaf whose clause is in K_o; but (if there is a node Ψ' in $\mathfrak{T}_4$) $\Psi' \leq$ the top-leaf of $\mathfrak{T}_2$ in $\mathfrak{T}^*$, so $\mathfrak{T}^*$ clearly has K_o-support. Also $\hat{c}(\text{root } \mathfrak{T}^*) = \hat{c}(\text{root } \mathfrak{T}_4) \subseteq \hat{c}(\text{root } \mathfrak{T}_3) = \hat{c}(\text{root } \mathfrak{T}) = \square$ so $\hat{c}(\text{root } \mathfrak{T}^*) = \square$. Now $\hat{m}(\mathfrak{T}^*) = \hat{m}(\mathfrak{T}_4) + \hat{m}(\mathfrak{T}_2)$ $= \hat{m}(\mathfrak{T}_4) \leq \hat{m}(\mathfrak{T}_3) = \hat{m}(\mathfrak{T}) - 1$, since Ψ is a merge of $\mathfrak{T}$ but not of $\mathfrak{T}_3$. Hence $\hat{m}(\mathfrak{T}^*) < \hat{m}(\mathfrak{T})$, so since $\mathfrak{T}^*$ satisfies the hypotheses of our theorem we obtain the desired tree $\mathfrak{T}'$.

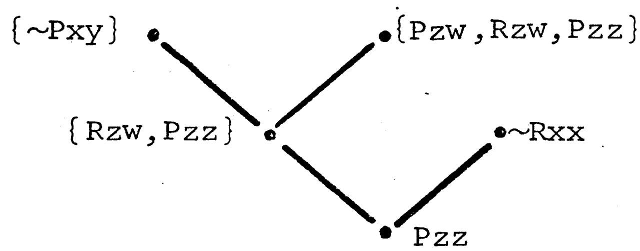
<u>Theorem</u> 7. Let K and S be sets of ground clauses with $K \subseteq S$ such that there is a refutation of S with K-support. Then there is a refutation of S by resolution with merging with K-support.

Proof:

From the given refutation of S with K-support we can construct a ground resolution tree with K-support with root clause $\square$ and with leaves whose clauses are in S. Let $K_o = \hat{m}\hat{s}(K,S)$ and $S_o = S \cup K_o$. Since $K \subseteq K_o$ and $S \subseteq S_o$ this tree satisfies the hypothesis of Theorem 6, so there is a vine $\mathfrak{J}'$ as described in Theorem 6. Let $K_1 = K_o \cap \{\gamma | \gamma$ is the clause of some leaf of $\mathfrak{J}'\}$. Then $\mathfrak{J}'$ represents a refutation by resolution with merging of $S \cup K_1$ with K_1-support. Since $K_1 \subseteq \hat{\hat{ms}}(K,S)$, for each clause γ in K_1 we can obtain a deduction of γ from S by resolution with merging with K-support, in which the final triple is $<\gamma, \hat{m}, \hat{s}>$. K_1 is finite, so we can place these deductions one after another, and add at the end the refutation represented by $\mathfrak{J}'$. This gives a single deduction which is a refutation of S by resolution with merging with K-support.

Having established our main theorem for the case where the given clauses are ground clauses, we must extend it to the general case. The obvious approach is to transform a refutation of a given set S of clauses to a refutation of a set S_o of ground clauses obtained by instantiating the clauses of S, then apply Theorem 7 to get a refutation of S_o by resolution with merging, and from this construct a refutation of S by resolution with merging; all this is to be done, of course, with due regard for the set of support. However, unless the set of support is trivial (in which case one can use the Ground Resolution Theorem of [2]), the first step in this process is not as simple as one might suppose. One might suppose that if $\mathfrak{J}$ is any resolution tree, then there is a ground resolution tree $\mathfrak{J}'$ order-isomorphic to $\mathfrak{J}$, such that the clause of each node of $\mathfrak{J}'$ is obtained from the corresponding clause of $\mathfrak{J}$ by instantiation. However, the tree in Figure 5 shows that this is false. This tree also provides a counter-example

Figure 5.



to the second part of Lemma 2 in [5].[3] Nevertheless, all we really need is to obtain from $\mathfrak{T}$ a ground resolution tree $\mathfrak{T}'$ with appropriate support, whose leaves have clauses in S_o. We shall show that we can do this in Lemma 8.

