20.59HRS. 19 NOV 1973.

[PRINTER]

DISCUSER 22 AT 20.59 ON [19 NOV 1973]

READY 50; 0 UNTIDY; DISCEND 110

[/ GENERAL1]
[/ FERTILIZ]
[/ REWRITE]
[/]
[/ REDUCE]
[/ INDUCT1]
[/ INDUCT2]

99	11	11.57	2 11 1973
86	13	11.56	2 11 1973
71	15	16.35	1 11 1973
69	2	16.34	1 11 1973
61	8	16.34	1 11 1973
38	23	16.33	1 11 1973
10	28	16.32	1 11 1973

[20.55 19 NOV 1973]

[/ INDUCT2]
[/ INDUCT1]
[/ REDUCE]
[/]
[/ REWRITE]
[/ FERTILIZ]
[/ GENERAL1]

[20.59 19 NOV 1973]

DUMP TITLE IS: [BIN DISC DUMP 22 20.55 19 NOV 1973]

```
[/ INDUCT2]                          [ 20.59 19 NOV 1973]
DTRACK 22    CREATED   16.32    1 11 1973

 1   COMMENT'THIS IS THE FILE WHICH CONSTRUCTS THE INDUCTION
 2   FORMULA AND LINKS THE INDUCTION PKG WITH THE REST OF THE
 3   THEOREM PROVER.`;
 4
 5
 6
 7   COMMENT'THE FOLLOWING FUNCTION IS USED TO PROCESS THE POCKET LIST
 8   AND FAILURES LIST RETURNED BY PICKINDCONST, BEFORE THE
 9   INDUCTION FORMULA IS ACTUALLY CONSTRUCTED.THIS FUNCTION
10   CREATES ALISTS OF THE FORM:
11     (INDUCTION CONSTANT . LIST OF DESTRUCTORS APPLIED).
12   IT RETURNS TWO SUCH ALISTS, ONE CORRESPONDING TO JUST
13   THE RECURSIVE DESTRUCTIONS, AND THE OTHER TO BOTH
14   RECURSIVE AND NON-RECURSIVE ONES.`;
15
16   FUNCTION GENDRALISTS RECPOCKETS FAILURES;
17   VARS F X Y DESTALIST L1 L2;
18   NIL->DESTALIST;
19   LAMBDA L1;
20   APPLIST(L1,
21           LAMBDA TERM;
22           GETARG(TERM)->X;
23           IF ASSOC(X,DESTALIST)
24              THEN
25              ->Y;
26              IF MEMBEREQUAL(TERM,BACK(Y))
27                 THEN ELSE TERM::BACK(Y)->BACK(Y);CLOSE;
28              ELSE CONSPAIR(X,[%TERM%])::DESTALIST->DESTALIST;CLOSE;
29           END);
30   END->F;
31   APPLIST(RECPOCKETS,F);
32   COPYLIST(DESTALIST);
33   APPLIST(FAILURES::NIL,F);
34   DESTALIST;
35   END;
36   COMMENT'(USE OF MEMBEREQUAL RATHER THAN MEMBERID IS
37   OK HERE SINCE TERM IS JUST A COLLECTION OF CARS AND CDRS
38   APPLIED TO A SKOCONSTS, AND HENCE IS IDENT IFF EQUAL.)`;
39
40
41   COMMENT'IMPORTANT NOTE:  THIS INDUCTION ROUTINE KNOWS ABOUT
42   NUMBERS.  THE FOLLOWING IS ASSUMED:  THE EVAL ROUTINE
43   KNOWS THAT THE CAR OF A NUMERIC SKOLEM CONSTANT IS NIL.
44   THIS GUARANTEES THAT NO "CAR" TERMS WILL OCCUR IN THE
45   LIST OF DESTRUCTORS OF A NUMERIC SKOLEM CONSTANT TO BE
46   INDUCTED UPON.`;
47   COMMENT'THE FOLLOWING ROUTINES KNOW ABOUT NUMBERS:
48   STEPFOR (GENERATES A NEST OF ADD1S AS DEEP AS THE
49   DEEPEST CDR NEST AROUND A NUMERIC SKO CONST),
50   BASESFOR (GENERATES A LIST OF THE NUMBERS
51   BETWEEN 0 AND N INSTEAD OF THE CORRESPONDING
52   CONSES, AND GETCOMP (WHICH IS USED RATHER THAN
53   EVAL BECAUSE OF THE PRESENCE OF "ADD1"S IN THE
54   CONCLUSION).  AS OF THIS WRITING, NO OTHER ROUTINES
55   ARE AFFECTED (INFECTED).`;
56
```

