Association of Terms and Phrases in medical field by relative correlation and self evolving data structure

Prepared By :

Pawan Parekh

12MCEC35

Internal Guide

Prof. K.P.Agrawal

Institute Of Technology,

Nirma University

External Guide

Mr. Satish Vagadia

J-KRI TECHLABS

Department Of Computer Science And Engineering

Institute Of Technology,

Nirma University

Ahmedabad

MAY-2014

Association of Terms and Phrases in medical field by relative correlation and self evolving data structure

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

PREPARED BY :

Pawan Parekh

12MCEC35

Internal Guide

Prof. K.P. Agrawal

Institute Of Technology

Nirma University

External Guide

Mr. Satish Vagadia

J-KRI TECHLABS

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD

DECLARATION

This is to certify that,

I, **Pawan Parekh**, **12MCEC35**, a student of semester III Master of Technology in Computer Science Engineering, Institute Of Technology , Nirma University, Ahmedabad , hereby declare that the project work **Association of Terms and Phrases in medical field by relative correlation and self evolving data structure** has been carried out by me under the guidance of Mr. Satish Vagadia , J-KRI TECHLABS, Pune and Prof. K.P. Agrawal, Department of Computer Science and Engineering, Institute Of Technology , Nirma University, Ahmedabad. This Project has been submitted in the partial fulfillment of the requirements for the award of degree Master of Technology (M.Tech.) in Computer Science and Engineering, Nirma University, Ahmedabad during the year 2013 - 2014.

I have not submitted this work in full or part to any other University or Institution for the award of any other degree.

Pawan Parekh(12MCEC35)

CERTIFICATE

This is to certify that the Major Project entitled **Association of Terms and Phrases in medical field by relative correlation and self evolving data structure** submitted by **Pawan Parekh(12MCEC35)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science Engineering of Nirma University of Science and Technology, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, have not been submitted to any other university or institution for award of any degree or diploma.

Mr. Satish Vagadia
External Guide,
J-KRI TECHLABS

Prof. K.P. Agrawal
Internal Guide,
Institute Of Technology,
Nirma University

Prof. Vijay Ukani
PG Coordinator - CSE,
Institute Of Technology,
Nirma University

Dr. Sanjay Garg
HOD - CSE,
Institute Of Technology,
Nirma University

DR. Ketan Kotecha
Director,
Institute Of Technology,
Nirma University

ACKNOWLEDGEMENT

First and foremost, sincere thanks to **Mr. Satish Vagadia** CEO, J-KRI TECHLABS, Pune. I enjoyed his vast knowledge and owe him lots of gratitude for having a profound impact on this report.Throughout the training, he has given me much valuable advice on project work. Without him, this project work would never have been completed.

I would also like to thank **Dr.Viral Bhatt**, J-KRI TECHLABS for his valuable guidance in Medical domain.

I would also like to thank my Internel guide **Prof. K.P. Agrawal**, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance.

I would also like to thank **Dr.Ketan Kotecha**, Director, Institute of Technology, Nirma University, Ahmedabad for providing me an opportunity to get an internship at J-KRI TECHLABS, Pune.

I would like to thank my all faculty members for providing encouragement, exchanging knowledge during my post-graduate program.

I also owe my colleagues in the J-KRI TECHLABS, special thanks for helping me on this path and for making project more enjoyable.

Pawan Parekh(12MCEC35)

Contents

List of Tables

List of Figures

Abstract

Since the emergence of the Internet medical knowledge is spreading around the globe increasingly fast. Though publicly available, it is a difficult task to determine individual relevance for most nonprofessionals. Additionally, relationships between medical terms are hard to discover even for professionals. The proposal is to build a "Clinical decision support system", which will help people to find remedies and treatments based on severity of emergencies in the form of natural language.

My goal in this project is to create self-evolving data structure based on Evolutionary algorithm. This data structure which will store medical data in a way that interlinks data so that terms and phrases will be evolutionary aligned by relative correlation by putting its medical relevancy in center .So that we can create our own raw storage meaningful schema on which users redefined query in the form of machine reasonable and retrieve possible remedies and possible other meaning of users query.

The focus is on creating various graph based structures representing the relatedness between medical symptoms and diseases and studying various graph algorithms and applying them on structures to obtain desired suggestion a user is demanding through search engine. So It will be very much easy to evolve graph based structure as the volume and complexity in connectedness between medical data increases.

Chapter 1

INTRODUCTION : Options for Storing Connected Data

The increasing interrelationship and exponential increase of volume of connected medical data leads to use other than relational database. For many years, developers have tried to settle connected, semi-structured datasets inside relational databases. But as they were created to model forms and tabular structures , they skirmish when modelinf the exceptional relationships for medical world data. The more rise in connectedness converts in the relational database into increased joins, which decrease performance and make it difficult for to evolve or modify an existing database in response to changing business needs.