In discussing substitutions we shall use the terminology of section 5 of [2], with one minor change. A substitution can be regarded as a function mapping literals to literals, or sets of literals to sets of literals. In [2] these functions are written to the right of their arguments, but to maintain consistency with our previous usage we shall continue to write functions to the left of their arguments.

<u>Definition</u>. A <u>ground substitution</u> is a substitution in which the term of each substitution component is a ground term (i.e., contains no individual variables).

<u>Lemma</u> 8. Let $\mathfrak{T}$ be a resolution tree with $\hat{c}(\text{root } \mathfrak{T}) = \square$ and let $\mathfrak{K}$ be a subset of the leaves of $\mathfrak{T}$ such that every non-leaf of $\mathfrak{T}$ is below some leaf in $\mathfrak{K}$. Then there is a ground resolution tree $\mathfrak{T}'$ and a one-one mapping h from the leaves of $\mathfrak{T}'$ into the leaves of $\mathfrak{T}$ such that

(1) for each leaf Λ of $\mathfrak{T}'$ there is a ground substitution $\hat{G}$ such that $\hat{c}\Lambda = \hat{G}\hat{c}h\Lambda$;

(2) every non-leaf of $\mathfrak{T}'$ is below some leaf in $h^{-1}(\mathfrak{K} \cap \text{range } h)$.

(3) $c(\text{root } \mathfrak{T}') = \square$.

---

[3] Let $A = \{\sim Pxy\}$, $B = \{Pzw, Rzw, Pzz\}$, $C = \{Rzw, Pzz\}$, $C' = \{Raa, Paa\}$. Then C is a resolvent of A and B, and C' is obtained from C by instantiation, but there are no clauses A' and B' obtainable by instantiation from A and B, respectively, such that C' is a resolvent of A' and B'. Fortunately only the first part of Lemma 2 of [5] seems to be actually used in [5].

Proof: by induction on the number of nodes in $\mathfrak{T}$.

If $\mathfrak{T}$ has more than one node, let $\mathfrak{T}'$ be $\mathfrak{T}$.

Suppose $\mathfrak{T}$ has more than one node. Let Ω be a non-leaf of $\mathfrak{T}$ which is below no other non-leaf. Let Φ and Ψ be the leaves immediately above Ω in $\mathfrak{T}$. Let $\mathfrak{T}_1$ be the sub-tree of $\mathfrak{T}$ obtained by pruning off Φ and Ψ, so that Ω is a leaf of $\mathfrak{T}_1$. Let $\mathfrak{K}_1 = \{\Lambda | \Lambda$ is a leaf of $\mathfrak{T}_1$, and $\Lambda \in \mathfrak{K}$ or $\Lambda = \Omega\}$. Clearly every non-leaf of $\mathfrak{T}_1$ is below some leaf in $\mathfrak{K}_1$, so we can apply the inductive hypothesis to $\mathfrak{T}_1$ to obtain a ground resolution tree $\mathfrak{T}_1'$ and map h_1 from the leaves of $\mathfrak{T}_1'$ into the leaves of $\mathfrak{T}_1$ satisfying conditions (1) - (3). If $\Omega \notin$ range h_1 then $\mathfrak{K}_1 \cap$ range $h_1 \subseteq \mathfrak{K}$ so we can take $\mathfrak{T}'$ to be $\mathfrak{T}_1'$.

