Automated Medical Diagnosis

By

Arunkumar Rajappan
Artificial Intelligence
Prof. Mithun Balakrishnan

1 PROBLEM DESCRIPTION

This project is about automated Medical diagnosis given a set of symptoms and a knowledge base.

2 PROPOSED SOLUTION

For implementing this, I will be creating a knowledge-base for diseases and symptoms. Then using this knowledge based I will use A* search algorithm for getting the solution.

3 FULL IMPLEMENTATION DETAILS

Following are implementation procedure:

Step 1: Create a Knowledge Base (KB) that contains:

Step A: "Symptom" class with 10 or more symptom instances under the "Symptom" class directly or under some sub-classes inside the "Symptom" class.

Step B: Add the appropriate properties for the symptom sub-classes/instances.

Step B: "Disease" class with 5 or more disease instances under the "Disease" class directly or under some sub-classes inside the "Disease" class.

Step C: Associate the appropriate symptom subclasses/instances with the disease subclasses/instances via properties/attributes/relationships.

Step i: Specify a weight for each one of those association instances.

Step ii: The sum of all the associated symptom weights for a particular disease is its total cost.

Step 2: Take the users symptoms as input.

Step 3: Using the KB and inference (explained below), map each user entered symptom to a instance/sub-class in the "Symptoms" class.

Step 4: Create a search node from the user input (i.e. a node that contains the unsatisfied input symptoms (initially all user symptoms), total cost so far (i.e. g(n) which is initially null), estimated goal cost (i.e. f(n) which is initially null), parent node (initially null), disease name (initially null) etc.) and add it into a priority queue that stores search nodes sorted on their f(n) weights.

Note: that the g(n) and h(n) value that I am using for this is:

- **G(n) – Number of Symptoms that are initialized at a particular state i.e. which have indicator/instance value as "1"**
- **H(n) – 1-(1/Number of diseases satisfying those initialized symptoms). In this way only the disease having all the symptoms would be having h(n) as ZERO. Thus heuristic at our goal node would be zero**
- **States – state is the list of input symptoms with indicator values i.e. they have been initialized or observed or not. i.e. [S1, S2] → [0,1]**
- **Initial State – [0,0,…0]**

- **Goal State – [1,1,…,1]**
- **Cost Function f(n) – G(n) + H(n)**

Step 5: Run a SPARQL query to get all the diseases that satisfies at least one unsatisfied symptom (any one). The weight associated for a disease for that satisfying symptom is the action cost (i.e. g(n) in the A* search cost computation equation). One heuristic (i.e. h(n) value) can be: the total cost of the disease associated with search node minus the action cost (i.e. f(n) = disease total cost).

Step 5: Add all the search nodes from Step 4 into the queue and sort them based on their f(n) weights.

Step 6: Repeat steps 4-5 until you get a node on the top of the queue that does NOT have any unsatisfied symptoms remaining. That node's disease label is our diagnosed disease for the patient's symptom input.