Most of the non-relational databases are whether key-value, document, or column-orientedstore sets of disconnected documents/values/columns. One strategy is add relationships to such stores is to embed an identifier inside the field belonging to another effective which we call foreign keys. But this requires joining identifiers at the application level, which quickly becomes expensive.

The rejuvenation of the graph database will overcome such limitations for connected data. Graph databases provide a needed structure for storing connected medical data and incorporating a dynamic schema.

Chapter 2

Literature Survey

Medical Nouns , Verbs and adverbs are grouped by Lexical databases of English such as Wordnet and Thesaurus which means into congitive synonyms of semantic conceptual and relations which are lexical. But structure of it will not contain not longer then medical words such as medical search query being passed. A frame will not be lexicalized concept which will not be medical situations. For example, verbs like mobile and cell must be related in the our lexicon to the calling frame. They also does not compute semantic distances among pairs of such medical concepts.[1][2]

Wikipedia was used in earlier time to compute semantic relatedness between concepts from high-dimensional space of concepts in vector model. Also There is one approach which computes same by health graph in which weights of edge were being considered into cooccurence score , which is Text Mining field. During Data -Modeling Procedure of such graph , The situation arrive when same word such as insulin can be considered as words such as gene and drug also , So new approach in which a few levels deep in transitive closure in such a relationship graph can find relationships with a great number of other entities.[3][4]

The heterogeneity conficts which can arrive while aligning medical terms can be divided according to the following abstractions. Schema Conficts: Schema conficts are due to different alternatives provided by one data model to develop schemas for the same reality. Semantic Conficts: Semantic conficts refer to disagreement about the meaning, interpretation use of the same or related data. .[5]

While exploring all available data structures I came to know that the concepts of graph database is the one which can help me to create requisite data structure . Neo4j is popular graph databases which is built on graph data structures which stores data in a graph, the most generic of data structures, capable of elegantly representing any kind of data in a highly accessible way.

Chapter 3

Graph Database Concepts

A graph database stores data in a graph, the most generic of data structures, capable of elegantly representing any kind of data in a highly accessible

A graph database is a databases whose specific purpose is the storage of graph-oriented data structures. Thus it is all about storing data as vertices and edges. By definition, a graph database is any storage solution where connected elements are linked together without using an index. The neighbours of an entity are accessible by dereferencing a physical pointer. There are several types of graphs that can be stored: from a single-type undirected graph to an hypergraph, including of course property graphs.

3.1 Nodes and Relationships

It starts from a single Node , has named valued referred to as Properties , which can grow to millions which is very unlikely in Medical field , At some point it makes sense to distribute data into multiple nodes , symptoms such as Fever and Body Temperature which are organized with expllicit relationships such as "CAUSES" in below example which can also have property as Degree Celsius.

- A Graph -[: $RECORDS\_DATA\_IN$] $\rightarrow$ $Nodes$−[: $WHICH\_HAVE$] $\rightarrow$ $Properties$.

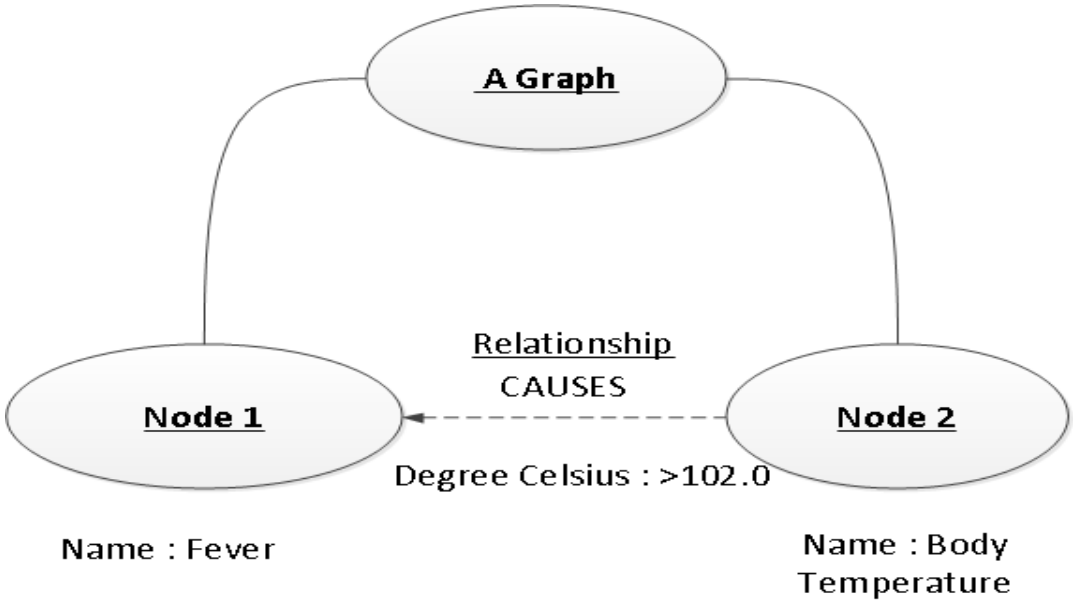

Figure 3.1: Nodes and Relationship
[6]

3.2 Query a Graph with a Traversal

Traversing a Graph means visiting symptoms of Medical field according to some rules. It start from as starting symptoms to Related symptoms according to an algorithms , finding diseases from available symptoms and relationship given between them , In most cases only a Subgraph is visited to obtain answers to given questions. .