Suppose $\Omega \in$ range h_1, so there is a leaf Ω' of $\mathfrak{T}_1'$ such that $h_1\Omega' = \Omega$. Then there is a ground substitution $\hat{G}$ such that $\hat{c}\Omega' = \hat{G}\hat{c}h\Omega' = \hat{G}\hat{c}\Omega$. There exist substitutions $\hat{A}$ and $\hat{B}$, an atom p, and clauses δ and ϵ such that $\hat{A}\hat{c}\Phi = \{p\} \dot{\cup} \delta$ and $\hat{B}\hat{c}\Psi = \{\sim p\} \dot{\cup} \epsilon$ and $\hat{c}\Omega = \delta \cup \epsilon$, so $\hat{c}\Omega' = \hat{G}(\delta \cup \epsilon)$. Let $\mathfrak{S}$ be the set of all variables which occur in p or occur in the term of some substitution component of $\hat{A}$ or of $\hat{B}$. Let $\hat{H}$ be the substitution $\{ \frac{a}{x} \mid x$ is an individual variable in $\mathfrak{S}\}$, where $\underline{a}$ is a fixed individual constant. Clearly the composite substitutions $\hat{H}\circ\hat{G}\circ\hat{A}$ and $\hat{H}\circ\hat{G}\circ\hat{B}$ are ground substitutions. Let Φ' be a node with $\hat{c}\Phi' = \hat{H}\hat{G}\hat{A}\hat{c}\Phi = \{\hat{H}\hat{G}p\} \cup \hat{H}\hat{G}\delta$ and let Ψ' be a node with $\hat{c}\Psi' = \hat{H}\hat{G}\hat{B}\hat{c}\Psi = \{\sim\hat{H}\hat{G}p\} \cup \hat{H}\hat{G}\epsilon$. Since the literals in $\hat{G}(\delta \cup \epsilon)$ are in $\hat{c}\Omega'$ they are ground literals, so $\hat{H}\hat{G}\delta = \hat{G}\delta$ and $\hat{H}\hat{G}\epsilon = \hat{G}\epsilon$. Clearly $\hat{H}\hat{G}p$ is a ground literal. Thus $\hat{c}\Phi'$ and $\hat{c}\Psi'$ are ground clauses. Although p does not occur in δ, certain literals in δ may become identified with $\hat{H}\hat{G}p$ when instantiated by $\hat{G}$. Therefore let δ' be the subset of δ from which such literals have

<u>Figure</u> 6: Proof of Lemma 8

$$
\begin{array}{llllllll}
 & \mathfrak{T} \supseteq & \mathfrak{T}_1 \xrightarrow{h_1} & \mathfrak{T}_1' \xrightarrow{f} & \mathfrak{T}_2 & \subseteq & \mathfrak{T}' & \\
\Phi & \quad \Psi & & & & & \Phi' : \{\hat{H}\hat{G}p\} \cup \hat{G}\delta' & \Psi' : \{\sim\hat{H}\hat{G}p\} \cup \hat{G}\epsilon' \\
 & \Omega & \Omega : \delta \cup \epsilon & \Omega' : \hat{G}\delta \cup \hat{G}\epsilon & \Omega_2 & & \Omega_2 : \hat{G}\delta' \cup \hat{G}\epsilon' &
\end{array}
$$

been deleted, so that $\hat{c}\Phi' = \{\hat{H}\hat{G}p\} \dot{\cup} \hat{G}\delta'$. Similarly let $\epsilon' \subseteq \epsilon$ be such that $\hat{c}\Psi' = \{\sim\hat{H}\hat{G}p\} \dot{\cup} G\epsilon'$.