```
    57
    58  COMMENT'NOW ON TO INDUCTION.  THE FIRST SET OF FUNCTIONS
    59  IS CONCERNED WITH GROWING THE LEAST STRUCTURED TERM
    60  ALLOWING EACH DESTRUCTOR COMBINATION TO FULLY OPERATE ON IT.
    61  FOLLOWING THIS IS A SET OF FUNCTIONS WHICH CONSTRUCT ALL OF
    62  THE BASES THAT MUST BE ALLOWED, GIVEN THE TERM GROWN ABOVE.';
    63
    64
    65  FUNCTION GROW TERM;
    66  VARS Y;
    67  IF ATOM(TERM) THEN MUNG;EXIT;
    68  GROW(HD(TL(TERM)))->Y;
    69  IF HD(TERM)="CAR"
    70     THEN
    71     IF HD(Y)="CONS"
    72        THEN IF ATOM(HD(TL(Y))) THEN TL(Y); ELSE HD(TL(Y));CLOSE;
    73        ELSE
    74        [%"CONS",GENSKO(CONST),HD(Y)%]->HD(Y);
    75        TL(HD(Y));
    76        CLOSE;
    77     ELSE
    78     IF HD(Y)="CONS"
    79        THEN IF ATOM(HD(TL(TL(Y)))) THEN TL(TL(Y)); ELSE HD(TL(TL(Y)));CLO
SE;
    80        ELSE
    81        [%"CONS",GENSKO(CONST),HD(Y)%]->HD(Y);
    82        TL(TL(HD(Y)));
    83        CLOSE;
    84     CLOSE;
    85  END;
    86
    87  COMMENT'THIS FUNCTION TAKES A SKOLEM CONSTANT AND A LIST
    88  OF DESTRUCTORS APPLIED TO IT, AND CONSTRUCTS THE LEAST
    89  STRUCTURED TERM ALLOWING EACH DESTRUCTOR TO OPERATE.';
    90
    91  [%CONSPAIR("CDR","ADD1")%]->CDRTOADD1;
    92
    93  FUNCTION STEPFOR CONST TERMLIST;
    94  VARS TERM X;
    95  IF NUMSKO(CONST)
    96     THEN
    97     HD(TERMLIST)->TERM;
    98     CONSCNT(TERM)->X;
    99     LOOPIF (TL(TERMLIST)->TERMLIST;TERMLIST/=NIL)
   100        THEN
   101        IF CONSCNT(HD(TERMLIST))>X
   102           THEN HD(TERMLIST)->TERM;CONSCNT(TERM)->X;CLOSE;
   103        CLOSE;
   104     APPSUBST(CDRTOADD1,TERM);
   105     EXIT;
   106  [%"CONS",GENSKO(CONST),CONST%]->SEED;
   107  LOOPIF TERMLIST/=NIL
   108     THEN
   109     SEED->MUNG;
   110     ERASE(GROW(HD(TERMLIST)));
   111     TL(TERMLIST)->TERMLIST;
   112     CLOSE;
   113  SEED;
   114  END;
   115
```

```
116   COMMENT'THIS FUNCTION RETURNS A LIST OF ALL THOSE
117   TERMS "LESS THAN" THE GIVEN TERM, BY REPLACING
118   ALL POSSIBLE COMBINATIONS OF SUB-CONSES BY NILS.
119   IT IS USED BY BASESFOR TO CONSTRUCT THE BASES FOR
120   A GIVEN CONSTANT TO BE INDUCTED UPON.';
121
122
123   FUNCTION SMALLER TERM;
124   IF ATOM(TERM) THEN NIL;EXIT;
125   NIL::TL([%APPLIST(HD(TL(TERM))::SMALLER(HD(TL(TERM))),
126                     LAMBDA ARG1;
127                     APPLIST(HD(TL(TL(TERM)))::SMALLER(HD(TL(TL(TERM)))),
128                             LAMBDA ARG2;
129                             [%"CONS",ARG1,ARG2%];
130                             END);
131                     END)%]);
132   END;
133
134
135
136   FUNCTION BASESFOR CONST TERM;
137   IF NUMSKO(CONST)
138      THEN
139      0->FOO1;
140      [% LOOPIF ISLINK(TERM)
141            THEN FOO1;FOO1+1->FOO1;HD(TL(TERM))->TERM;CLOSE%];
142      EXIT;
143   SMALLER(TERM);
144   END;
145
146
147   COMMENT'THE FUNCTION BELOW CONJOINS A LIST OF THINGS';
148
149   FUNCTION CONJOIN L;
150   IF TL(L)=NIL THEN HD(L);
151      ELSE [%"AND",HD(L),CONJOIN(TL(L))%];CLOSE;
152   END;
153
154
155   COMMENT'"GETCOMP" BEHAVES JUST LIKE EVAL, FOR A NEST OF
156   CARS AND CDRS APPLIED TO A SKOLEM CONSTANT, WHERE THE
157   CONSTANT IS BOUND ON AN ALIST CALLED THE STEPALIST.
158   IT IS USED TO DETERMINE THE SUBSTRUCTURE OF THE "STEP"
159   FOR WHICH A HYPOTHESIS WILL BE SUPPLIED.  THE REASON
160   EVAL IS NOT USED IS THAT, FOR I/O PURPOSES, SOME
161   STEPS MIGHT BE WITH ADD1 TERMS RATHER THAN [CONS NIL ..]
162   AND THEY WOULD HAVE TO BE FULLY EVALD FIRST.';
163
164   FUNCTION GETCOMP TERM;
165   IF ATOM(TERM)
166      THEN TERM;BACK(ERASE(ASSOC(TERM,STEPALIST)));EXIT;
167   GETCOMP(HD(TL(TERM)))->FOO1;
168   IF HD(TERM)="CAR" OR HD(FOO1)="ADD1"
169      THEN HD(TL(FOO1));
170      ELSE HD(TL(TL(FOO1)));CLOSE;
171   END;
172
173
174
175   COMMENT'THIS IS THE FUNCTION WHICH CONSTRUCTS THE INDUCTION
```