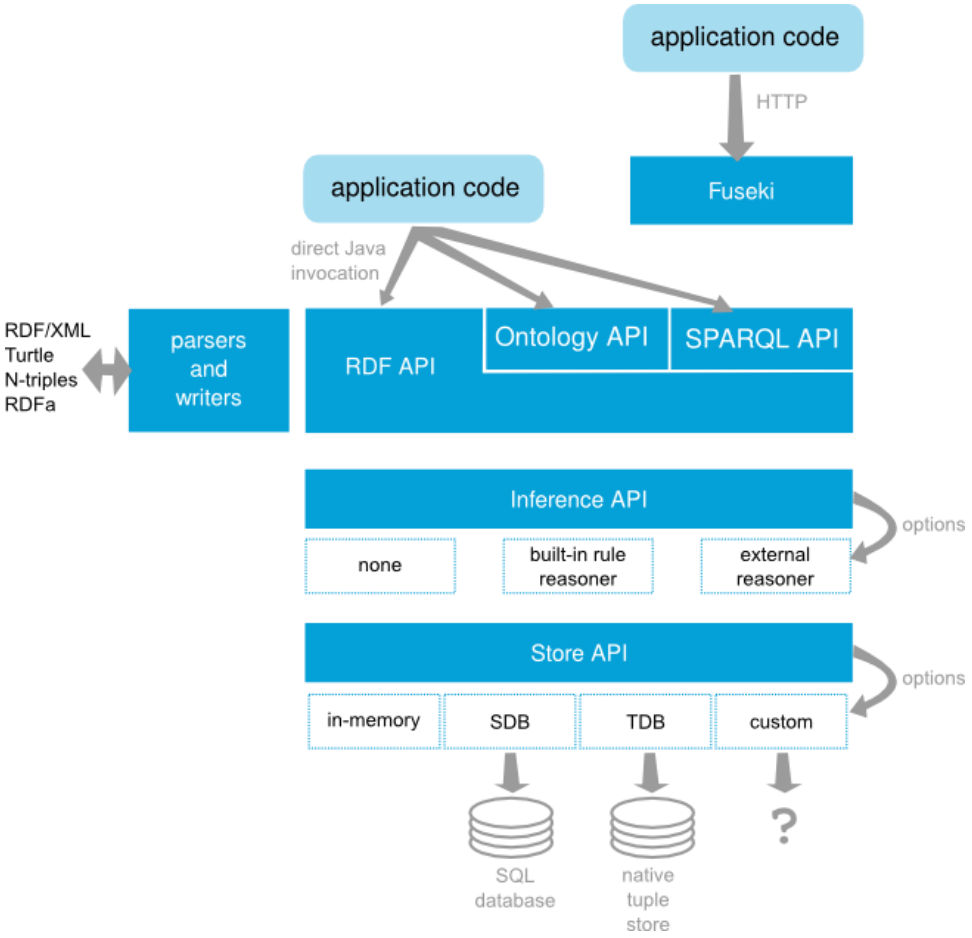
4 EXAMPLES

NA

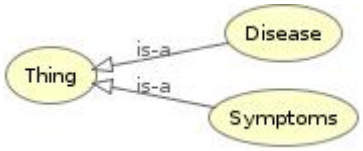
5 PROGRAMMING TOOLS (INCLUDING THIRD PARTY SOFTWARE TOOLS USED)

1) Protégé for creating OWL Ontology
2) Apache Jena for processing the OWL KB
3) Eclipse/Java