Now $\hat{G}\delta' \cup \hat{G}\epsilon' = \hat{G}(\delta' \cup \epsilon') \subseteq \hat{G}(\delta \cup \epsilon) = \hat{c}\Omega'$. We wish to replace $\mathfrak{I}_1'$ by a tree $\mathfrak{I}_2$ in which Ω' is replaced by a node with clause $\hat{G}\delta' \cup \hat{G}\epsilon'$. So we apply Lemma 2 to $\mathfrak{I}_1'$. Let $d\Lambda = \hat{c}\Lambda$ if Λ is any leaf of $\mathfrak{I}_1'$ other than Ω', while $d\Omega' = \hat{G}\delta' \cup \hat{G}\epsilon'$. Let $\mathcal{K}_1' = h_1^{-1}(\mathcal{K}_1 \cap \text{range } h_1)$. $\Omega' \in \mathcal{K}_1'$ and every non-leaf of $\mathfrak{I}_1'$ is below some leaf in $\mathcal{K}_1'$, so by Lemma 2 we obtain a ground resolution tree $\mathfrak{I}_2$ and a one-one map f from the leaves of $\mathfrak{I}_2$ into the leaves of $\mathfrak{I}_1'$ such that $\hat{c}\Lambda = df\Lambda$ for each leaf Λ of $\mathfrak{I}_2$, $\hat{c}$ root $\mathfrak{I}_2 = \square$, and each non-leaf of $\mathfrak{I}_2$ is below some leaf in $f^{-1}(\mathcal{K}_1' \cap \text{range } f)$.

Let $h_2 = h_1 \circ f$. Then h_2 is a one-one map from the leaves of $\mathfrak{I}_2$ into the leaves $\mathfrak{I}_1$, and every non-leaf of $\mathfrak{I}_2$ is below some leaf in $h_2^{-1}(\mathcal{K}_1 \cap \text{range } h_2)$. If $\Omega' \in \text{range } f$ let $\Omega_2 = f^{-1}\Omega'$. If Λ is any leaf of $\mathfrak{I}_2$ other than Ω_2 then, since $f\Lambda$ is a leaf of $\mathfrak{I}_1'$, there is a ground substitution $\hat{G}$ such that $\hat{c}\Lambda = df\Lambda = \hat{c}f\Lambda = \hat{G}\hat{c}h_1 f\Lambda = \hat{G}\hat{c}h_2\Lambda$. Now suppose Ω' is not in range f, so Ω is not in range h_2. Then h_2 maps the leaves of $\mathfrak{I}_2$ into the leaves of $\mathfrak{I}$, so we let $\mathfrak{I}'$ be $\mathfrak{I}_2$ and h be h_2.

On the other hand, suppose there is a leaf Ω_2 of $\mathfrak{I}_2$ such that $\Omega' = f\Omega_2$. Then $h_2\Omega_2 = h_1 f\Omega_2 = \Omega$. Also $\hat{c}\Omega_2 = df\Omega_2 = d\Omega' = \hat{G}\delta' \cup \hat{G}\epsilon'$. Let $\mathfrak{I}'$ be the tree obtained from $\mathfrak{I}_2$ by placing the nodes Φ' and Ψ' immediately above Ω'. Clearly $\mathfrak{I}'$ is a ground resolution tree with root clause $\square$. Let h be the extension of h_2 mapping the leaves of $\mathfrak{I}'$ into the leaves of $\mathfrak{I}$, such that $h\Phi' = \Phi$ and $h\Psi' = \Psi$. Since either Φ or Ψ must be in $\mathcal{K}$, it is easy to see that every non-leaf of $\mathfrak{I}'$ is below some leaf in $h^{-1}(\mathcal{K} \cap \text{range } h)$. Also it is easy to check that (1) is satisfied, so the proof is complete.

Theorem 9. Let K and S be sets of clauses with $K \subseteq S$ such that there is a refutation of S with K-support. Then there is a refutation of S by resolution with merging with K-support.

Proof:

From the given refutation of S with K-support we can construct a resolution tree $\mathfrak{T}_1$ with K-support with root clause $\square$ and with leaves whose clauses are in S. We let $\mathcal{K}$ be the set of leaves of $\mathfrak{T}$ whose clauses are in K. Then we can apply Lemma 8 to obtain a ground resolution tree $\mathfrak{T}_2$ with root clause $\square$ and a one-one map h from the leaves of $\mathfrak{T}_2$ into the leaves of $\mathfrak{T}_1$ as described in Lemma 8. Let S_2 be the set of clauses of the leaves of $\mathfrak{T}_2$, and let K_2 be the set of clauses of the leaves in $h^{-1}(\mathcal{K} \cap \text{range } h)$. Then $K_2 \subseteq S_2$, S_2 is a set of ground clauses, and $\mathfrak{T}_2$ represents a refutation of S_2 with K_2-support. For each clause $\beta \in S_2$ there is a clause $\alpha \in S$ and a ground substitution $\hat{G}$ such that $\beta = \hat{G}\alpha$.