```
176   FORMULA.  FIRST IT SETS UP THE STEPALIST, A LIST OF THE
177   THINGS INDUCTED UPON, PAIRED WITH THE TERM TO REPLACE THEM
178   IN THE CONCLUSION.  THIS TERM IS THE LEAST STRUCTURED TERM
179   WHICH ALLOWS ALL THE DESTRUCTORS TO FULLY OPERATE ON IT.
180   THEN IT SETS UP THE HYPALISTLIST, WHICH IS A LIST OF ALISTS;
181   EACH ALIST PAIRS A CONST TO BE INDUCTED UPON WITH WHAT IT
182   IS TO BE REPLACED BY IN THE HYPOTHESIS.  THIS IS GENERATED
183   BY APPLYING THE RECURSIVE DESTRUCTORS TO THE LEAST
184   STRUCTURED TERM DESCRIBED ABOVE.  THERE IS SUCH AN ALIST FOR
185   EACH RECURSIVE POCKET.`;
186   COMMENT`NEXT, IT SETS UP THE BASES LIST,
187   WHICH IS THE LIST OF ALL THE BASES TO BE ESTABLISHED.  THESE
188   ARE JUST THE THEOREM INSTANTIATED TO ALL THE TERMS
189   SMALLER THAN THE ONE IN THE CONCLUSION, FOR EACH INDUCTION CONST.
190   FINALLY, IT SETS UP THE HYPLIST, WHICH IS A LIST OF
191   ALL THE HYPOTHESES, ONE FOR EACH ALIST ON THE HYPALISTLIST;`;
192
193   COMMENT`ONCE ALL THIS IS DONE, IT CONSTRUCTS THE FORMULA IN
194   THE OBVIOUS WAY.`;
195
196   FUNCTION INDFORMULA RECPOCKETS DESTALIST INDTERM;
197   VARS ALIST;
198   [%APPLIST(DESTALIST,
199             LAMBDA X;
200             CONSPAIR(FRONT(X),STEPFOR(FRONT(X),BACK(X)));END)%];
201   ->STEPALIST;
202
203   [%APPLIST(RECPOCKETS,
204             LAMBDA POCKET;
205             [%APPLIST(POCKET,
206                       LAMBDA TERM;
207                       CONSPAIR(GETCOMP(TERM));
208                       END)%]
209             END)%]
210   ->HYPALISTLIST;
211
212   [%APPLIST(STEPALIST,
213             LAMBDA X;
214             FRONT(X)->CONST;
215             APPLIST(BASESFOR(CONST,BACK(X)),
216                     LAMBDA TERM;SUBST(TERM,CONST,INDTERM);END);
217             END)%]
218   ->BASES;
219
220   [%APPLIST(HYPALISTLIST,
221             LAMBDA ALIST;
222             APPSUBST(ALIST,INDTERM);
223             END)%]
224   ->HYPLIST;
225
226   [%"AND",CONJOIN(BASES),
227     [%"IMPLIES",CONJOIN(HYPLIST),
228       APPSUBST(STEPALIST,INDTERM)%]%];
229   END;
230
231   FUNCTION INDREPORT;
232   IF VERBOSE
233      THEN
234      POPTTON();
235      PRSEQAND(4,'INDUCT ON `,INDCONSTS,PR);
```

```
236       CLOSE;
237    END;
238
239    FUNCTION INDUCT INDTERM;
240    IF NOT(PICKINDCONSTS(INDTERM)) THEN 0;EXIT;
241    ->INDCONSTS;
242    ->RECPOCKETS;
243    ->OTHERFAILS;
244    GENDRALISTS(RECPOCKETS,OTHERFAILS)->DESTALIST->RECALIST;
245    INDCONSTS<>INDLIST->INDLIST;
246    INDFORMULA(RECPOCKETS,DESTALIST,INDTERM);
247    REPORT("I"::INDCONSTS,INDREPORT,"INDUCT");
248    1;
249    END;
250
251
```

```
 1
 2   COMMENT'THIS FILE CONTAINS THE FUNCTIONS WHICH CHOOSE WHICH
 3   CONSTANTS TO INDUCT UPON.';
 4   NIL->INDLIST;
 5
 6   FUNCTION INDUCTABLE TERM;
 7   VARS X;
 8   LOOPIF ISLINK(TERM)
 9      THEN
10      HD(TERM)->X;
11      IF X/="CDR" AND X/="CAR"
12         THEN 0;EXIT;
13      HD(TL(TERM))->TERM;
14      CLOSE;
15   TERM;
16   1;
17   END;
18
19   FUNCTION GETARG TERM;
20   LOOPIF ISLINK(TERM) THEN HD(TL(TERM))->TERM;CLOSE;
21   TERM;
22   END;
23
24   COMMENT'THE FOLLOWING THREE FUNCTIONS ARE USED TO SWEEP
25   THROUGH THE EXPANDED FUNCTION DEFNS SET UP BY SYMEVAL
26   AND COLLECT INFORMATION ON HOW THEY BOMBED.  IN PARTICULAR,
27   FAULT DESCRIPTIONS ARE BUILT, WHICH ARE LISTS CONTAINING TWO
28   SUBLISTS:  THE FIRST IS A LIST OF THE DESTRUCTORS APPLIED
29   IN THE RECURSIVE CALLS, AND THE SECOND IS A LIST OF ALL
30   OTHER DESTRUCTORS APPLIED.';
31
32
33   FUNCTION INDUCTSWEEP;
34   [%APPLIST(TOPLEXP,
35             LAMBDA X;
36             VARS BOMBLIST OTHERFAILS HDTERM;
37             NIL->BOMBLIST;
38             NIL->OTHERFAILS;
39             HD(HD(X))->HDTERM;
40             INDSW1(HD(TL(X)));
41             IF BOMBLIST/=NIL THEN [%BOMBLIST,OTHERFAILS%];CLOSE;
42             END)%];
43   END;
44
45   FUNCTION INDSW1 TERM1;
46   IF ATOM(TERM1) THEN EXIT;
47   IF ISINTER(TERM1)
48      THEN
49      TERM1::OTHERFAILS->OTHERFAILS;
50   ELSEIF HD(TERM1)=HDTERM
51      THEN
52      [%APPLIST(TL(TERM1),LAMBDA TERM2; IF ISINTER(TERM2) THEN TERM2;
53                                              CLOSE;END)%]->FOO1;