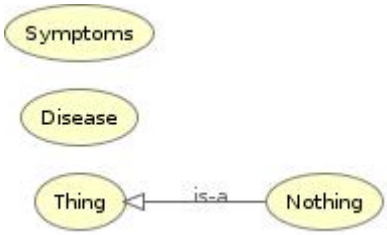
6 ARCHITECTURAL DIAGRAM



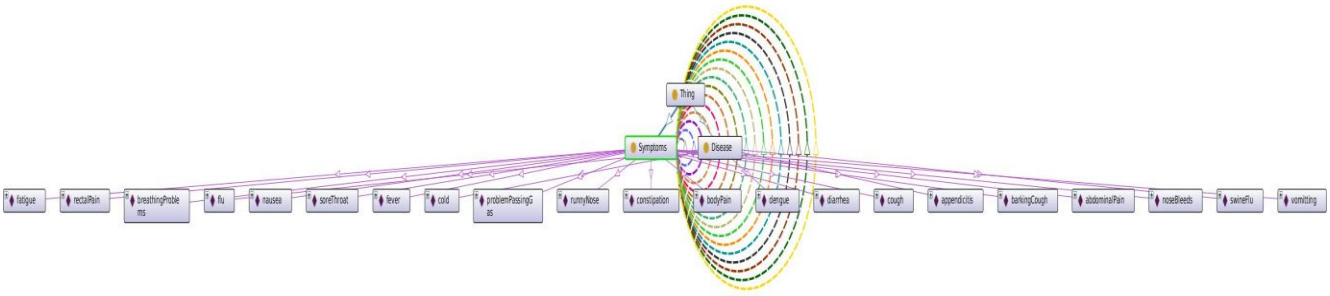
Jena API Architecture Overview

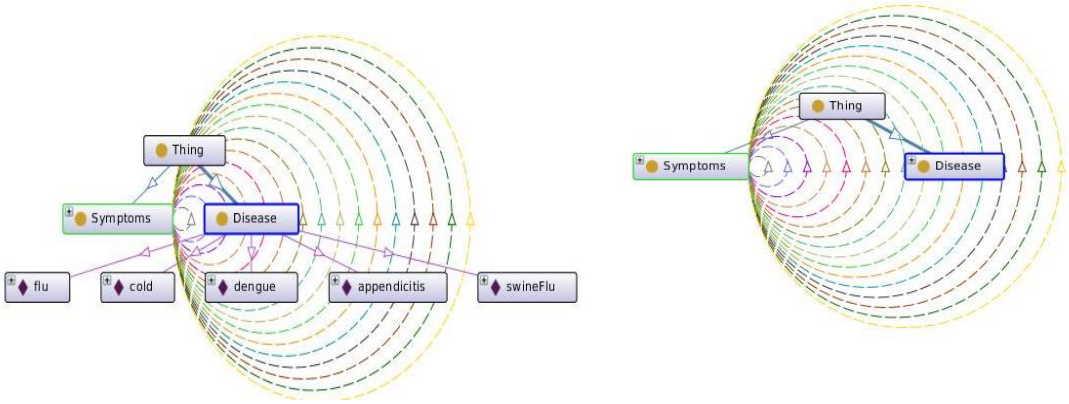


Owl Asserted Model

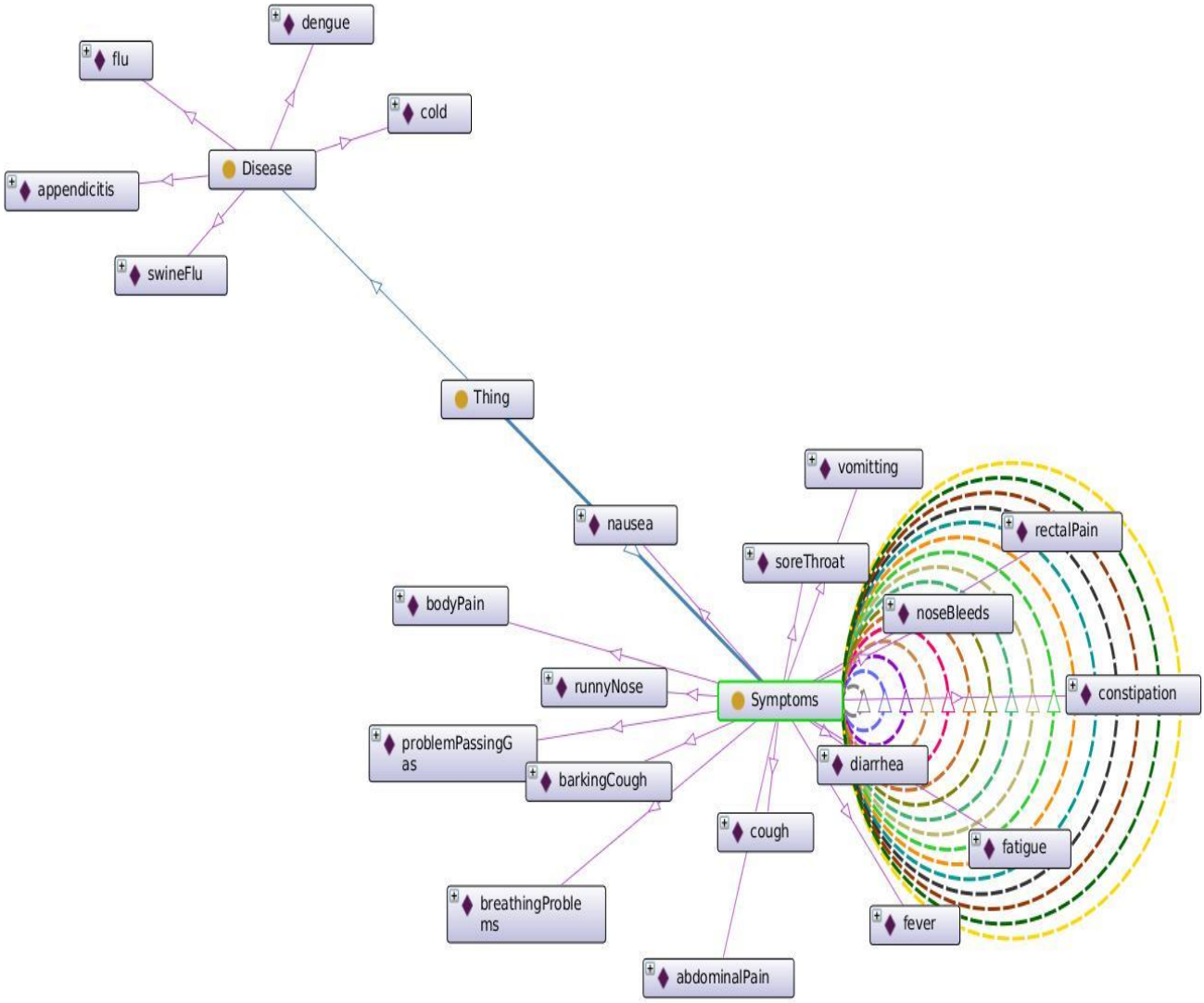


OWL Inferred Model

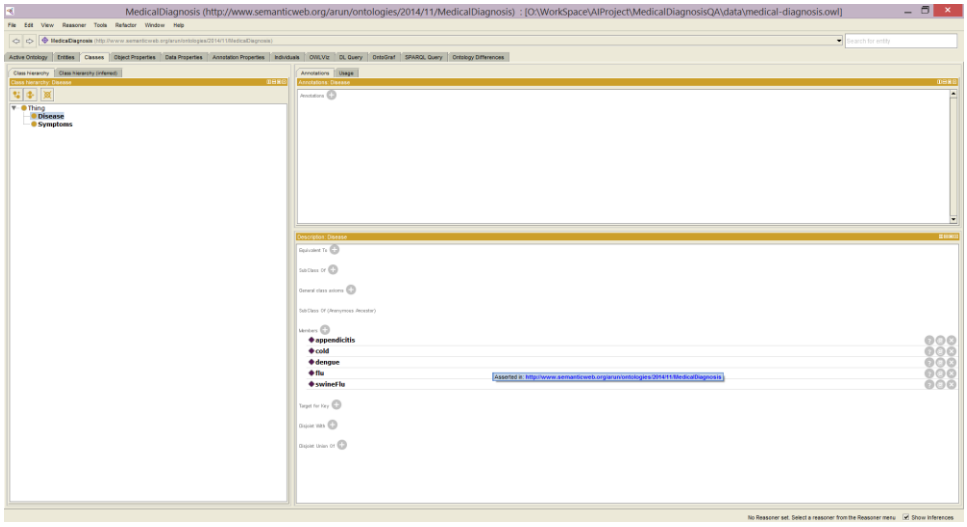


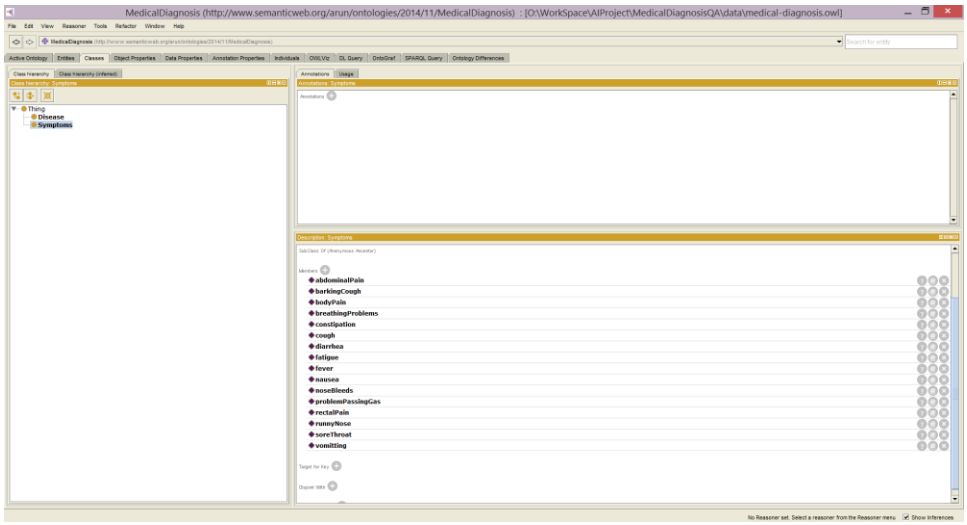


Ontology Graph-1

