





% This file contains the Forward Chaining Interpreter for Joel Wilhelm

% and Dallas Barnett's Homework 4 / 5.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Directives (must be at the top of your SWI Prolog program.

% Syntax may differ in other Prolog interpreters. This is

% needed so the program can assert the fact predicate.

:- dynamic fact(1) .

% Operator definitions needed for forward chaining if-then rules.

:- op(800,fx,if).

:- op(700,xfx,then).

:- op(300,xfy,or).

:- op(200,xfy,and).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%defining the BN relations

parent(julianA, didHack).

parent(julianA, possessedInfo).

parent(julianA, didEscape).

parent(edwardS, didEScape).

parent(edwardS, hasClearance).

parent(edwardS, didStealInfo).

parent(didHack, inEmbassy).

parent(didHack, possessedInfo).

parent(didEscape, possessedInfo).

parent(didEscape, hasClearance).

parent(didStealInfo, hasClearance).

parent(didStealInfo, govWorker).

parent(inEmbassy, passedLeaks).

parent(possessedInfo, passedLeaks).

parent(possessedInfo, infoLeaked).

parent(hasClearance, infoLeaked).

parent(hasClearance, stoodTrial).

parent(govWorker, stoodTrail).

%prior probabilities

p(julianA, 0.8).

p(edwardS, 0.8).

%conditional probabilities

p(didHack, [julianA], 0.5).

p(didHack, [not(julianA)], 0.4).

p(didEscape, [julianA, edwardS], 0.5).

p(didEscape, [julianA, not(edwardS)], 0.3).

p(didEscape, [not(julianA), edwardS], 0.3).

p(didEscape, [not(julianA), not(edwardS)], 0.4).

p(didStealInfo, [edwardS], 0.7).

p(didStealInfo, [not(edwardS)], 0.7).

p(inEmbassy, [didHack], 0.456).

p(inEmbassy, [not(didHack)], 0.56).

p(possessedInfo, [didHack, julianA, didEscape], 0.543).

p(possessedInfo, [didHack, julianA, not(didEscape)], 0.62).

p(possessedInfo, [didHack, not(julianA), didEscape], 0.454).

p(possessedInfo, [didHack, not(julianA), not(didEscape)], 0.756).

p(possessedInfo, [not(didHack), julianA, didEscape], 0.456).

p(possessedInfo, [not(didHack), julianA, not(didEscape)], 0.523).

p(possessedInfo, [not(didHack), not(julianA), didEscape], 0.675).

p(possessedInfo, [not(didHack), not(julianA), not(didEscape)], 0.345).

p(hasClearance, [didEscape, edwardS, didStealInfo], 0.4).

p(hasClearance, [didEscape, edwardS, not(didStealInfo)], 0.6).

p(hasClearance, [didEscape, not(edwardS), didStealInfo], 0.3).

p(hasClearance, [didEscape, not(edwardS), not(didStealInfo)], 0.8).

p(hasClearance, [not(didEscape), edwardS, didStealInfo], 0.45).

p(hasClearance, [not(didEscape), edwardS, not(didStealInfo)], 0.42).

p(hasClearance, [not(didEscape), not(edwardS), didStealInfo], 0.24).

p(hasClearance, [not(didEscape), not(edwardS), not(didStealInfo)], 0.32).

p(govWorker, [didStealInfo], 0.7).

p(govWorker, [not(didStealInfo)], 0.8).

p(passedLeaks, [inEmbassy, possessedInfo], 0.89).

p(passedLeaks, [inEmbassy, not(possessedInfo)], 0.46).

p(passedLeaks, [not(inEmbassy), possessedInfo], 0.3).

p(passedLeaks, [not(inEmbassy), not(possessedInfo)], 0.1).

p(infoLeaked, [possessedInfo, hasClearance], 0.9).

p(infoLeaked, [possessedInfo, not(hasClearance)], 0.1).

p(infoLeaked, [not(possessedInfo), hasClearance], 0.54).

p(infoLeaked, [not(possessedInfo), not(hasClearance)], 0.2).

p(stoodTrial, [hasClearance, govWorker], 0.34).

p(stoodTrial, [hasClearance, not(govWorker)], 0.34).

p(stoodTrial, [not(hasClearance), govWorker], 0.5).

p(stoodTrial, [not(hasClearance), not(govWorker)], 0.1).

justice:- guilty,sentencing.

