Notre Dame Journal of Formal Logic
Volume XIX, Number 1, January 1978
NDJFAM

# AN AUTOMATIC THEOREM PROVER FOR SUBSTITUTION AND DETACHMENT SYSTEMS

JEREMY GEORGE PETERSON

1 As mentioned in [3], the proofs exhibited in that paper were found with the aid of a theorem-proving computer program. This paper is a short summary of the algorithms and heuristics used.

By a substitution and detachment system we mean any formal system whose language contains denumerably many variables and a finite set of connectives, including a distinguished binary connective which we call $C$. The theorems are the well-formed formulae which may be derived from a given set of well-formed formulae, called the axioms, by the rules of *modus ponens* (with respect to $C$) and substitution. We will assume all formulae to be in Polish prefix notation. It is known (*cf.* [7], p. 4) that Meredith's condensed detachment operator **D** provides an efficient method of presenting the proof of a theorem in such a system, but it does more than this. The result of applying *modus ponens* (with some appropriate substitutions) to the formulae $C\alpha\beta$ and $\gamma$ may be an infinite set of substitution instances of $\beta$. Some restriction must be placed on this set if *modus ponens* is to be mechanised. In using condensed detachment we choose the one member of this set, namely $\mathbf{D}C\alpha\beta.\gamma$, which is most general in the sense that every other member of the set is a substitution instance of it [7], p. 4, as our result.

J. Kalman has proved that every theorem which can be derived by *modus ponens* and substitution from a set of axioms is a substitution instance of a theorem which may be derived by condensed detachment alone. Thus we may use a theorem prover whose only rule of inference is condensed detachment to prove any theorem in a substitution and detachment system, if we consider that at each step not only the theorem derived but also all substitution instances of it have been proved. In practice it is rarely necessary to consider the substitution instances since theorems are usually required in the strongest possible form. That we can construct such a theorem prover follows from the fact that an algorithm is known for condensed detachment [1].

*Received July 1, 1976*

2 In describing the basic theorem prover two algorithms will be given. The first generates $D\delta.\gamma$ for two given formulae $\delta$ and $\gamma$ or shows that it does not exist, and the second generates an exhaustive "search" of the set of all formulae derivable from a given set of axioms using condensed detachment as the sole rule of inference. The following data structure is assumed: the formulae are represented by strings of non-zero integers stored left-justified in array rows such that the variables are represented by positive integers, $C$ is represented by -1, and the other connectives are represented by -2, -3 etc. The algorithm for condensed detachment due to J. Kalman [1] operates as follows. To generate $DC\alpha\beta.\gamma$, Robinson's unification algorithm [5], p. 32 is used to unify $\gamma$ and $\alpha$ and the most general unifier found is applied to $\beta$ to yield $DC\alpha\beta.\gamma$. The unification fails exactly when $DC\alpha\beta.\gamma$ does not exist. More precisely, let $\alpha$, $\beta$, $\gamma$ be stored in arrays A, B, E then the following algorithm generates $DC\alpha\beta.\gamma$ or shows that it does not exist:

1. Renumber the variables of $\gamma$ so that $C\alpha\beta$ and $\gamma$ have no variables in common. I: = 0.
2. Do I: = I + 1 until A[I] ≠ E[I] or E[I] = 0.
3. If E[I] = 0 then STOP. The content of B is $DC\alpha\beta.\gamma$. Otherwise continue.
4. If E[I] is a variable then H: = E[I] and place the subformula of A beginning with A[I] in the array F. Go to 7. Otherwise continue.
5. If A[I] is a variable then H: = A[I] and place the subformula of E beginning with E[I] in the array F. Go to 7. Otherwise continue.
6. STOP. $DC\alpha\beta.\gamma$ does not exist.
7. If H occurs in F then STOP. $DC\alpha\beta.\gamma$ does not exist. Otherwise continue.
8. Substitute the content of F for H throughout A and B and E. Go to 2.

