CS 6374 Computational Logic: Project
The Automatic Pharmacist

Nithya Shree Kannan (nxk210045) Saurabh Sanjay Mathur (sxm200015)

1. Introduction

The pharmacy environment is a unique environment in the field of medicine being that it is a place of informal counseling between the pharmacist and the patient. In most cases, in a retail or clinical pharmacy setting, pharmacists have plenty of opportunities to counsel patients concerning their medications and overall general health. Pharmacists have an important role in the distribution of medications along with bearing the knowledge of the importance of medication adherence.

2. Problem definition

Our project focuses on automating the process of the pharmacists specific to over the counter (OTC) medicines using Answer Set Programming(ASP). This enables patients suffering from minor ailments to self-medicate with the correct information about which medicine to consume, the side effects and if it interacts with any medication the patients are currently on.

Background knowledge: Interactions of OTC medicine with existing conditions.
Given: A set of symptoms and a set of existing medical conditions of a patient.
Todo: (1) Diagnose the patient with an ailment.
(2) Suggest safe OTC medicine for ailment and symptoms.

3. Approach and Challenges

One of the biggest challenges we faced was the immense number of OTCs in the market. Our solution was to reduce the scope of the program to the most common ailments and their prescribed medication. Our variables were the symptoms and their medication.We chose symptoms of the common cold and flu such as headache,runny nose, sneezing, chills, sore throat and fever. We also covered conjunctivitis, food poisoning, rash and measles. The OTCs we focused on are tylenol, advil, aspirin, motrin, nasalcrom, flonase, hydrocortisone, zaditor, benadryl and dramamine. We also factored in the factors in the patient's lifestyle which could interact with the medicines such as usage of alcohol, pregnancies, intake of any other medications such as for heart diseases (blood thinners), tuberculosis, blood pressure (ACE inhibitors), and lithium. We linked these factors to the medicines it affects using the `adverse_interaction` predicates and used that as a condition in the `can_take` predicates.

One of the challenges was reducing the scope of the project since there are so many options for each condition and we chose the most common and popular options used by everyone. To make our project more presentable and user friendly, we have added a web interface for the `should_take` query.

3.1 Logical Ontology

We formalized our approach using the following logical ontology. We used this ontology to create the knowledge base using information from DrugBank (https://go.drugbank.com/) and webmd (https://www.webmd.com/).

Predicate	**Explanation**
`symptom(headache).`	Headache is a symptom.
`ailment(common_cold).`	Common cold is an ailment
`factor(alcohol).`	Alcohol can be a factor that interacts with a medication.

`has_ailment(P, flu) :-`
`person(P),`
`has_symptom(P, fever),`
`has_symptom(P, headache),`
`has_symptom(P, chills),`
`has_symptom(P, body_ache).`
	Patient P has ailment flu if P has a symptom of fever,headache,chills,body ache.

`medicine(tylenol).`	Tylenol is a medicine.
`treats(tylenol, fever).`	Tylenol treats fever.
`adverse_interaction(tylenol,alcohol).`	Tylenol has an adverse interaction towards alcohol.

`should_take(P, M) :-`
`person(P),`
`medicine(M),`
`has_ailment(P, A),`
`treats(M, A),`
`can_take(P, M).`
	Patient P should take medicine M if patient P has an ailment A and the medicine M treats ailment A and if P can take medicine M

`should_take(P, M) :-`
`person(P),`
`medicine(M),`
`has_symptom(P, S),`
`treats(M, S),`
`can_take(P, M).`
	Patient P should take medicine M if patient P has symptom S and the medicine M treats symptom S and if P can take medicine M.

`can_take(P, M) :-`
`  person(P),`
`  medicine(M),`
`  not -can_take(P, M).`
	Patient P may take a medicine M unless some exceptional circumstance makes them unable to take it.

`-can_take(P, M) :-`
`  person(P),`
`  factor(X),`
`  existing(P, X),`
`  adverse_interaction(M, X).`
	Patient P cannot take medicine M if factor X exists in patient P and factor X has an adverse interaction with medicine M.