54      IF FOO1/=NIL THEN FOO1::BOMBLIST->BOMBLIST;CLOSE
55      EXIT;
56   APPLIST(TL(TERM1),INDSW1);
```

```
  57   END;
  58
  59   FUNCTION ISINTER TERM;
  60   IF ATOM(TERM) THEN 0;EXIT;
  61   HD(TERM)->TERM;
  62   IF TERM="CDR"
  63      THEN 1; ELSE TERM="CAR";CLOSE;
  64   END;
  65   COMMENT'THE FOLLOWING FUNCTION TRANSFORMS FAULT DESCRIPTIONS
  66   INTO FOUR TUPLES TO MAKE IT EASIER TO SORT THROUGH THEM
  67   TO FIND WHAT TO INDUCT UPON.  IT THROWS OUT ANY REQUIRING
  68   INDUCTION ON NON SKOLEM CONSTANTS.';
  69
  70   FUNCTION TRANSFAULT FAULTDESC;
  71   VARS ARGLIST X;
  72   NIL->ARGLIST;
  73   XAPPLIST(HD(FAULTDESC),
  74            LAMBDA POCKET;
  75            XAPPLIST(POCKET,
  76                     LAMBDA TERM;
  77                     IF INDUCTABLE(TERM) THEN
  78                        ->X;
  79                        IF MEMBER(X,ARGLIST) THEN ELSE X::ARGLIST->ARGLIST;
CLOSE;
  80                        ELSE 1->XAPPFLAG;CLOSE;
  81                     END);
  82            END);
  83   IF XAPPFLAG THEN EXIT;
  84   [%1,ARGLIST,HD(FAULTDESC),
  85     [%APPLIST(HD(TL(FAULTDESC)),LAMBDA TERM;
  86                                 IF INDUCTABLE(TERM) AND MEMBER((),ARGLIST)
 THEN TERM;CLOSE;
  87                                 END)%]%];
  88   END;
  89
  90   COMMENT'(THE FIRST COMPONENT ABOVE WILL BE USED TO SCORE
  91   THE CANDIDATES)';
  92
  93   FUNCTION GETCANDS FAULTLIST;
  94   [%APPLIST(FAULTLIST,TRANSFAULT)%];
  95   END;
  96
  97   FUNCTION MERGECANDS CANDLIST;
  98   VARS CAND1;
  99   CANDLIST;
 100   LOOPIF TL(CANDLIST)/=NIL
 101      THEN
 102      HD(CANDLIST)->CAND1;
 103      TL(CANDLIST)->CANDLIST;
 104      XAPPLIST(CANDLIST,
 105               LAMBDA CAND2;
 106               IF INTSECTP(HD(TL(CAND1)),HD(TL(CAND2)),NONOP=)
 107                  THEN
 108                  1->XAPPFLAG;
 109                  UNION(HD(TL(CAND1)),HD(TL(CAND2)),NONOP=)->HD(TL(CAND2));

 110                  UNION(HD(TL(TL(CAND1))),HD(TL(TL(CAND2))),EQUAL)->HD(TL(T
L(CAND2)));
 111                  UNION(HD(TL(TL(TL(CAND1)))),HD(TL(TL(TL(CAND2)))),EQUAL)-
>
```

```
112                     HD(TL(TL(TL(CAND2))));
113                     HD(CAND2)+HD(CAND1)->HD(CAND2);
114                     0->HD(CAND1);
115                     CLOSE;
116                 END);
117     CLOSE;
118 END;
119
120
121 FUNCTION CHOOSEHIGH CANDLIST;
122 VARS HIGH ANS;
123 -10000->HIGH;
124 IF TL(CANDLIST)=NIL THEN CANDLIST;EXIT;
125 LOOPIF CANDLIST/=NIL
126     THEN
127     IF HD(HD(CANDLIST))>HIGH AND HD(HD(CANDLIST))
128         THEN
129         HD(HD(CANDLIST))->HIGH;
130         HD(CANDLIST)::NIL->ANS;
131     ELSEIF HD(HD(CANDLIST))=HIGH
132         THEN
133         HD(CANDLIST)::ANS->ANS;
134         CLOSE;
135     TL(CANDLIST)->CANDLIST;
136     CLOSE;
137 ANS;
138 END;
139
140 FUNCTION CHOOSENEW CANDLIST;
141 APPLIST(CANDLIST,
142         LAMBDA CAND;
143         1->HD(CAND);
144         APPLIST(HD(TL(CAND)),
145                 LAMBDA TERM;
146                 IF NOT(MEMBER(TERM,INDLIST))
147                     THEN 1+HD(CAND)->HD(CAND);
148                     CLOSE;
149                 END);
150         END);
151 CHOOSEHIGH(CANDLIST);
152 END;
153
154
155
156 COMMENT'THE FUNCTION BELOW MERGES ALL RECURSIVE POCKETS WHICH
157 HAVE NON-NIL INTERSECTIONS.';
158
159 FUNCTION MERGEPOCKETS POCKETLIST;
160 IF POCKETLIST=NIL THEN NIL;
161     ELSE ADDPOCKET(HD(POCKETLIST),MERGEPOCKETS(TL(POCKETLIST)));CLOSE;
162 END;
163
164 FUNCTION ADDPOCKET POC POCLIST;
165 IF POCLIST=NIL THEN [%POC%];
166 ELSEIF INTSECTP(POC,HD(POCLIST),EQUAL)
167     THEN UNION(POC,HD(POCLIST),EQUAL)::TL(POCLIST);
168     ELSE HD(POCLIST)::ADDPOCKET(POC,TL(POCLIST));CLOSE;
169 END;
170
171
```

```
172
173  COMMENT'A POCKET IS SUBSUMED BY ANOTHER IF ALL OF ITS
174  TERMS OCCUR AS SUBTERMS IN ANY TERM IN THE OTHER.';
175
176  FUNCTION SUBSUMED POCKET1 POCKET2;
177  VARS TERM1;
178  LOOPIF POCKET1/=NIL
179     THEN
180     HD(POCKET1)->TERM1;
181     IF (XAPPLIST(POCKET2,
182                  LAMBDA TERM2;OCCUR(TERM1,TERM2)->XAPPFLAG;END);XAPPFL

183        THEN; ELSE 0;EXIT;
184     TL(POCKET1)->POCKET1;
185     CLOSE;
186  1;
187  END;
188
189  COMMENT'THIS FUNCTION TRANSFORMS A LIST OF POCKETS INTO