guilty:-

write('Did the suspect hack an agency? (y or n): '), read(Ans1), trans1(Ans1, DidHack),

write('Did the suspect escape the country? (y or n): '), read(Ans2), trans2(Ans2, DidEscape),

write('Did the suspect steal any information? (y or n): '), read(Ans3), trans3(Ans3, DidStealInfo),

L = [DidHack, DidEscape, DidStealInfo],

prob(edwardS, L, EdwardProb),

prob(julianA, L, JulianProb),

mostGuilty(EdwardProb, JulianProb).

mostGuilty(X,Y):- (X<Y), write('Edward Snowden is most guilty, with a probability of '), write(X), nl.

mostGuilty(X,Y):- (Y<X), write('Julian Assange is most guilty, with a probability of '), write(Y), nl.

trans1(y, didHack).

trans1(n, not(didHack)).

trans2(y, didEscape).

trans2(n, not(didEscape)).

trans3(y, didStealInfo).

trans3(n, not(didStealInfo)).

% Starts the interpreter

sentencing :- ask\_user, forward.

ask\_user :- write('Was the suspect in the military? (y or n): '), read(Ans), assert\_military(Ans).

assert\_military(y) :- assert(fact(military)), question\_2.

assert\_military(n) :- question\_2.

question\_2 :- write('Does the suspect have a criminal record? (y or n): '), read(Ans), assert\_criminal(Ans).

assert\_criminal(y) :- assert(fact(criminal)), question\_3.

assert\_criminal(n) :- question\_3.

question\_3 :- write('Does the suspect show remorse? (y or n): '), read(Ans), assert\_remorse(Ans).

assert\_remorse(y) :- assert(fact(remorse)), question\_4.

assert\_remorse(n) :- question\_4.

question\_4 :- write('Did the suspect willingly commit this crime? (y or n): '), read(Ans), assert\_willing(Ans).

assert\_willing(y) :- assert(fact(willing)), question\_5.

assert\_willing(n) :- question\_7.

question\_5 :- write('Did the suspect intend to interfere with the operation or success of the

armed forces of the United States of America, or promote success of its enemies? (y or n): '),

read(Ans), assert\_serious(Ans).

assert\_serious(y) :- assert(fact(serious)), question\_7.

assert\_serious(n) :- question\_6.

question\_6 :- write('Did the suspect intend to cause insubordination, disloyalty, mutiny, refusal

of duty, or to obstruct the recruitment or enlistment service of the United States of America?

(y or n): '), read(Ans), assert\_petty(Ans).

assert\_petty(y) :- assert(fact(petty)), question\_7.

assert\_petty(n) :- question\_7.

question\_7 :- write('Was the suspect mentally ill? (y or n): '), read(Ans), assert\_ill(Ans).

assert\_ill(y) :- assert(fact(ill)), write('End of questions').

assert\_ill(n) :- write('End of questions.').

% Rule derivation

if military then the\_punishment\_for\_the\_suspect\_is\_death.

if not(military) and criminal and willing and serious then the\_punishment\_for\_the\_suspect\_is\_death.

if not(military) and criminal and willing and petty then the\_punishment\_for\_the\_suspect\_is\_life\_with\_possible\_parole.

if not(military) and criminal and not(willing) and not(ill) then the\_punishment\_for\_the\_suspect\_is\_10000\_dollar\_fine\_and\_20\_years.

if not(military) and criminal and not(willing) and ill then the\_punishment\_for\_the\_suspect\_is\_20\_years.

if not(military) and not(criminal) and willing and serious then the\_punishment\_for\_the\_suspect\_is\_death.

if not(military) and not(criminal) and willing and petty then the\_punishment\_for\_the\_suspect\_is\_10000\_dollar\_fine\_and\_30\_years.

if not(military) and not(criminal) and not(willing) and remorse and ill then the\_punishment\_for\_the\_suspect\_is\_5000\_dollar\_fine\_and\_mental\_health\_help.

if not(military) and not(criminal) and not(willing) and remorse and not(ill) then the\_punishment\_for\_the\_suspect\_is\_5000\_dollar\_fine\_and\_1\_year.

if not(military) and not(criminal) and not(willing) and not(remorse) and ill then the\_punishment\_for\_the\_suspect\_is\_7500\_dollar\_fine\_and\_5\_years.

if not(military) and not(criminal) and not(willing) and not(remorse) and not(ill) then the\_punishment\_for\_the\_suspect\_is\_7500\_dollar\_fine\_and\_10\_years.