- A Traversal - navigates $\rightarrow aGraph.it-identifies \rightarrow Paths-whichorder \rightarrow Nodes$

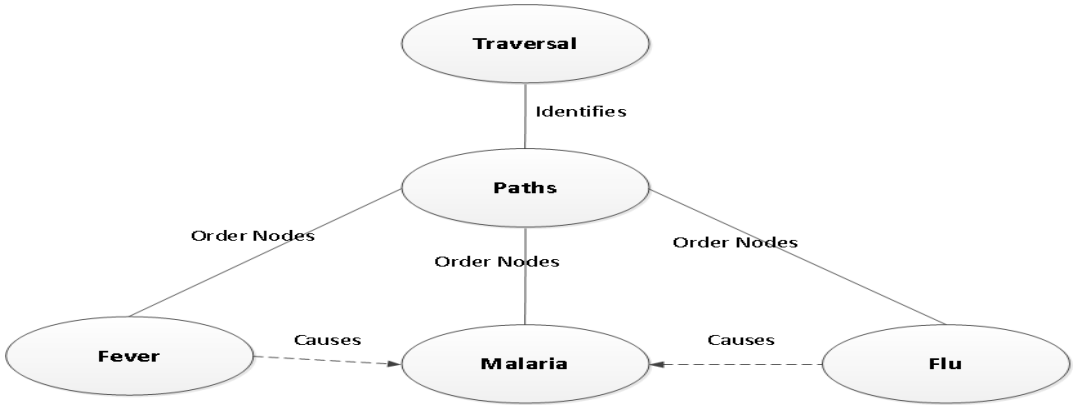

Figure 3.2: Navigation Through Graph
[6]

3.3 Indexes Look-up Nodes or Relationships

Indexes are something which are tied to common proprties of Major medical symptoms or properies of relationship between them . The traversal of medical symptoms to find desired disease can be limited to a subgraph by tarversing in the context of Indexes.

An Index - maps from $\rightarrow Properties - toeither \rightarrow NodesorRelationships$.

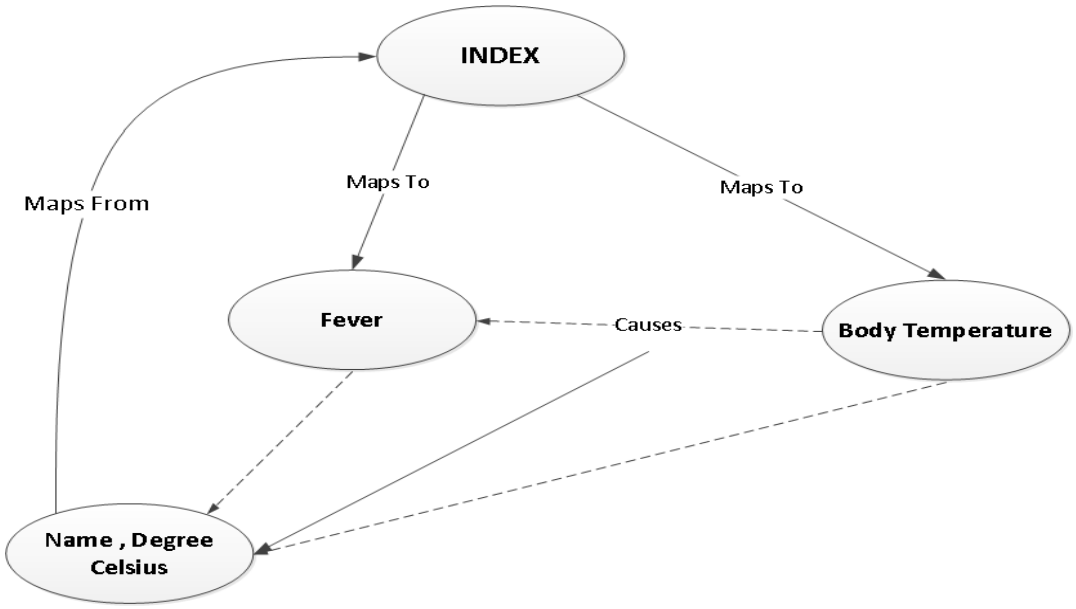

Figure 3.3: Lookups by Indexes
[6]

Chapter 4

Neo4j

Neo4j is an open-source chart database backed by Neo Technology. Neo4j is a chart database permitting putting away information as hubs associated by circular segments. Since information is effortlessly spoken to as charts, gives diagram database more characteristic deliberation for such information than social database. A construction less diagram database additionally permits to effectively include new sorts of information and relations.

