AI Homework 2

Assumptions

1) all x all y (eats(x,y) & pizza(y)) -> happy(x)
2) all x exists y (foodie(x) -> eats(x,y) & (pizza(y) | salad(y)))
3) all x all y ((eats(x,y) & salad(y)) -> healthy(x))
4) all x (healthy(x) -> gyms(x))
5) all x all y (happy(x) & nice(y) -> -dated(y,x))
6) nice(Ann)
7) foodie(Peter)

Goal

-gyms(Peter) -> -dated(Ann,Peter)

Prover9 Input

```
assign(report_stderr, 2).
set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.

if(Mace4).   % Options for Mace4
  assign(max_seconds, 60).
end_if.

formulas(assumptions).

nice(Ann).
foodie(Peter).
-eats(x,y) | -pizza(y) | happy(x).
```

-foodie(x) | (eats(x,f(x)) & pizza(f(x))) | (eats(x,f(x)) & salad(f(x))).
-eats(x,y) | -salad(y) | healthy(x).
-healthy(x) | gyms(x).
-happy(x) | -nice(y) | -dated(y, x).

end_of_list.

formulas(goals).

gyms(Peter) | -dated(Ann,Peter).

end_of_list.

Prover9 Output

========== Prover9
==========
Prover9 (32) version Dec-2007, Dec 2007.
Process 15457 was started by sahith on Mahalaxmis-MacBook-Pro.local,
Mon Mar 9 11:01:35 2020
The command was
"/private/var/folders/vx/w46b2r650kg7c64c4jhkrrlr0000gp/T/AppTranslocation/E98239BA-3200-4D9B-8BA0-E2786CCD0512/d/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
========== end of head
==========

========== INPUT
==========
assign(report_stderr,2).
set(ignore_option_dependencies).
if(Prover9).
% Conditional input included.
assign(max_seconds,60).
end_if.
if(Mace4).
% Conditional input omitted.

end_if.

formulas(assumptions).
nice(Ann).
foodie(Peter).
-eats(x,y) | -pizza(y) | happy(x).
-foodie(x) | eats(x,f(x)) & pizza(f(x)) | eats(x,f(x)) & salad(f(x)).
-eats(x,y) | -salad(y) | healthy(x).
-healthy(x) | gyms(x).
-happy(x) | -nice(y) | -dated(y,x).
end_of_list.

formulas(goals).
gyms(Peter) | -dated(Ann,Peter).
end_of_list.

============================== end of input ==========================

% Enabling option dependencies (ignore applies only on input).

============================== PROCESS NON-CLAUSAL FORMULAS ==========

% Formulas that are not ordinary clauses:
1 -foodie(x) | eats(x,f(x)) & pizza(f(x)) | eats(x,f(x)) & salad(f(x)) # label(non_clause). [assumption].
2 gyms(Peter) | -dated(Ann,Peter) # label(non_clause) # label(goal). [goal].

============================== end of process non-clausal formulas ===

============================== PROCESS INITIAL CLAUSES ===============

% Clauses before input processing:

formulas(usable).
end_of_list.

formulas(sos).

nice(Ann). [assumption].
foodie(Peter). [assumption].
-eats(x,y) | -pizza(y) | happy(x). [assumption].
-foodie(x) | eats(x,f(x)). [clausify(1)].
-foodie(x) | pizza(f(x)) | salad(f(x)). [clausify(1)].
-eats(x,y) | -salad(y) | healthy(x). [assumption].
-healthy(x) | gyms(x). [assumption].
-happy(x) | -nice(y) | -dated(y,x). [assumption].
-gyms(Peter). [deny(2)].
dated(Ann,Peter). [deny(2)].
end_of_list.

formulas(demodulators).
end_of_list.

============================== PREDICATE ELIMINATION ==========

Eliminating nice/1
3 -happy(x) | -nice(y) | -dated(y,x). [assumption].
4 nice(Ann). [assumption].
Derived: -happy(x) | -dated(Ann,x). [resolve(3,b,4,a)].

Eliminating foodie/1
5 -foodie(x) | eats(x,f(x)). [clausify(1)].
6 foodie(Peter). [assumption].
Derived: eats(Peter,f(Peter)). [resolve(5,a,6,a)].
7 -foodie(x) | pizza(f(x)) | salad(f(x)). [clausify(1)].
Derived: pizza(f(Peter)) | salad(f(Peter)). [resolve(7,a,6,a)].

Eliminating eats/2
8 eats(Peter,f(Peter)). [resolve(5,a,6,a)].
9 -eats(x,y) | -pizza(y) | happy(x). [assumption].
10 -eats(x,y) | -salad(y) | healthy(x). [assumption].
Derived: -pizza(f(Peter)) | happy(Peter). [resolve(8,a,9,a)].
Derived: -salad(f(Peter)) | healthy(Peter). [resolve(8,a,10,a)].

Eliminating healthy/1
11 -salad(f(Peter)) | healthy(Peter). [resolve(8,a,10,a)].