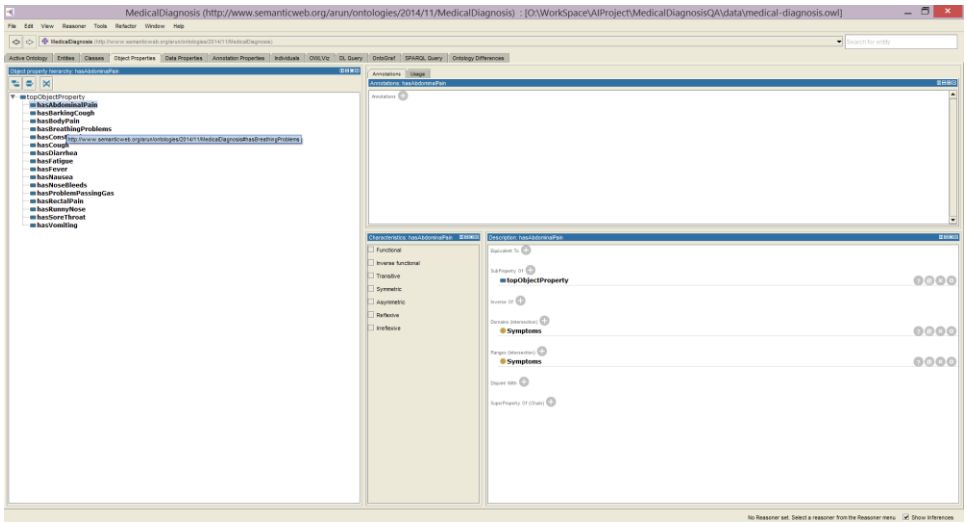


Ontology Graph – 2

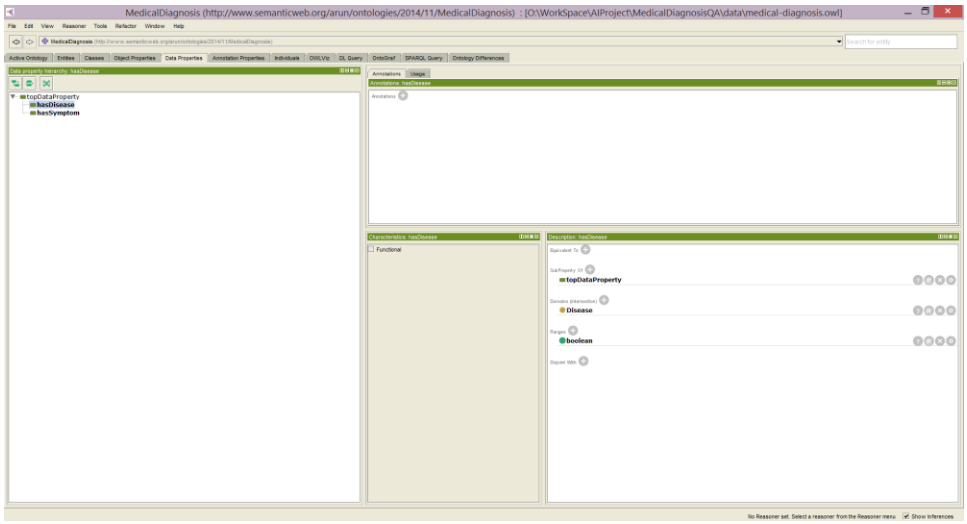


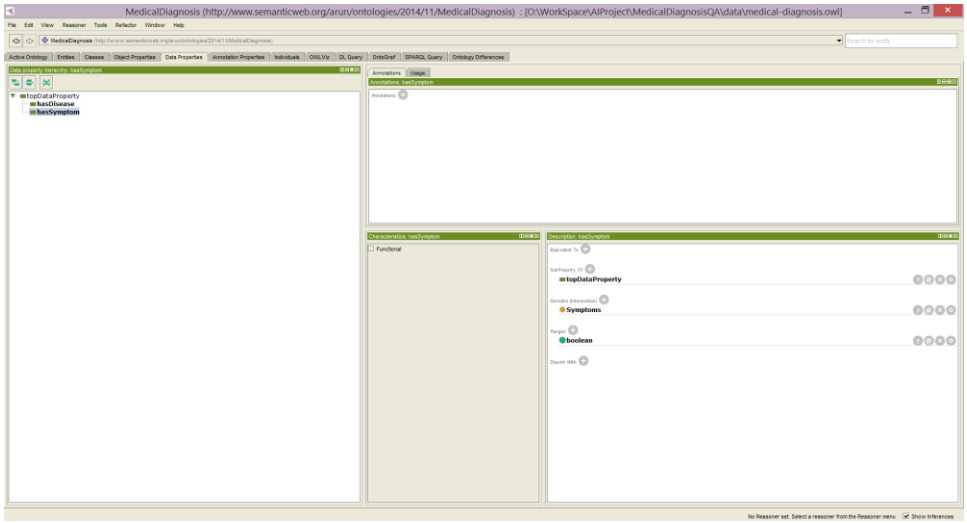


Classes in OWL

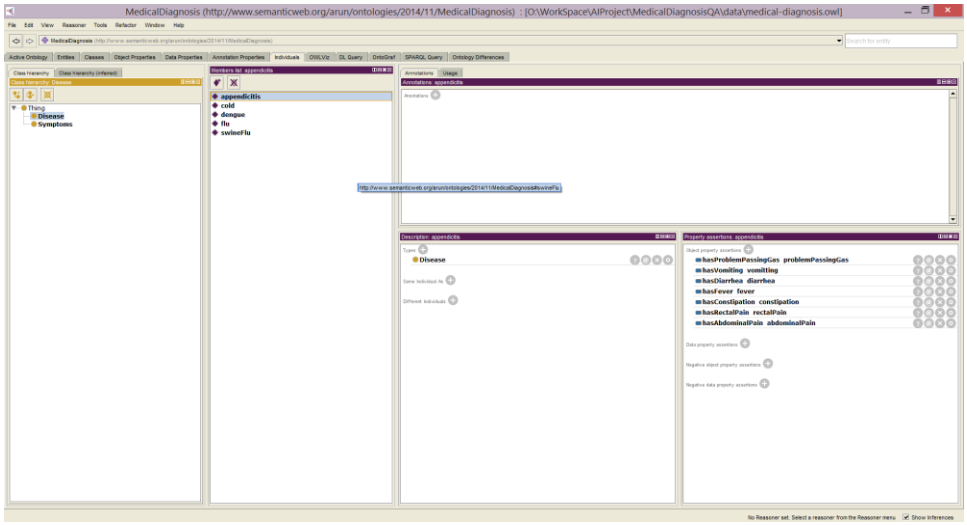


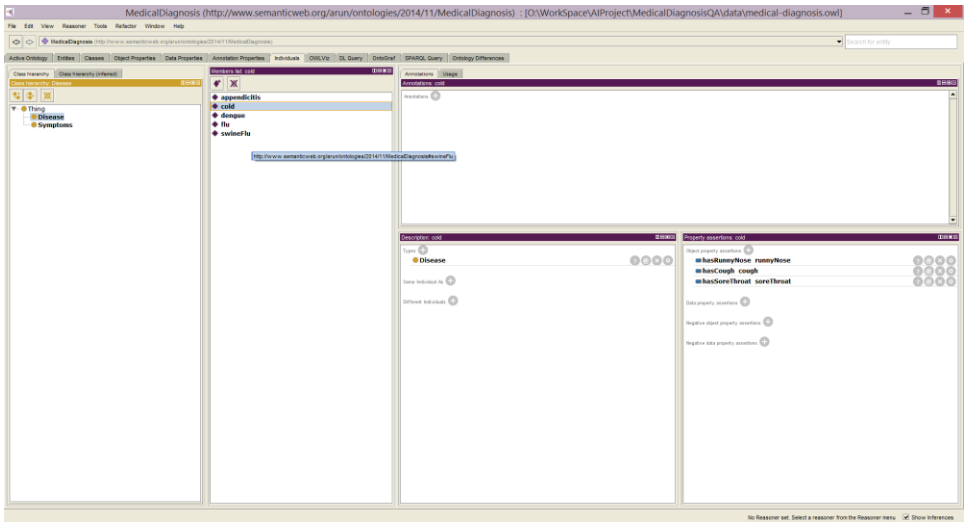
Object Properties in OWL

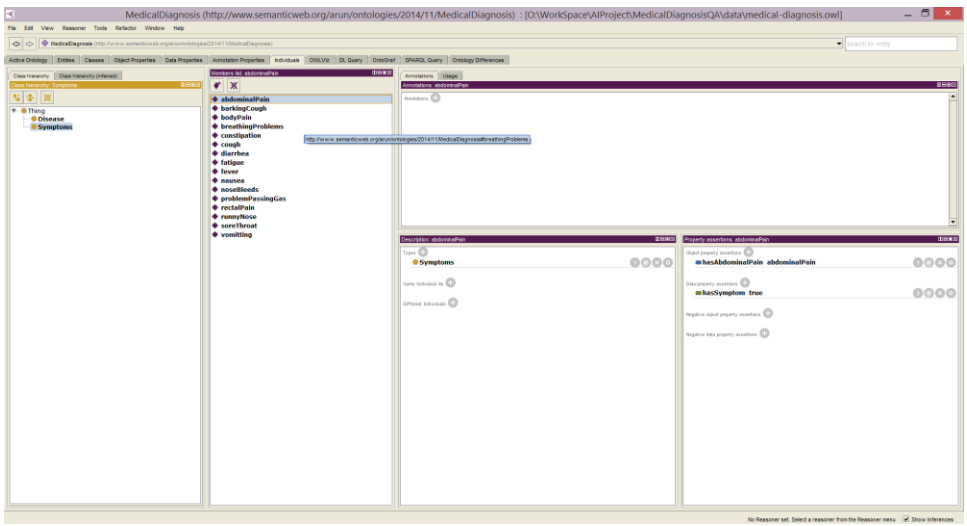


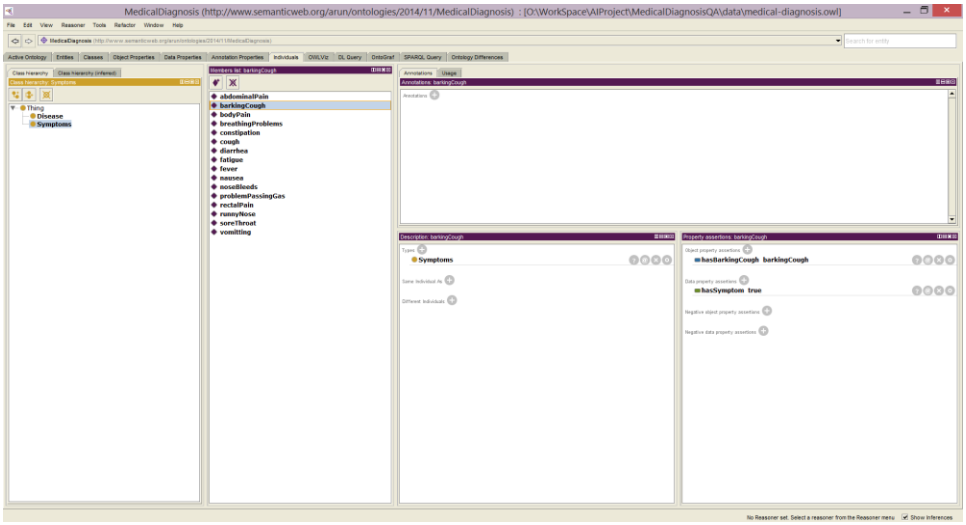


Data Properties in OWL