3.2 Implementation

Our project has two kinds of executables:

1. `main.lp`: A pure s(CASP) file. This can be used along with the s(CASP) Command Line Interface (CLI)
2. `web/main.py`: A web server based user-interface. This requires Python 3.7 and Flask along with the s(CASP) CLI.

Results and Sample Queries

s(CASP) CLI and `main.lp`

Instructions to run the code:

1. Download the entire project.
2. Run `scasp --dcc -s0 main.lp`

Case 1: Suggesting Medicine

```
person(alice).
has_symptom(alice, fever).
?- should_take(alice, X).

Answer sets
   1. X = tylenol
   2. X = advil
   3. X = motrin
```

Case 2: Suggesting Medicine

```
person(alice).
has_symptom(alice, fever).
existing(alice, alcohol).
?- should_take(alice, X).

Answer sets
   1. X = motrin
```

Case 3: Suggesting Medicine

```
person(alice).
has_symptom(alice, fever).
existing(alice, alcohol).
existing(alice, pregnancy).
?- should_take(alice, X).

Answer sets
      None
```

Case 4: Checking for conflicts

```
% Which pre-existing factors prevent Alice from taking medication for her
symptoms?
person(alice).
has_symptom(alice, fever).
existing(alice, alcohol).
existing(alice, pregnancy).
?-has_symptom(alice,S),treats(M,S),existing(alice,E),adverse_interaction(M,E)
.

Answer sets:
   1. S = fever, M = motrin, E = pregnancy
   2. S = fever, M = tylenol, E = alcohol
   3. S = fever, M = advil, E = alcohol
   4. S = fever, M = advil, E = pregnancy
```

Case 5: Diagnosing ailments

```
person(alice).
has_symptom(alice, fever).
has_symptom(alice, headache).
has_symptom(alice, chills).
has_symptom(alice, body_ache).
?- has_ailment(alice, A).

Answer sets:
   1. A = flu
```

Case 6: Diagnosing ailments

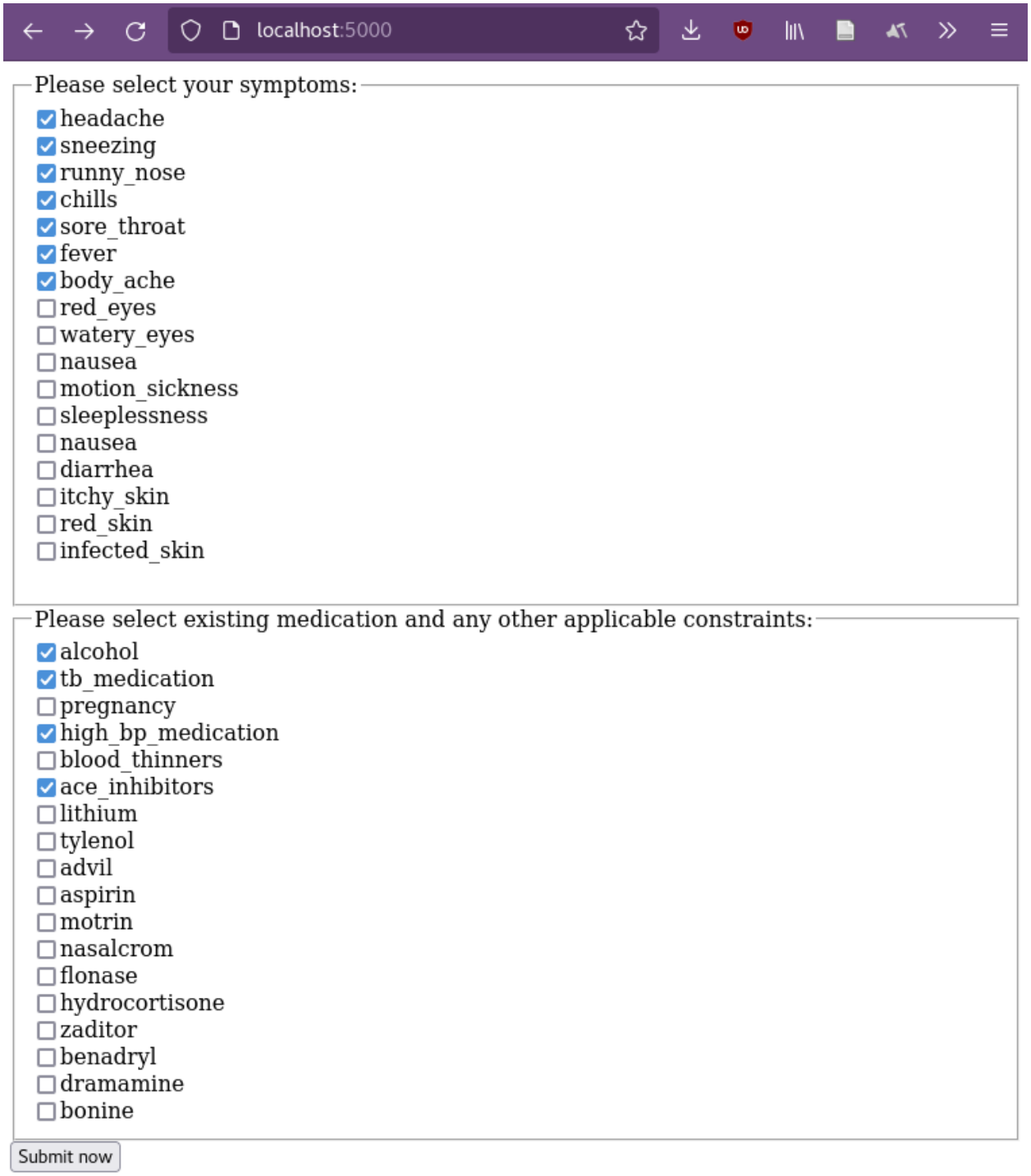
```
person(alice).
has_symptom(alice, red_eyes).
has_symptom(alice, watery_eyes).
?- has_ailment(alice, A).

Answer sets:
   1. A = conjunctivitis
```