190  A LIST OF POCKETS THAT IS SUBSUMPTION FREE.';
191
192  FUNCTION SUBSUME POCLIST;
193  [%APPLIST(POCLIST,
194            LAMBDA POCKET1;
195            XAPPLIST(POCLIST,
196                     LAMBDA POCKET2;
197                     IF POCKET1=POCKET2 THEN EXIT;
198                     SUBSUMED(POCKET1,POCKET2)->XAPPFLAG;
199                     END);
200            IF XAPPFLAG THEN ELSE POCKET1;CLOSE;
201            END)%];
202  END;
203
204
205  COMMENT'THE FOLLOWING SUBSTITUTION IS USED TO REPLACE
206  CAR, AND CDRS OCCURING EXPLICITLY IN THE
207  THEOREM BY DUMMY SYMBOLS TO AVOID CONFUSING THEM WITH
208  RECURSIVE ONES IN THE EXPANDED FN DEFNS.';
209
210  [%CONSPAIR("CDR","DUMMYCDR"),
211    CONSPAIR("CAR","DUMMYCAR")%]
212  ->DUMMYSUBST;
213
214  FUNCTION PICKINDCONSTS INDTERM;
215  VARS CANDLIST;
216  1->ININDUCT;
217  ERASE(SYMEVAL(APPSUBST(DUMMYSUBST,INDTERM)));
218  0->ININDUCT;
219  GETCANDS(INDUCTSWEEP())->CANDLIST;
220  IF CANDLIST=NIL THEN 0;EXIT;
221  MERGECANDS(CANDLIST)->CANDLIST;
222  CHOOSEHIGH(CANDLIST)->CANDLIST;
223  IF TL(CANDLIST)/=NIL
224     THEN
225     CHOOSENEW(CANDLIST)->CANDLIST;
226     CLOSE;
227  HD(CANDLIST)->CANDLIST;
228  HD(TL(TL(TL(CANDLIST))));
229  SUBSUME(MERGEPOCKETS(HD(TL(TL(CANDLIST)))));
230  HD(TL(CANDLIST));
```

```
231     1;
232     END;
233
234
```

```
 1
 2   COMMENT'THIS IS THE REDUCE FUNCTION. IN-LINE COMMENTS EXPLAIN
 3   THE REWRITE RULES APPLIED.';
 4
 5
 6   VARS REDUCE;
 7
 8
 9   FUNCTION REDUCE1 TERM CONSLIST;
10   VARS TERM1 TERM2 TERM3;
11   RECURSE:
12   COMMENT'IF TERM IS ATOM OR NON-IF, QUIT';
13   IF ATOM(TERM) OR HD(TERM)/="IF"
14      THEN
15      TERM;
16      EXIT;
17
18   COMMENT'GET COMPONENTS OF THE IF';
19   HD(TL(TERM))->TERM1;
20   HD(TL(TL(TERM)))->TERM2;
21   HD(TL(TL(TL(TERM))))->TERM3;
22
23   COMMENT'IF TERM1 IS NIL OR CONS, EVAL IT';
24   IF TERM1==NIL
25      THEN
26      TERM3->TERM;
27      GOTO RECURSE;
28   ELSEIF EXPLCONS(TERM1) OR MEMBERID(TERM1,CONSLIST)
29      THEN
30      TERM2->TERM;
31      GOTO RECURSE;
32      CLOSE;
33
34   COMMENT'(IF ATOM A B) => (IF ATOM R(A(ATOM/CONS)) R(B(ATOM/NIL)))';
35   IF ATOM(TERM1)
36      THEN
37      GOTO SUBSTCONS;
38      CLOSE;
39
40   COMMENT'(IF (EQUAL A SPECLIST) B C) => (IF (EQUAL A SPECLIST)
41     R(B(A/SPECLIST))
42     R(C((EQUAL A SPECLIST)/NIL)))';
43   IF HD(TERM1)="EQUAL"
44      THEN
45      IF ISSPEC(HD(TL(TERM1)))
46         THEN SUBST(HD(TL(TERM1)),HD(TL(TL(TERM1))),TERM2)->TERM2;
47      ELSEIF ISSPEC(HD(TL(TL(TERM1))))
48         THEN SUBST(HD(TL(TL(TERM1))),HD(TL(TERM1)),TERM2)->TERM2;
49         ELSE GOTO SUBSTTRUE;CLOSE;
50      GOTO ASSEMBOOL;
51      CLOSE;
52
53   COMMENT'(IF (IF ...) A B) => (IF R(IF) R(A) R(B))';
54   IF HD(TERM1)="IF"
55      THEN
56      REDUCE1(TERM1,CONSLIST)->TERM1;
```

```
57       REDUCE1(TERM2,CONSLIST)->TERM2;
58       REDUCE1(TERM3,CONSLIST)->TERM3;
59       IF TERM3==NIL THEN GOTO CONTINUE;CLOSE;
60       [%"IF",TERM1,TERM2,TERM3%];
61       EXIT;
62
63    CONTINUE:
64
65    COMMENT'(IF BOOL A B) => (IF BOOL R(A(BOOL/T)) R(B(BOOL/NIL)))';
66    IF BOOLEAN(TERM1)
67       THEN
68       SUBSTTRUE:
69       SUBST(T,TERM1,TERM2)->TERM2;
70       ASSEMBOOL:
71       [%"IF",TERM1,
72         REDUCE1(TERM2,CONSLIST),
73         REDUCE1(SUBST(NIL,TERM1,TERM3),CONSLIST)%];
74       EXIT;
75
76    COMMENT'(IF RANDOM A B) => (IF RANDOM R(A(RANDOM/CONS))
77                                          R(B(RANDOM/NIL)))';
78    SUBSTCONS:
79    [%"IF",TERM1,REDUCE1(TERM2,TERM1::CONSLIST),
80      REDUCE1(SUBST(NIL,TERM1,TERM3),CONSLIST)%];
81
82    END;
83
84    REDUCE1(%NIL%)->REDUCE;
85
86
```

```
[/]
DTRACK 22     CREATED    [ 21.01 19 NOV 1973]
                         16.34     1 11 1973

 1
 2    VARS SLASH9 SLASH22 SLASH36;
 3    [[/PROPS][/GEN][/APPFILE][PPR][/GENSYM][/METAGEN][/INPUT][/TYPE]
 4    [/MACCONS][/EVAL]]
 5    ->SLASH9;
 6    [[/REWRITE][/REDUCE][/FERTILIZE][/GENERALIZE][/INDUCT1][/INDUCT2]]
 7    ->SLASH22;
 8
 9    [[/VERBOSE][/PROVE]]->SLASH36;
10
11    DTRACK(9);
12    APPLIST(SLASH9,DCOMP);
13    DTRACK(22);
14    APPLIST(SLASH22,DCOMP);
15
16    DTRACK(36);
17    APPLIST(SLASH36,DCOMP);
18    APPFILE([/DEFS],DEFINE);