By Theorem 7 we see that there is a refutation of S_2 with K_2-support by resolution with merging. For convenience we represent this refutation by a ground resolution tree $\mathfrak{T}_3$. Let $\mathfrak{T}_4$ be the tree obtained from $\mathfrak{T}_3$ upon replacing each leaf Λ of $\mathfrak{T}_3$ by a leaf Λ' with a clause $\alpha \in S$ such that there is a ground substitution $\hat{G}$ such that $\hat{G}\alpha = \hat{C}\Lambda$. We must establish that $\mathfrak{T}_4$ is a resolution tree. We can ignore the trivial case where $\mathfrak{T}_3$ and $\mathfrak{T}_4$ have just one node. So let Φ be any leaf of $\mathfrak{T}_3$, let Ω be the node immediately below Φ, and let Ψ be the other node immediately above Ω. (Ψ may or may not be a leaf of $\mathfrak{T}_3$). There are clauses δ and ϵ and an atom p such that $\hat{c}\Phi = \{p\} \dot{\cup} \delta$ and $\hat{c}\Psi = \{\sim p\} \dot{\cup} \epsilon$ and $\hat{c}\Omega = \delta \cup \epsilon$. There is a ground substitution $\hat{A}$ such that $\hat{A}\hat{c}\Phi' = \hat{c}\Phi = \{p\} \dot{\cup} \delta$, so $\hat{c}\Omega$ is a resolvent of $\hat{c}\Phi'$ and $\hat{c}\Psi$. (Take $\hat{B}$ as the trivial substitution in the definition of resolvent.) If Ψ is a leaf of $\mathfrak{T}_3$ then there is a ground substitution $\hat{B}$ such that $\hat{B}\hat{c}\Psi' = \hat{c}\Psi = \{\sim p\} \dot{\cup} \epsilon$, so $\hat{c}\Omega$ is a resolvent of $\hat{c}\Phi'$ and $\hat{c}\Psi'$. In either case we see that $\mathfrak{T}_4$ is a resolution tree. If a leaf Λ of $\mathfrak{T}_3$ has its clause in K_2, then $\hat{c}\Lambda' \in K$, so $\mathfrak{T}_4$ has K-support. Since a merge of $\mathfrak{T}_3$ is still a merge in $\mathfrak{T}_4$, $\mathfrak{T}_4$ represents a refutation of S with K-support by resolution with merging.

§4 Remarks.

In §1 we introduced a rather general definition of resolvent, since we permitted the substitutions $\hat{A}$ and $\hat{B}$ to be chosen arbitrarily. We should now like to point out that while this generality conveniently simplifies our theoretical discussion, it is superfluous in practice, since the substitutions need only be chosen to accomplish certain limited and specific purposes.

If α and β are clauses to be resolved, we first apply substitutions $\hat{X}$ and $\hat{Y}$ (the x- and y-standardizations of [2]) to obtain alphabetic variants $\hat{X}\alpha$ and $\hat{Y}\beta$ of α and β, respectively, which have no variables in common. This assures that occurrences of variables in the resolvent will not be occurrences of the same variable without good reason, and permits us to define just one further substitution $\hat{Z}$ to obtain $\hat{A} = \hat{Z}\circ\hat{X}$ and $\hat{B} = \hat{Z}\circ\hat{Y}$. We choose $\hat{Z}$ in such a way as to identify certain atoms of $\hat{X}\alpha \cup \hat{Y}\beta$ with one another, namely, the atoms of the literals we wish to resolve upon, the atoms of the merge literals (if the resolvent is to be a merge), and perhaps other atoms of literals which we wish to factor (see [4]).