Web interface

Instructions to run the web server:

1. Download the entire project.
2. Change directory to web: `cd web`
3. Run the web server: `python main.py`
4. Navigate to `http://localhost:5000` on a web browser.


localhost:5000
Please select your symptoms:
headache
sneezing
runny_nose
chills
sore_throat
fever
body_ache
red_eyes
watery_eyes
nausea
motion_sickness
sleeplessness
nausea
diarrhea
itchy_skin
red_skin
infected_skin
Please select existing medication and any other applicable constraints:
alcohol
tb_medication
pregnancy
high_bp_medication
blood_thinners
ace_inhibitors
lithium
tylenol
advil
aspirin
motrin
nasalcrom
flonase
hydrocortisone
zaditor
benadryl
dramamine
bonine
Submit now


```
AILMENT(S) diagnosed:

Answer 1:
Ailment = cold

Answer 2:
Ailment = flu
---------------


MEDICINE(S) found:

Answer 1:
Medicine = benadryl
---------------


CONFLICT(S):
Answer 1:
Symptom = headache
Medicine = advil
Existing = alcohol

Answer 2:
Symptom = fever
Medicine = tylenol
Existing = alcohol

Answer 3:
Symptom = headache
Medicine = motrin
Existing = ace_inhibitors

Answer 4:
Symptom = fever
Medicine = advil
Existing = alcohol

Answer 5:
Symptom = fever
Medicine = advil
Existing = high_bp_medication

Answer 6:
Symptom = fever
Medicine = motrin
Existing = high_bp_medication

Answer 7:
Symptom = fever
Medicine = motrin
Existing = ace_inhibitors

Answer 8:
Symptom = headache
Medicine = advil
Existing = high_bp_medication

Answer 9:
Symptom = headache
Medicine = motrin
Existing = high_bp_medication
```