19
```

```
1
2   COMMENT'THIS IS THE NORMALIZE FUNCTION.  IN-LINE COMMENTS EXPLAIN
3   THE REWRITE RULES APPLIED.`;
4
5   VARS REWRITEFN;
6
7   IDENTFN->REWRITEFN;
8
9
10  FUNCTION REWRITE TERM;
11  VARS TERM1 TERM2 TERM3;
12
13  COMMENT'IF TERM IS AN EQUALITY`;
14
15  IF HD(TERM)="EQUAL" THEN
16     HD(TL(TERM))->TERM1;
17     HD(TL(TL(TERM)))->TERM2;
18
19  COMMENT'(EQUAL KNOWN1 KNOWN2) => T OR NIL`;
20     IF TERM1==TERM2 THEN T;EXIT;
21     IF NOTIDENT THEN NIL;EXIT;
22
23  COMMENT'(EQUAL BOOL T) => BOOL`;
24     IF TERM1=T AND BOOLEAN(TERM2) THEN TERM2 EXIT;
25     IF TERM2=T AND BOOLEAN(TERM1) THEN TERM1 EXIT;
26
27  COMMENT'(EQUAL (EQUAL A B) C) =>
28                  (IF (EQUAL A B) (EQUAL C T) (IF C NIL T))`;
29     IF SHD(TERM1)="EQUAL" OR SHD(TERM2)="EQUAL" AND (SWAP;1)
30        THEN
31        [%"IF",TERM1,
32          REWRITE([%"EQUAL",TERM2,T%]),
33          REWRITE([%"IF",TERM2,NIL,T%])%]->TERM;
34        GOTO CONDL;
35        CLOSE;
36
37  COMMENT'(EQUAL X NIL) => (IF X NIL T)`;
38     IF TERM1==NIL OR TERM2==NIL AND (SWAP;1)
39        THEN
40        [%"IF",TERM2,NIL,T%]->TERM;
41        GOTO CONDL;
42        CLOSE;
43
44  COMMENT'GO SEE IF ONE ARG IS A IF`;
45     GOTO CONDARG;
46
47
48
49  COMMENT'TERM IS A IF`;
50
51
52  ELSEIF HD(TERM)="IF" THEN
53
54     CONDL:
55     TL(TERM)->TERM3;
56     HD(TERM3)->TERM1;
```

```
57      TL(TERM3)->TERM3;
58      HD(TERM3)->TERM2;
59      HD(TL(TERM3))->TERM3;
60
61    COMMENT'(IF KNOWN X Y) => X OR Y';
62      IF TERM1==NIL THEN TERM3;EXIT;
63      IF NOTIDENT THEN TERM2;EXIT;
64
65    COMMENT'(IF X Y Y) => Y';
66      IF TERM2==TERM3 THEN TERM2;EXIT;
67
68    COMMENT'(IF X X NIL) => X';
69      IF TERM1==TERM2 AND TERM3==NIL THEN TERM1;EXIT;
70
71    COMMENT'(IF BOOL T NIL) => BOOL';
72      IF BOOLEAN(TERM1) AND TERM2=T AND TERM3==NIL
73         THEN TERM1;EXIT;
74
75    COMMENT'(IF X T (IF Y NIL T)) => (IF Y (IF X T NIL) T)';
76      IF TERM2=T AND SHD(TERM3)="IF" AND
77         HD(TL(TL(TERM3)))==NIL AND HD(TL(TL(TL(TERM3))))=T
78         THEN
79         IF BOOLEAN(TERM1)
80            THEN TERM1;
81            ELSE [%"IF",TERM1,T,NIL%]CLOSE;
82         ->TERM2;
83         HD(TL(TERM3))->TERM1;
84         T->TERM3;
85         [%"IF",TERM1,TERM2,TERM3%]->TERM;
86         CLOSE;
87
88
89    COMMENT'IF TERM1 IS AN IF, DECIDE IF IT SHOULD BE
90    DISTRIBUTED.';
91
92      IF SHD(TERM1)="IF" THEN
93
94    COMMENT'(IF (IF A T2 T3) B C) => (IF A (IF T2 B C)
95             (IF T3 B C)) WHERE T2 OR T3 ISNIL';
96
97         IF HD(TL(TL(TERM1)))==NIL OR HD(TL(TL(TL(TERM1))))==NIL
98            THEN
99            GOTO CONDCOND;
100           CLOSE;
101
102
103
104   COMMENT'(IF (IF A T (* N)) T (* M)) => (IF A T (* N M))';
105        IF TERM2=T AND SHD(TERM3)="*" AND HD(TL(TL(TERM1)))=T
106           AND SHD(HD(TL(TL(TL(TERM1)))))="*"
107           THEN
108           [%"IF",HD(TL(TERM1)),T,"*"::(TL(HD(TL(TL(TL(TERM1)))))
109                                        <>TL(TERM3))%];
110           EXIT;
111
112   COMMENT'(IF (IF A B C) D E)=> (IF A (IF B C E) (IF C D E))
113        WHERE D AND E ARE NOT NIL OR D AND E ARE T AND NIL';
114
115        IF TERM2==NIL AND TERM3/=T THEN GOTO SKIP;
116        ELSEIF TERM3==NIL AND TERM2/=T THEN GOTO SKIP;CLOSE;
```

```
117          CONDCOND:
118          IF SHD(TERM2)="*" OR SHD(TERM3)="*" THEN GOTO SKIP;CLOSE;
119          REWRITE([%"IF",HD(TL(TL(TERM1))),TERM2,TERM3%]);
120          REWRITE([%"IF",HD(TL(TL(TL(TERM1)))),TERM2,TERM3%]);
121          ->TERM3->TERM2;
122          [%"IF",HD(TL(TERM1)),TERM2,TERM3%]->TERM;
123          GOTO CONDL;
124          SKIP:
125          CLOSE;
126
127    COMMENT'TERM IS A NON-IF, NON-EQ FUNCTION CALL';
128       ELSE
129
130    COMMENT'(FOO X (IF A B C) Y) =>
131    (IF A (FOO X B Y) (FOO X C Y))';
132
133       CONDARG:
134       TL(TERM)->TERM1;
135       LOOPIF TERM1/=NIL AND SHD(HD(TERM1))/="IF"
136          THEN
137          TL(TERM1)->TERM1;
138          CLOSE;
139       IF TERM1/=NIL
140          THEN
141          HD(TERM1)->TERM1;
142          [%"IF",HD(TL(TERM1)),REWRITE(SUBST(HD(TL(TL(TERM1))),TERM1,
143                                                TERM)),REWRITE(SUBST(HD(TL(TL(T