4.1 Cypher graph query language

Neo4j gives a question dialect called Cipher, which is a definitive broadly useful diagram inquiry dialect. Figure has been affected by SQL and SPARQL, and it permits expressive and proficient questioning of diagram databases without needing to compose point by point diagram traversals. The fundamental contrast between Cipher and SPARQL is that SPARQL is intended for the area of Semantic Web, though Cipher is a general reason diagram question dialect. The improvement of Cipher began from a need to have less demanding grammar than existing broadly useful diagram question dialects.

START x=node:indexName ($idxPropName =' propValue'$)

$MATCH \quad x - [: relationName] \rightarrow y$

$WHERE \quad (x.property_1 =' value1' OR \quad x.property_1 =' value2') \, AND \quad y.property_2 =' value3'$

$RETURN x, y$

Above illustration displays a sample Cipher inquiry. The question chooses a beginning stage by doing record lookup from file indexname, bringing hubs that have property idxpropname with worth propvalue. At that point the question experiences the brought hubs attempting to match the relationship and whatever is left of the criteria. At last, the question gives back all the hubs x and y that match the example and fulfill the criteria

4.2 Apache Lucene indexing

Neo4j does not give own indexing result, and rather as a default utilizes Apache Lucene3 hunt and indexing library. In Neo4j, Lucene gives intends to indexing hubs and connections. Because of Lucene files, discovering Neo4j hubs by hub properties is to a great degree quick. In test setup, the questions were executed in around 10 milliseconds on normal. Be that as it may, if the records were not used, the inquiries ran more than one extent slower. Subsequently, to addition greatest question execution, Lucene files ought to be completely used.

4.3 Interfaces and operation modes

Neo4j helps three working modes: implanted, server what's more inserted server. In installed mode, the database can just be gotten to by the requisition into which it is inserted. In server mode the database might be gotten to anyplace through the REST API. In implanted server mode the requisition has immediate access to the database, and remote gatherings have admittance through REST API. In the examination setup, installed server mode was picked in light of the fact that it gives quick database access to the provision, and probability to get to the database through a web interface.

Chapter 5

Traversal Algorithm

5.1 Breadth-first search

In graph theory, breadth-first search (BFS) is a used for (a) node visiting and inspecting node of a graph; (b) visiting the nodes which are neighbor of currently visited node. begining from at a root node and inspecting all the nodes which are neighbours of it. This process continues untill all not all nodes are visited.

Sample Cypher Query

START n=node(1)

MATCH p = n-[*1..]→ m

$RETURN p, length(p), last(p) ORDER BY length(p) asc$[5]

5.2 Spreading Activation

Spreading activation is a method for associative networks searching , neural networks searching , or semantic networks searching. The searching is started from first label a set of source (concepts in a semantic) with weights and then it propogates those weights to nodes which are being link to the first node. often these activation are real values that decay as weight propagates through the semantic network. When the activation are different this process is referred to as passing the marker. Weights may originate from one by one paths, identified by different markers, and end when two alternate paths reach the same node..

5.3 Dijkstra's algorithm

This algorithm works as finding th e source vertex node which has lowest cost or we can say shortest path between that node and every other node , This algorithm can also be used for finding traversal time of smallest cose from source node to destination , here we have to stop algorithm once the destination node with shortest time has been found.

Sample Cypher Query

START a=node $(...), b = node(...)$
$MATCHp = a - [r * 2..5] \rightarrow b$
$WHEREnot(a \rightarrow b)$
$WITHp, relationships(p)\, asrcoll$
$RETURNp, reduce(totalTime = 0, xinrcoll : totalTime + x.time)\, astotalTime$
$ORDERBYtotalTime[5]$

5.4 Experimental Results on Algorithms

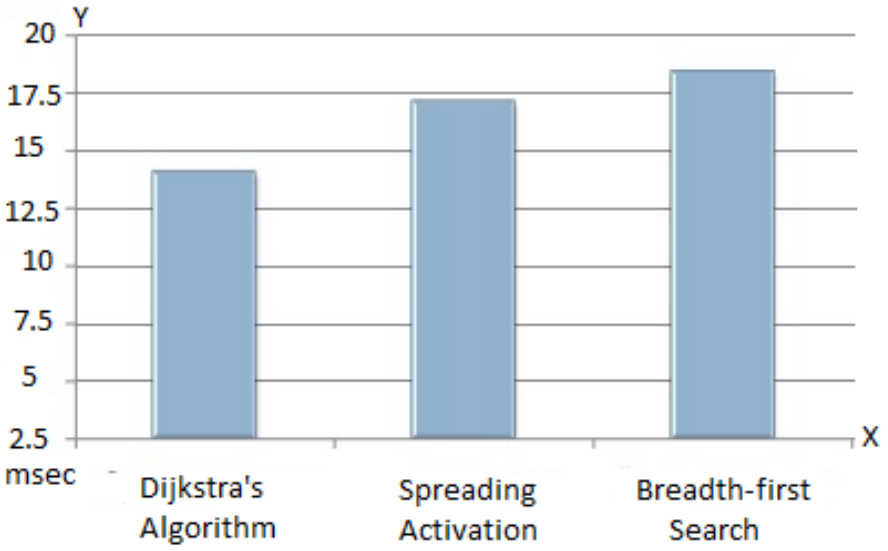

Figure 5.1: Comparison between Traversal ALgorithms [6]