In [2] it was proved that there is a most general unifier of a unifiable set of well-formed expressions. We next prove a useful corollary and extension of this theorem.

Definition. Let α_i be a finite set of well-formed expressions for $1 \leq i \leq n$. If $\hat{W}$ is a substitution such that $\hat{W}\alpha_i$ is a singleton for each i, then $\hat{W}$ simultaneously unifies $\alpha_1, \ldots,$ and α_n, and we say that $\alpha_1, \ldots \alpha_n$ are simultaneously unifiable. If $\hat{W}$ simultaneously unifies $\alpha_1, \ldots,$ and α_n, and for any substitution $\hat{Z}$ which simultaneously unifies $\alpha_1, \ldots,$ and α_n there is a substitution $\hat{V}$ such that $\hat{W} = \hat{V} \circ \hat{Z}$, then $\hat{W}$ is a <u>most general simultaneous unifier</u> of $\alpha_1, \ldots,$ and α_n.

<u>Simultaneous Unification Theorem</u>.

Let $\alpha_1, \ldots,$ and α_n be finite sets of well-formed expressions which are simultaneously unifiable. Then there is a most general simultaneous unifier of $\alpha_1, \ldots,$ and α_n.

Proof:

We prove the theorem by induction on n. For $n = 1$ we use the Unification Theorem of [2].

Suppose $\alpha_1, \ldots, \alpha_n$, and α_{n+1} are simultaneously unifiable by $\hat{Z}$, and let $\hat{A}$ be a most general unifier for $\alpha_1, \ldots,$ and α_n. Since $\hat{Z}$ simultaneously unifies $\alpha_1, \ldots,$ and α_n, there is a substitution $\hat{V}$ such that $\hat{Z} = \hat{V} \circ \hat{A}$. Now $\hat{Z}\alpha_{n+1} = \hat{V}\hat{A}\alpha_{n+1}$ is a singleton, so $\hat{V}$ unifies $\hat{A}\alpha_{n+1}$. Hence by the Unification Theorem there is a most general unifier $\hat{B}$ of $\hat{A}\alpha_{n+1}$. We assert that $\hat{B} \circ \hat{A}$ is a most general simultaneous unifier for $\alpha_1, \ldots, \alpha_n$, and α_{n+1}.

If $i \leq n$, $\hat{A}\alpha_i$ is a singleton so $\hat{B}\hat{A}\alpha_i$ is also. But $\hat{B}\hat{A}\alpha_{n+1}$ is a singleton, so $\hat{B} \circ \hat{A}$ simultaneously unifies $\alpha_1, \ldots, \alpha_n$, and α_{n+1}. Now suppose $\hat{Y}$ is any simultaneous unifier of $\alpha_1, \ldots, \alpha_n$, and α_{n+1}. Then $\hat{Y}$ simultaneously unifies $\alpha_1, \ldots,$ and α_n, so there is a substitution $\hat{U}$ such that $\hat{Y} = \hat{U} \circ \hat{A}$. Since $\hat{Y}\alpha_{n+1} = \hat{U}\hat{A}\alpha_{n+1}$, $\hat{U}$ is a unifier of $\hat{A}\alpha_{n+1}$, so there is a substitution $\hat{X}$ such that $\hat{U} = \hat{X} \circ \hat{B}$. Hence $\hat{Y} = \hat{U} \circ \hat{A} = (\hat{X} \circ \hat{B}) \circ \hat{A} = \hat{X} \circ (\hat{B} \circ \hat{A})$, so the required substitution $\hat{X}$ exists.

It should be noted that if $\hat{W}_1$ and $\hat{W}_2$ are two most general simultaneous unifiers of $\alpha_1, \ldots,$ and α_n, then there are substitutions

$\hat{X}$ and $\hat{Y}$ such that $\hat{W}_1 = \hat{X} \circ \hat{W}_2$ and $\hat{W}_2 = \hat{Y} \circ \hat{W}_1$. Hence it is clear that if p is any well-formed expression, $\hat{W}_1 p$ and $\hat{W}_2 p$ are alphabetic variants of one another. Thus the most general simultaneous unifier of $\alpha_1, \ldots,$ and α_n is essentially unique.