L(TERM1)))),TERM1,TERM))%]
144          ->TERM;
145          GOTO CONDL;
146          CLOSE;
147       CLOSE;
148    REWRITEFN();
149    TERM
150    END
151
152
153    FUNCTION NORMALIZE TERM;
154    IF ATOM(TERM) THEN TERM EXIT;
155    REWRITE(HD(TERM)::MAPLIST(TL(TERM),NORMALIZE));
156    END
157
158
159
```

```
[/ FERTILIZ]                    [ 21.02 19 NOV 1973]
DTRACK 22    CREATED   11.56    2 11 1973


 1
 2   FUNCTION FERTREPORT;
 3   IF VERBOSE
 4      THEN
 5      POPTTON();
 6      NL(4);PRSTRING('FERTILIZE WITH ');PPRIND(TERM1,15,1);
 7      PRSTRING('.');NL(2);
 8      CLOSE;
 9   END;
10
11
12
13   FUNCTION FERTILIZE TERM;
14   VARS TERM1 TERM2 TERM3 LHS1 RHS1;
15   IF SHD(TERM)/="IF" THEN 0;EXIT;
16   HD(TL(TERM))->TERM1;
17   HD(TL(TL(TERM)))->TERM2;
18   HD(TL(TL(TL(TERM))))->TERM3;
19
20   COMMENT'LOOK FOR TERMS OF THE FORM (IF (EQUAL LHS RHS) BOOL1 BOOL2)
21      WHERE BOOL2 IS NOT NIL.  IF FOJND, FERTILIZE
22      LHS=RHS INTO BOOL1 AND HIDE IT.';
23
24   IF SHD(TERM1)="EQUAL" AND TERM3/=NIL
25      AND TERM3 AND BOOLEAN(TERM2) AND BOOLEAN(TERM3)
26      THEN
27      HD(TL(TERM1))->LHS1;
28      HD(TL(TL(TERM1)))->RHS1;
29      IF ISSPEC(LHS1) OR ISSPEC(RHS1)
30         THEN GOTO NOFERT;CLOSE;
31      IF FERTIL1(TERM2)
32         THEN
33         ->TERM2;
34         IF FERTILIZE(TERM2) THEN ->TERM2;CLOSE;
35         [%"IF",TERM2,T,GENSTAR([%"IF",TERM1,NIL,TERM3%])%]
36         ->TERM2;
37         REPORT([%"F",STARCOUNT%],FERTREPORT,"FERTILIZE");
38         IF FERTILIZE(TERM3) THEN ->TERM3;CLOSE;
39         IF TERM3=T
40            THEN TERM2;
41            ELSE
42            [%"IF",TERM2,
43              [%"IF",TERM3,
44                T,
45                TERM1%],
46              NIL%];
47            CLOSE;
48         1;
49         EXIT;
50      CLOSE;
51
52   NOFERT:
53   0->TERM1;
54   [%"IF",APPLIST(TL(TERM),
55                  LAMBDA TERM;
56                  IF FERTILIZE(TERM) THEN 1->TERM1; ELSE TERM;CLOSE;
```

```
                END),
  ( IF TERM1 THEN ELSE
      ERASE(ERASE(),ERASE(),ERASE(),ERASE());0;EXIT)%];
1;
END;

FUNCTION FERTIL1 TERM;
VARS LHS2 RHS2;
IF ATOM(TERM) THEN 0;EXIT;
IF HD(TERM)="EQUAL"
   THEN
   HD(TL(TERM))->LHS2;
   HD(TL(TL(TERM)))->RHS2;
COMMENT'NOW LOOK FOR (THE BEST) CROSS FERTILIZATION';
   IF OCCUR(RHS1,RHS2)
      THEN
      IF OCCUR(LHS1,LHS2)
         THEN
         IF CONSCNT(RHS1)<CONSCNT(LHS1)
            THEN SUBST(RHS1,LHS1,LHS2)->LHS2;
            ELSE SUBST(LHS1,RHS1,RHS2)->RHS2;CLOSE;
         ELSE SUBST(LHS1,RHS1,RHS2)->RHS2;CLOSE;
      ELSE
      IF OCCUR(LHS1,LHS2)
         THEN SUBST(RHS1,LHS1,LHS2)->LHS2;
         ELSE GOTO MASSSUBST;EXIT;
      CLOSE;
   [%"EQUAL",LHS2,RHS2%];
   1;
   EXIT;
COMMENT'IF TERM IS AN IF, LOOK FOR ITS "CORE" AND FERTILIZE IT';
IF HD(TERM)="IF"
   THEN
COMMENT'(IF X CORE T) => (IF X FERT(CORE) T), PROVIDED
X DOES NOT CONTAIN LHS1 OR RHS1.';
   IF HD(TL(TL(TL(TERM))))=T
      THEN
      IF OCCUR(LHS1,HD(TL(TERM))) THEN GOTO CHKRHSOCC;
      ELSEIF OCCUR(RHS1,HD(TL(TERM))) THEN GOTO SUBSTLR;
      ELSEIF FERTIL1(HD(TL(TL(TERM))))
         THEN
         ->FOO1;
         [%"IF",HD(TL(TERM)),FOO1,T%];
         1;
         EXIT;
COMMENT'(IF CORE T (*N)) => (IF FERT(CORE) T (*N))';
   ELSEIF HD(TL(TL(TERM)))=T AND SHD(HD(TL(TL(TL(TERM)))))="*"
      THEN
      IF FERTIL1(HD(TL(TERM)))
         THEN
         ->FOO1;
         [%"IF",FOO1,T,HD(TL(TL(TL(TERM))))%];
         1;
         EXIT;
      CLOSE;
COMMENT'IF NOT OF EITHER OF THE ABOVE FORMS, FALL THROUGH
TO MASSIVE SUBSTITUTION.';
   CLOSE;

COMMENT'IF CROSS FERTILIZATION NOT POSSIBLE, TRY MASSIVE
```

```
117   SUBSTITUTION';
118
119   MASSSUBST:
120   IF OCCUR(LHS1,TERM)
121      THEN
122      CHKRHSOCC:
123      IF OCCUR(RHS1,TERM)
124         THEN
125         IF CONSCNT(RHS1)<CONSCNT(LHS1)
126            THEN SUBST(RHS1,LHS1,TERM);
127            ELSE SUBST(LHS1,RHS1,TERM);CLOSE;
128         ELSE SUBST(RHS1,LHS1,TERM);CLOSE;
129   ELSEIF OCCUR(RHS1,TERM)
130      THEN
131      SUBSTLR:SUBST(LHS1,RHS1,TERM);
132      ELSE 0;EXIT;
133   1;
134
135   END;
136
137
138
```

```
[/ GENERALI]                    [ 21.02 19 NOV 1973]
DTRACK 22    CREATED   11.57    2 11 1973