Chapter 6

Status of Implementation

6.1 Main Page

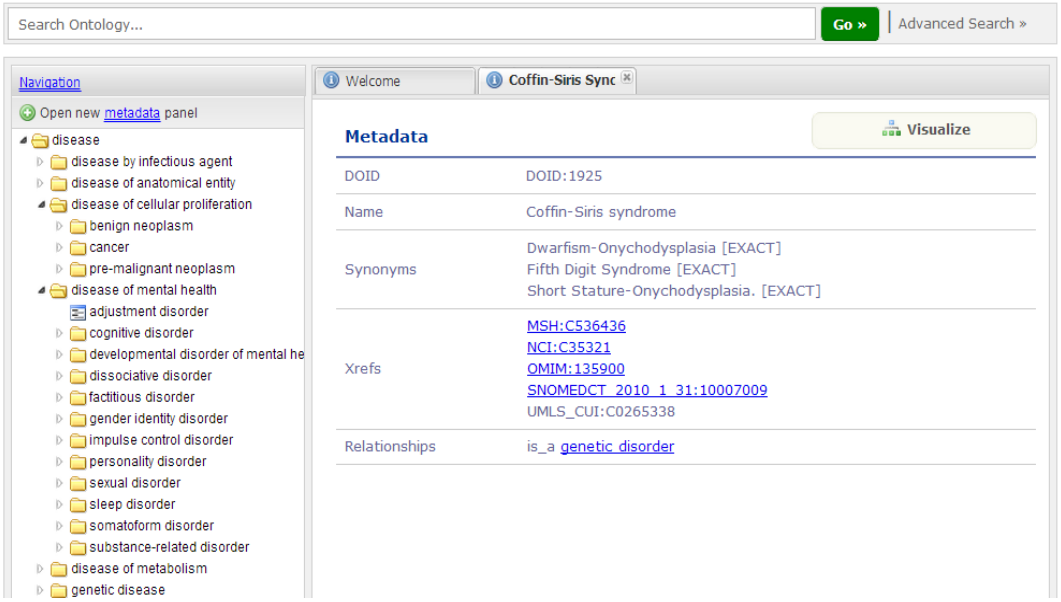

Figure 6.1: Intermediate Results

6.2 Navigation Panel

- The Diseases may be navigated through the use of the Navigation tree found on the page.
- The diseases can be walked from here in a hierachical fashion, moving from relation to relation.

6.3 Search Panel

- Our basic search functionality is as simple as typing in a term to search the disease on and hitting the 'Go' button or pressing the 'Enter' key.
- The basic search field searches over every field in our database.

6.4 Content Panel

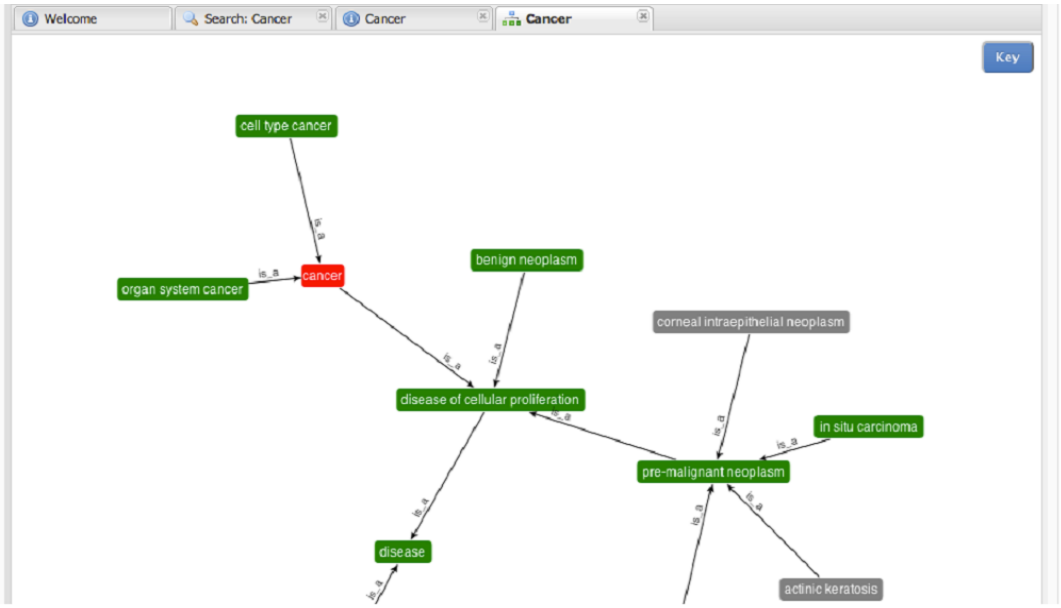

Figure 6.2: Visualization
[6]