Implicit in our proof of the theorem above is an algorithm for finding the most general simultaneous unifier of $\alpha_1, \ldots,$ and α_n, which involves treating each of the α_i in turn. As a consequence of the essential uniqueness of the most general simultaneous unifier, it is clear that it does not matter in what order we treat the α_i.

Returning to our discussion of the resolvent of clauses α and β, once we have decided which subsets of $\hat{X}\alpha \cup \hat{Y}\beta$ we wish to unify, we might as well take $\hat{Z}$ to be the most general simultaneous unifier of these subsets, since the effect of using any other substitution which simultaneously unifies these subsets could be obtained by applying an additional substitution later. Thus even with our liberalized definition of resolvent, there are essentially only finitely many resolvents of α and β, since $\hat{X}\alpha \cup \hat{Y}\beta$ has only finitely many subsets.

In forming a resolvent of α and β, one might as well first unify the atoms of the literals in $\hat{X}\alpha \cup \hat{Y}\beta$ which are to be resolved upon, since we know we can treat the unifiable subsets in any order we please. One thus obtains the resolvent of α and β as defined [2], and one can make additional substitutions to complete the merge (if the resolvent is to be a merge). It may sometimes happen that the merge can be made in several different ways (but not in all of these ways at once), but there is no obvious criterion for deciding which is the best way of forming the merge. In such cases it seems reasonable to defer the actual merge until later, when it will occur as a factoring of the resolved clause as it is resolved with some other clause. Thus a refutation by resolution with merging may look exactly like a refutation as defined in [2]. However, the choice of resolvents is governed by the requirement that certain of the resolvents must be at least potential merges in accordance with the definition of a deduction by resolution with merging.

In general, when one is concerned with the problem of searching for refutations, it is advantageous to make the definition of a deduction as restrictive as possible (without loss of completeteness, and without rendering deductions unnecessarily long or awkward), so as to minimize the choices which must be made in the search for a refutation. There is an obvious way of making the definition of a deduction by resolution with merging with K-support more restrictive; namely, modify part (ii) of the definition to require that both $\hat{p}_1$ be $\hat{m}$ and $\hat{r}_1$ be $\hat{s}$, or that both $\hat{p}_2$ be $\hat{m}$ and $\hat{r}_2$ be $\hat{s}$. However, this definition is actually too restrictive, as the following example shows. Let S contain the following clauses:

(1)	$\{\sim p, \sim q\}$	(2)	$\{p, r\}$	(3)	$\{q, r\}$
(4)	$\{\sim s, \sim t\}$	(5)	$\{s, \sim r\}$	(6)	$\{t, \sim r\}$.

Let K contain just clauses (1) and (4). Then it is easy to check that S has a refutation by resolution with merging with K-support under our present definition, but not under the modified definition mentioned above.

Bibliography

[1] Peter B. Andrews, "On Simplifying the Matrix of a Wff," to appear in the Journal of Symbolic Logic.

[2] J. A. Robinson, "A Machine-Oriented Logic Based on the Resolution Principle." Journal of the Association for Computing Machinery, vol. 12 (1965), pp. 23-41.

[3] J. A. Robinson, "Automatic Deduction with Hyper-Resolution," International Journal of Computer Mathematics, vol. I (1965), pp. 227-234.

[4] Lawrence Wos, Daniel F. Carson, and George A. Robinson, "The Unit Preference Strategy in Theorem Proving," AFIPS Conference Proceedings 26, Spartan Books, Washington, D.C., 1964, pp. 615-621.

[5] Lawrence Wos, George A. Robinson, and Daniel F. Carson, "Efficiency and Completeness of the Set of Support Strategy in Theorem Proving," Journal of the Association for Computing Machinery, vol. 12 (1965), pp. 536-541.