 1
 2
 3   COMMENT'THIS FILE GENERALIZES THE TERM ABOUT TO BE PROVED BY INDUCTION.
 4   WE GENERALIZE ON THE COMMON SUBTERMS ON EITHER SIDE
 5   OF "EQUAL", "IMPLIES" AND "OR" STMTS.';
 6
 7
 8
 9
10
11   COMMENT'FIND ALL COMMON NON-ATOMIC NON-PRIMITIVE SUBTERMS OF TWO TERMS.'
;
12
13   VARS T2 GENRLTLIST ATOMLIST;
14
15   FUNCTION COMSUBT1 T1;
16   VARS X;
17   IF ATOM(T1)
18      THEN
19      OCCUR(T1,T2);
20      ELSE
21      TL(T1)->X;
22      IF (1; LOOPIF X/=NIL THEN LOGAND(COMSUBT1(HD(X)));TL(X)->X;CLOSE;)
23         THEN
24         IF NOT(LISPPRIM(T1)) AND OCCUR(T1,T2)
25            THEN
26            IF NOT(MEMBERID(T1,GENRLTLIST))
27               THEN T1::GENRLTLIST->GENRLTLIST;CLOSE;
28            1;EXIT;
29         CLOSE;
30      0;
31      CLOSE;
32   END;
33
34   FUNCTION COMSUBTERMS T1 T2;
35   IF CONSCNT(T1)>CONSCNT(T2) THEN T1;T2->T1->T2;CLOSE;
36   ERASE(COMSUBT1(T1));
37   END;
38
39
40
41   COMMENT'FIND ALL COMMON SUBTERMS OCCURRING ACROSS EQS AND
42   IMPLIES AND ORS.';
43
44   FUNCTION GENRLT1 TERM;
45   IF ATOM(TERM) THEN EXIT;
46   IF HD(TERM)="EQUAL"
47      THEN
48      COMSUBTERMS(HD(TL(TERM)),HD(TL(TL(TERM))));
49   ELSEIF HD(TERM)="IF"
50      THEN
51      IF ATOM(HD(TL(TERM))) THEN
52      ELSEIF HD(TL(TL(TERM)))=T
53         THEN
54         APPLIST(TL(HD(TL(TERM))),
55                  LAMBDA TERM1;
```

```
   56                 COMSUBTERMS(TERM1,HD(TL(TL(TL(TERM)))));
   57                 END);
   58       ELSEIF HD(TL(TL(TL(TERM))))=T
   59          THEN
   60          APPLIST(TL(HD(TL(TERM))),
   61                 LAMBDA TERM1;
   62                 COMSUBTERMS(TERM1,HD(TL(TL(TERM))));
   63                 END);
   64          CLOSE;
   65       CLOSE;
   66    APPLIST(TL(TERM),GENRLT1);
   67    END;
   68
   69
   70    FUNCTION GENRLTERMS;
   71    VARS GENRLTLIST;
   72    NIL->GENRLTLIST;
   73    GENRLT1();
   74    GENRLTLIST;
   75    END;
   76
   77
   78
   79    COMMENT'THIS FUNCTION MAKES A VERBOSE REPORT ON THE PROGRESS
   80    OF GENERALIZATION.';
   81
   82    FUNCTION GENREPORT;
   83    IF VERBOSE
   84       THEN
   85       POPTTON();
   86       NL(2);
   87       PRSEQAND(4,'GENERALIZE COMMON SUBTERMS BY REPLACING ',
   88                SUBSTLIST,LAMBDA P;PR(FRONT(P));PRSTRING(' BY ');PR(BACK(P))
;END);
   89       NL(2);
   90       PRSTRING('THE GENERALIZED TERM IS:');
   91       NL(2);
   92       PPR(TERM);
   93       NL(2);
   94       CLOSE;
   95    END;
   96
   97    COMMENT'THIS IS THE TOP-LEVEL FUNCTION.  IT GENERALIZES ITS
   98    ARGUMENT AS DESCRIBED, AND THEN PRINTS
   99    A VERBOSE COMMENT IF NEEDED.  NOTE THAT IF THE TERM GENERALIZED
  100    IS NUMERIC, A NUMERIC SKOLEM CONSTANT IS GENERATED FOR IT.';
  101
  102    FUNCTION GENERALIZE TERM;
  103    VARS X SUBSTLIST;
  104    GENRLTERMS(TERM)->X;
  105    IF X=NIL THEN TERM;EXIT;
  106    MAPLIST(X,
  107             LAMBDA TERM;
  108             IF NUMERIC(TERM) THEN "INTGR"; ELSE "XLIST";CLOSE;
  109             GENSKO()->X;
  110             CONSPAIR(TERM,X);
  111             END)->SUBSTLIST;
  112    IF SUBSTLIST=NIL THEN TERM;EXIT;
  113    APPSUBST(SUBSTLIST,TERM)
  114    ->TERM;
```

```
115   (REPORT([%"G",APPLIST(SUBSTLIST,BACK)%],GENREPORT,"GENERALIZE"));
116   APPLIST(SUBSTLIST,
117           LAMBDA X;
118           FRONT(X);BACK(X)->FRONT(X)->BACK(X);
119           END);
120   SUBSTLIST<>GENRLALIST->GENRLALIST;
121   TERM;
122   END;
123
124
```