s(CASP) code

```
symptom(headache).
symptom(sneezing).
symptom(runny_nose).
symptom(chills).
symptom(sore_throat).
symptom(fever).
symptom(red_eyes).
symptom(watery_eyes).
symptom(nausea).
symptom(motion_sickness).
symptom(sleeplessness).
symptom(nausea).
symptom(diarrhea).
symptom(itchy_skin).
symptom(red_skin).
symptom(infected_skin).
symptom(body_ache).

ailment(common_cold).
ailment(flu).
ailment(conjunctivitis).
ailment(measles).
ailment(food_poisoning).
ailment(rash).

medicine(tylenol).
medicine(advil).
medicine(aspirin).
medicine(motrin).
medicine(nasalcrom).
medicine(flonase).
medicine(hydrocortisone).
medicine(zaditor).
medicine(benadryl).
medicine(dramamine).
medicine(bonine).


treats(tylenol, fever).
treats(advil, fever).
treats(motrin, fever).
```

```
treats(motrin, headache).
treats(advil, headache).
treats(flonase, rash).
treats(hydrocortisone, rash).
treats(zaditor, conjunctivitis).
treats(benadryl, sleeplessness).
treats(benadryl, cold).
treats(dramamine, nausea).
treats(dramamine, motion_sickness).
treats(bonine, motion_sickness).


factor(alcohol).
factor(tb_medication).
factor(pregnancy).
factor(high_bp_medication).
factor(blood_thinners).
factor(ace_inhibitors).
factor(lithium).

factor(tylenol).
factor(advil).
factor(aspirin).
factor(motrin).
factor(nasalcrom).
factor(flonase).
factor(hydrocortisone).
factor(zaditor).
factor(benadryl).
factor(dramamine).
factor(bonine).

% adverse_interaction(M, F) : medicine M has an adverse reaction with an
existing factor F. F may be a drug or a condition.
adverse_interaction(tylenol, alcohol).
adverse_interaction(tylenol, tb_medication).

adverse_interaction(advil, alcohol).
adverse_interaction(advil, aspirin).
adverse_interaction(advil, pregnancy).
adverse_interaction(advil, high_bp_medication).

adverse_interaction(aspirin, alcohol).
```

```
adverse_interaction(aspirin, blood_thinners).
adverse_interaction(aspirin, advil).

adverse_interaction(motrin, ace_inhibitors).
adverse_interaction(motrin, lithium).
adverse_interaction(motrin, high_bp_medication).
adverse_interaction(motrin, pregnancy).

adverse_interaction(hydrocortisone, blood_thinners).
adverse_interaction(hydrocortisone, aspirin).


% has_ailment(P, A) : person P is diagnosed with ailment A
has_ailment(P, flu) :- person(P), has_symptom(P, fever), has_symptom(P,
headache), has_symptom(P, chills), has_symptom(P, body_ache).
has_ailment(P, cold) :- person(P), has_symptom(P, headache), has_symptom(P,
runny_nose), has_symptom(P, sneezing), has_symptom(P, sore_throat).
has_ailment(P, conjunctivitis) :- person(P), has_symptom(P, red_eyes),
has_symptom(P, watery_eyes).
has_ailment(P, food_poisoning) :- person(P), has_symptom(P, bloating),
has_symptom(P, nausea).
has_ailment(P, food_poisoning) :- person(P), has_symptom(P, diarrhea).
has_ailment(P, rash) :- person(P), has_symptom(P, itchy_skin), has_symptom(P,
red_skin).
has_ailment(P, rash) :- person(P), has_symptom(P, infected_skin).

% can_take(P, M) : person P can take medicine M
can_take(P, M) :- person(P), medicine(M), not -can_take(P, M).

% -can_take(P, M) : exception case - person P cannot take medicine M
-can_take(P, M) :- person(P), factor(X), existing(P, X),
adverse_interaction(M, X).


% should_take(P, M) : person P should take medicine M to treat some symptom
or some ailment
should_take(P, M) :- person(P), medicine(M), has_ailment(P, A), treats(M, A),
can_take(P, M).
should_take(P, M) :- person(P), medicine(M), has_symptom(P, S), treats(M, S),
can_take(P, M).
```

```
% Query
person(alice).
has_symptom(alice, red_eyes).
has_symptom(alice, watery_eyes).
?- has_ailment(alice, A).
```