Chapter 7

Implementation Methodology

I am using Graph database of Neo4j Technology for managing and querying medical data in graph structure. All medical symptoms are stored in Nodes (records) that has named valued referred to as properties which can grow to millions. The user input will be taken in form of natural language through search engine created in Python programming language. The multilevel relations between symptoms and diseases will be defined by establishing properties on relationship between medical symptoms treated as nodes in here. The output given to user will be suggestion of diseases and possible symptoms which will be retrieved by querying through cypher query which is a graph query language. Cypher query gives output by matching patterns in a graph.

7.1 Algorithm

- Create Nodes and Assign Medical Symptoms as Properties to Nodes.
- Identify relationship between them and add Property to relationships
- Make All nodes Unvisited.
- Identify paths from Cypher Query
- Traverse through graph to obtain Disease

Chapter 8

Intermediate Results

I have created various graph structures in neo4j to represent relationships between medical symptoms. I am displaying here snapshot of sample graph structures.

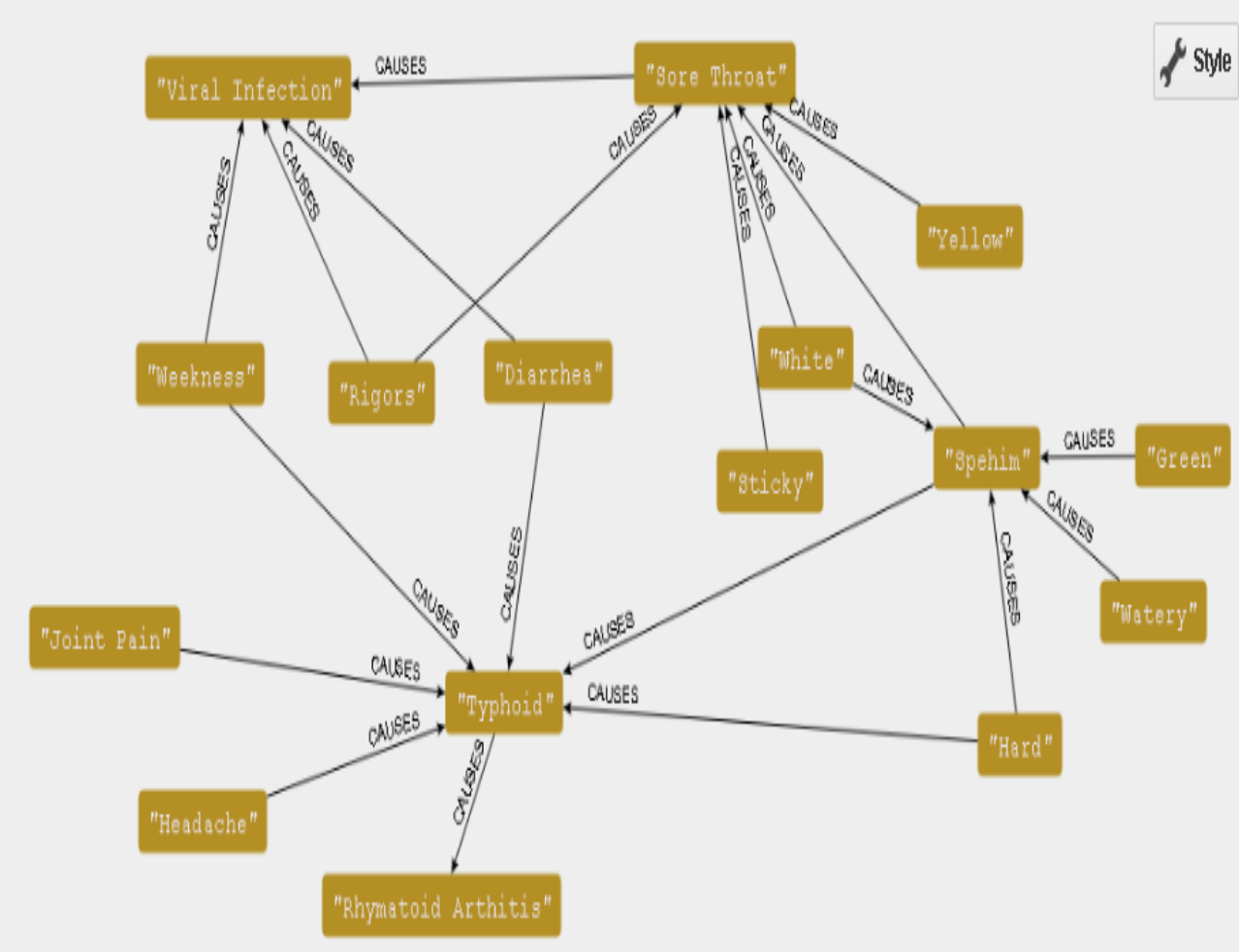

Figure 8.1: Graph Structure for Typhoid Disease
[6][9]