The basic search strategy known as the level saturation or breadth-first method operates by listing the axioms then, starting at the top of the list, using each theorem to perform condensed detachment with all the theorems occurring above it on the list. If condensed detachment does not fail it is convenient at this stage to test whether the theorem has been previously generated before adding it to the end of the list. If the list of axioms is in the first $n$ rows of the array THEOREM[L,I], then the following algorithm performs the breadth-first search:

1. I1: = 1, L: = $n$ + 1.
2. I2: = 1. If I1 = L then STOP. All possible detachments have been performed.
3. Form DI1.I2. If a new theorem is formed (one not previously generated) then place it in row L of THEOREM, L: = L + 1.
4. If I1 = I2 then I1: = I1 + 1. Go to 2. Otherwise continue.
5. Form DI2.I1. If a new theorem is produced then place it in row L of THEOREM, L: = L + 1.
6. I2: = I2 + 1. Go to 3.

In practice this algorithm requires a bound on the length of the theorems to be stored related to the dimension of the array THEOREM. This restriction is called the length heuristic by some authors (e.g., [6]).

3 The theorem prover is coded in algol and run on the Auckland University B6700 computer. It is about 245 lines long. In providing the derivations in [3] the program had at its disposal about 64,000 words of storage and a maximum of 2,700 seconds of processing time. It was not required of the theorem prover that it find a deduction completely to the required result but only that enough material be generated to enable a simple hand calculation to complete the solution.

The breadth-first technique applied to axiom (1) of [3] yielded 600 theorems in 168 seconds including axioms (2), (3), (5), and (10). This technique applied to (2) and to (3) again yielded 600 theorems very quickly and derivations to other axioms could be easily constructed. Except for these and (4) and (5) which are closely linked ([2], p. 185), none of the 10 axioms yielded enough information in a breadth-first search for a derivation to another axiom to be constructed. Thus for the purposes of [3] some heuristic considerations were necessary.

The best heuristic found in this work involves the maximum length of the stored theorems. It is found that, as the search continues and more theorems are generated, the maximum length of the theorems which are accepted for storage by the program may be reduced without the program exhausting the pairs of theorems available to it for generation and terminating. For example when the breadth-first method is applied to (9) with theorems of length at most 19 stored, the program terminates after generating only two theorems. With the storage length increased to 23 the breadth-first method generates 600 theorems in 600 seconds but no derivation may be easily constructed to another axiom. However if, after 50 theorems have been generated, only theorems of length at most 15 are stored, 600 theorems are generated in 412 seconds, but more importantly 120 of the first 600 theorems have only 11 primitive symbols compared with 25 of the 600 with a constant maximum of 23, a considerable advantage for subsequent hand calculation. The heuristic which proved most useful for the work in [3] accepted theorems of maximum length 27 for the first 50 theorems and theorems of maximum length 15 thereafter. It also gave preference to theorems of length 11 for one of the parents. Other combinations of bounds for the length of stored theorems have been successfully used in the search spaces generated by other axiom sets.

A fuller account of this work appears in [4], section 6, p. 23 including a formalization of the search space and an account of the relative efficiency of the heuristic outlined above.

## REFERENCES

[1] Kalman, J. A., "An algorithm for Meredith's condensed detachment," (abstract), *The Journal of Symbolic Logic*, vol. 39 (1974), p. 206.

[2] Meredith, C. A., and A. N. Prior, "Axiomatics of the propositional calculus," *Notre Dame Journal of Formal Logic*, vol. IV (1963), pp. 171-187.

[3] Peterson, J. G., "Shortest single axioms for the classical equivalential calculus," *Notre Dame Journal of Formal Logic*, vol. XVII (1976), pp. 267-271.

[4] Peterson, J. G., "Single axioms for the classical equivalential calculus," *Auckland University Department of Mathematics Report Series*, No. 78 (1975).

[5] Robinson, J. A., "A machine-oriented logic based on the resolution principle," *Journal of the Association for Computing Machinery*, vol. 12 (1965), pp. 23-41.

[6] Siklóssy, L., A. Rich, and V. Marinov, "Breadth-first search: some surprising results," *Artificial Intelligence*, vol. 4 (1973), pp. 1-27.

[7] Zeman, J. J., *Modal Logic*, Clarendon Press, Oxford (1973).

*University of Auckland*
*Auckland, New Zealand*