MedicalDiagnosis (http://www.semanticweb.org/arun/ontologies/2014/11/MedicalDiagnosis) : [O:\WorkSpace\AIProject\MedicalDiagnosisQA\data\medical-diagnosis.owl]
appendicitis
cold
dengue
flu
swineFlu
Disease
hasProblemPassingGas problemPassingGas
hasVomiting vomiting
hasDiarrhea diarrhea
hasFever fever
hasConstipation constipation
hasRectalPain rectalPain
hasAbdominalPain abdominalPain


MedicalDiagnosis (http://www.semanticweb.org/arun/ontologies/2014/11/MedicalDiagnosis) : [O:\WorkSpace\AIProject\MedicalDiagnosisQA\data\medical-diagnosis.owl]
appendicitis
cold
dengue
flu
swineFlu
Disease
hasRunnyNose runnyNose
hasCough cough
hasSoreThroat soreThroat


MedicalDiagnosis (http://www.semanticweb.org/arun/ontologies/2014/11/MedicalDiagnosis) : [O:\WorkSpace\AIProject\MedicalDiagnosisQA\data\medical-diagnosis.owl]
abdominalPain
barkingCough
bodyPain
breathingProblems
constipation
cough
diarrhea
fatigue
fever
nausea
noseBleeds
problemPassingGas
rectalPain
runnyNose
soreThroat
vomitting
Symptoms
hasAbdominalPain abdominalPain
hasSymptom true



Individuals in OWL

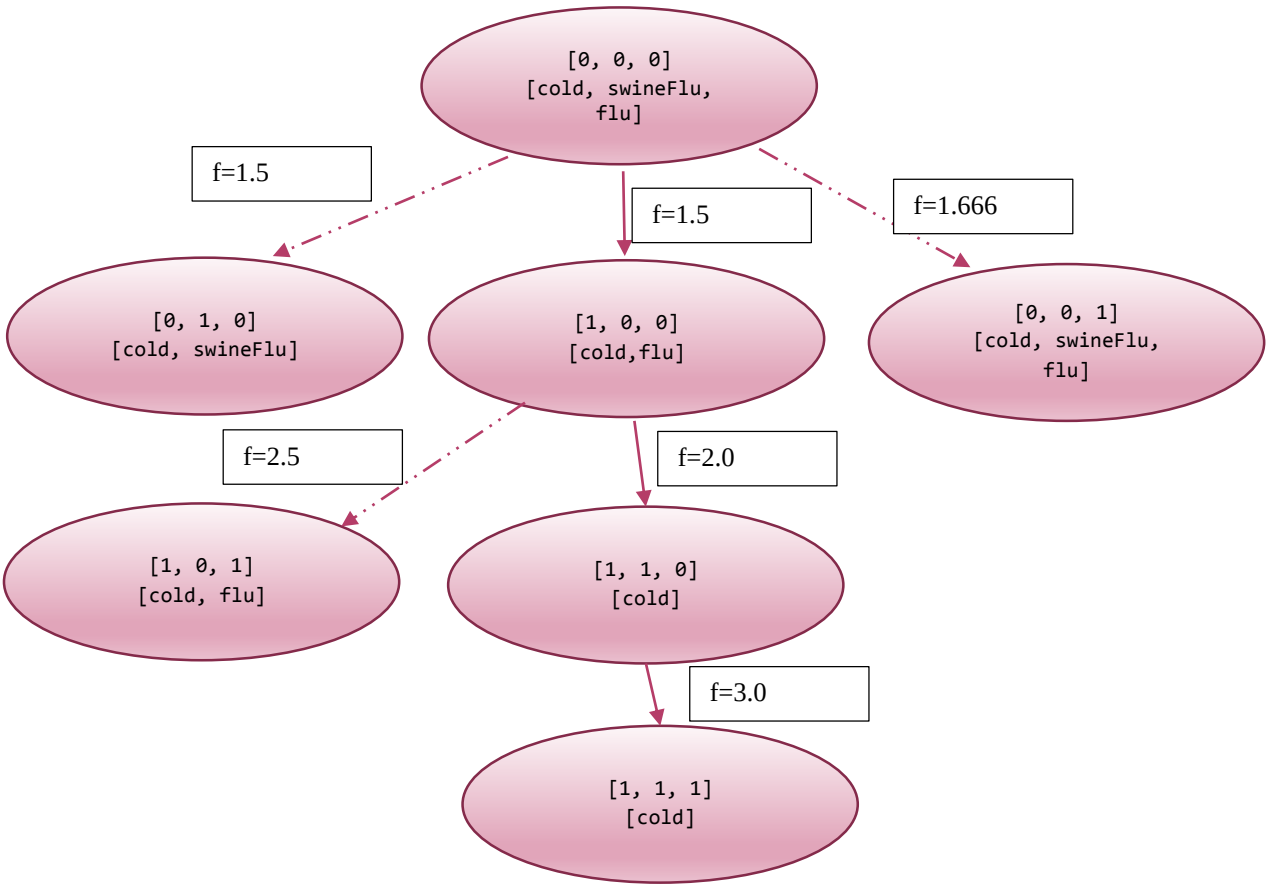
7 RESULTS (IF ANY)