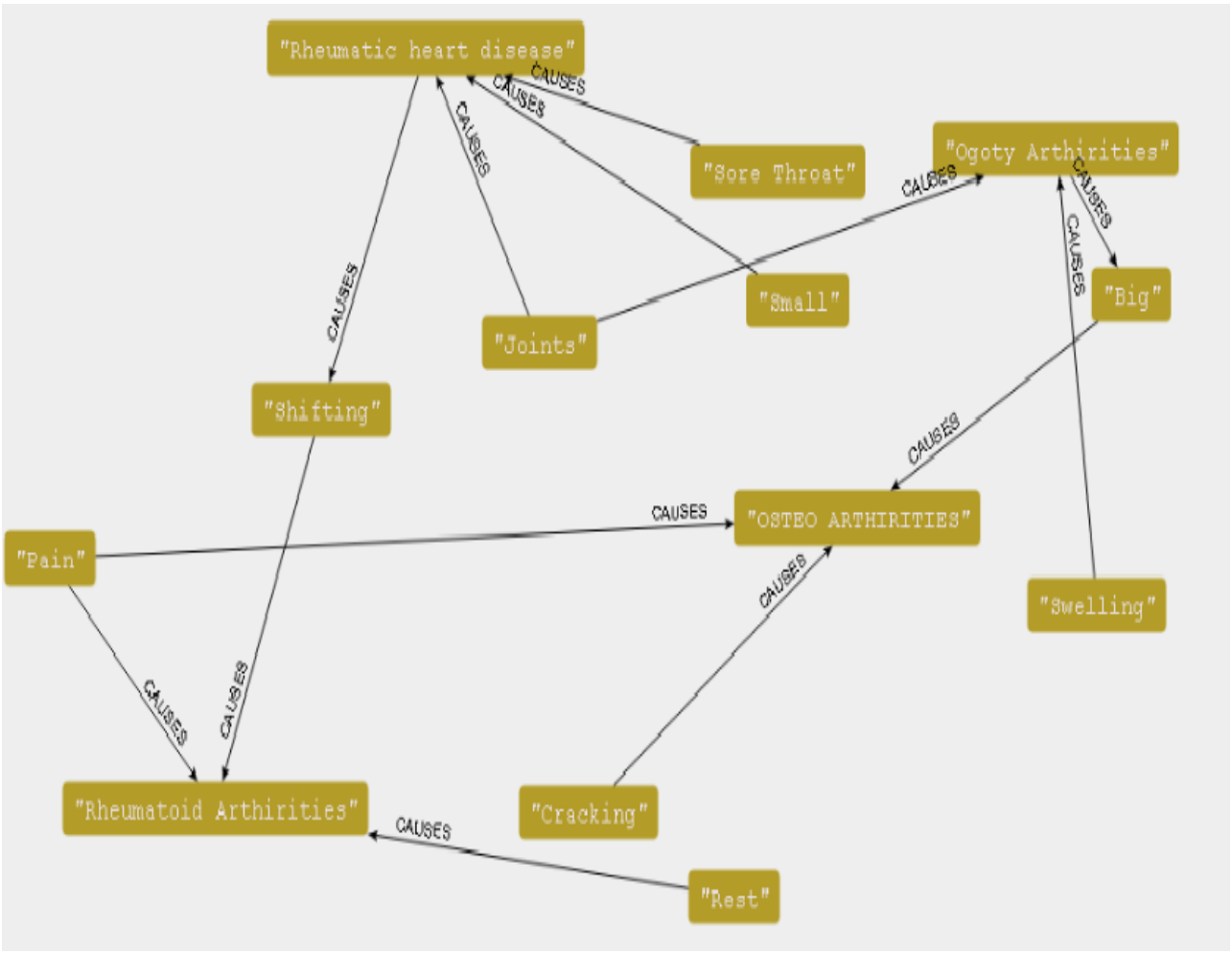

Figure 8.2: Graph Structure for OSTEO ARTHIRITIES
[6][9]