12 -healthy(x) | gyms(x). [assumption].
Derived: -salad(f(Peter)) | gyms(Peter). [resolve(11,b,12,a)].

Eliminating gyms/1
13 -salad(f(Peter)) | gyms(Peter). [resolve(11,b,12,a)].
14 -gyms(Peter). [deny(2)].
Derived: -salad(f(Peter)). [resolve(13,b,14,a)].

Eliminating dated/2
15 -happy(x) | -dated(Ann,x). [resolve(3,b,4,a)].
16 dated(Ann,Peter). [deny(2)].
Derived: -happy(Peter). [resolve(15,b,16,a)].

Eliminating pizza/1
17 -pizza(f(Peter)) | happy(Peter). [resolve(8,a,9,a)].
18 pizza(f(Peter)) | salad(f(Peter)). [resolve(7,a,6,a)].
Derived: happy(Peter) | salad(f(Peter)). [resolve(17,a,18,a)].

Eliminating salad/1
19 happy(Peter) | salad(f(Peter)). [resolve(17,a,18,a)].
20 -salad(f(Peter)). [resolve(13,b,14,a)].
Derived: happy(Peter). [resolve(19,b,20,a)].

Eliminating happy/1
21 happy(Peter). [resolve(19,b,20,a)].
22 -happy(Peter). [resolve(15,b,16,a)].
Derived: $F. [resolve(21,a,22,a)].

============================== end predicate elimination =============

Auto_denials: (no changes).

Term ordering decisions:
Predicate symbol precedence: predicate_order([ ]).
Function symbol precedence: function_order([ ]).
After inverse_order: (no changes).
Unfolding symbols: (none).

Auto_inference settings:

```
  % set(neg_binary_resolution).  % (HNE depth_diff=0)
  % clear(ordered_res).  % (HNE depth_diff=0)
  % set(ur_resolution).  % (HNE depth_diff=0)
    % set(ur_resolution) -> set(pos_ur_resolution).
    % set(ur_resolution) -> set(neg_ur_resolution).

Auto_process settings:  (no changes).

============================== PROOF
================================

% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 23.
% Level of proof is 7.
% Maximum clause weight is 0.
% Given clauses 0.

1 -foodie(x) | eats(x,f(x)) & pizza(f(x)) | eats(x,f(x)) & salad(f(x)) # label(non_clause).
[assumption].
2 gyms(Peter) | -dated(Ann,Peter) # label(non_clause) # label(goal).  [goal].
3 -happy(x) | -nice(y) | -dated(y,x).  [assumption].
4 nice(Ann).  [assumption].
5 -foodie(x) | eats(x,f(x)).  [clausify(1)].
6 foodie(Peter).  [assumption].
7 -foodie(x) | pizza(f(x)) | salad(f(x)).  [clausify(1)].
8 eats(Peter,f(Peter)).  [resolve(5,a,6,a)].
9 -eats(x,y) | -pizza(y) | happy(x).  [assumption].
10 -eats(x,y) | -salad(y) | healthy(x).  [assumption].
11 -salad(f(Peter)) | healthy(Peter).  [resolve(8,a,10,a)].
12 -healthy(x) | gyms(x).  [assumption].
13 -salad(f(Peter)) | gyms(Peter).  [resolve(11,b,12,a)].
14 -gyms(Peter).  [deny(2)].
15 -happy(x) | -dated(Ann,x).  [resolve(3,b,4,a)].
16 dated(Ann,Peter).  [deny(2)].
17 -pizza(f(Peter)) | happy(Peter).  [resolve(8,a,9,a)].
18 pizza(f(Peter)) | salad(f(Peter)).  [resolve(7,a,6,a)].
19 happy(Peter) | salad(f(Peter)).  [resolve(17,a,18,a)].
20 -salad(f(Peter)).  [resolve(13,b,14,a)].
21 happy(Peter).  [resolve(19,b,20,a)].
```

22 -happy(Peter). [resolve(15,b,16,a)].
23 $F. [resolve(21,a,22,a)].

========== end of proof ==========

========== STATISTICS
==========

Given=0. Generated=1. Kept=0. proofs=1.
Usable=0. Sos=0. Demods=0. Limbo=0, Disabled=21. Hints=0.
Weight_deleted=0. Literals_deleted=0.
Forward_subsumed=0. Back_subsumed=0.
Sos_limit_deleted=0. Sos_displaced=0. Sos_removed=0.
New_demodulators=0 (0 lex), Back_demodulated=0. Back_unit_deleted=0.
Demod_attempts=0. Demod_rewrites=0.
Res_instance_prunes=0. Para_instance_prunes=0. Basic_paramod_prunes=0.
Nonunit_fsub_feature_tests=0. Nonunit_bsub_feature_tests=0.
Megabytes=0.02.
User_CPU=0.00, System_CPU=0.01, Wall_clock=0.

========== end of statistics ==========

========== end of search ==========

THEOREM PROVED

Exiting with 1 proof.

Process 15457 exit (max_proofs) Mon Mar 9 11:01:35 2020

Conclusion

Hence by using resolution refutation, the conclusion is proved to be true.