Output after running the program for input *"hasCough,hasRunnyNose,hasSoreThroat":*

```
================================================================
Intial States: [false, false, false]
Intial Disease List: [cold, swineFlu, flu]
================================================================
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Values on the current set of OpenNodes
NodeState: [true, false, false] || Diseases: [cold, flu] || g(n):
1.0 || h(n): 0.5 || f(n)[g(n) + h(n)]: 1.5
NodeState: [false, true, false] || Diseases: [swineFlu, cold] ||
g(n): 1.0 || h(n): 0.5 || f(n)[g(n) + h(n)]: 1.5
NodeState: [false, false, true] || Diseases: [cold, swineFlu,
flu] || g(n): 1.0 || h(n): 0.6666666666666667 || f(n)[g(n) +
h(n)]: 1.6666666666666667
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
================================================================
Current State: [true, false, false]
Current Disease List: [cold, flu]
================================================================
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Values on the current set of OpenNodes
NodeState: [true, true, false] || Diseases: [cold] || g(n): 2.0
|| h(n): 0.0 || f(n)[g(n) + h(n)]: 2.0
NodeState: [true, false, true] || Diseases: [cold, flu] || g(n):
2.0 || h(n): 0.5 || f(n)[g(n) + h(n)]: 2.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
================================================================
```

```
Current State: [true, true, false]
Current Disease List: [cold]
================================================================
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Values on the current set of OpenNodes
NodeState: [true, true, true] || Diseases: [cold] || g(n): 3.0 ||
h(n): 0.0 || f(n)[g(n) + h(n)]: 3.0
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
================================================================
Current State: [true, true, true]
Current Disease List: [cold]
================================================================
================================================================
Final State: [true, true, true]
Input Symptom List: [hasCough, hasRunnyNose, hasSoreThroat]
Final Disease List: [cold]
================================================================
```

State diagram for this example would be:



8 A SUMMARY OF THE PROBLEMS ENCOUNTERED DURING THE PROJECT AND HOW THESE ISSUES WERE RESOLVED

The main problem that I had encountered here was the how to represent the disease and symptoms in OWL ontology in an efficient way. Even now I still believe my proposed ontology can be still improved and optimized for different relationships.

Second problem that I have faced is that my KB is not humongous and is not covering many real life scenarios. I would rather say that this project is just a prototype on how we can use ontologies to describe a problem in an efficient way. Such that the system can learn on its own based on the given ontology and gradually increase and optimize its KB.

9 PENDING ISSUES

Pending issues are as follows:

1. Very small KB
2. Currently able to diagnose only few disease and there are might be a case where the program would not diagnose with any disease as the set of symptoms provided won't be satisfying any relationships in KB

10 POTENTIAL IMPROVEMENTS

Potential improvements for this project would be:

1. Increase the size of KB and optimize all the relationships, so that it can diagnose more and more diseases.
2. Much better heuristic and search algorithm may be used for larger KB