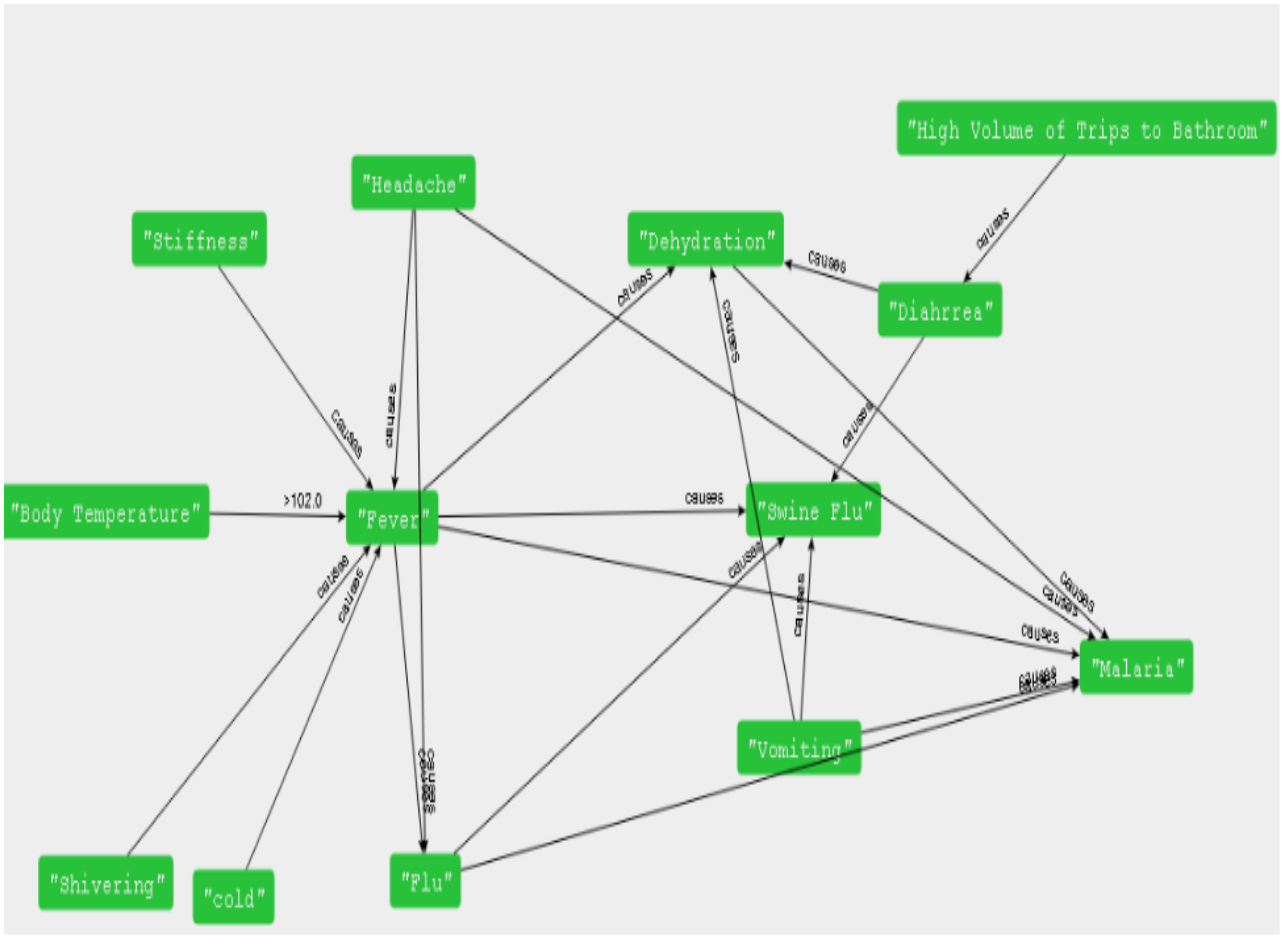

Figure 8.3: Graph Structure of Sample Medical Data for Fever
[6][9]

Table 8.1: Relational Table with "Symptoms" Relationship

Symptom	Disease
Headache	Flu
Headache	Fever
Headache	Malaria
Fever	Flu
Flu	Swine Flu
Shivering	Fever
Flu	Malaria
Fever	Swine Flu
Vomiting	Swine Flu
Stiffness	Fever

If medical symptoms are stored in relational database and if the connection between them are in table format then it waste all the time finding the data which meets criteria for search, The problem is if the data collection is as larger , it will take more time to find matches in the set of data , as here database has not to inspect every data in the collection.

For Example In order to see if symptom "Shivering" causes "Malaria", the database has to find all the diseases because of "Shivering", and see if "Malaria" is in that list. If not, find diseases by it, and then see if "Malaria" is in that list. The full table has to be scanned each time , if you double rows number in table , content of data to iterate will double the time which is to find what you are looking for.(In indexing also it will find index tree values in which traverval time of the tree will be involved , and as new record adds the tree for indexes will grow larger , as we have to start from root of the tree it wil also take more time to traverse the tree)

we have to look entirely new table/index each time we look at new batch of diseases. so search time will increase if there are more symptoms.

Iteratively, a graph database looks only at symptoms that are directly connected to other symptoms. If it is given a limit on how many "symptoms" it is allowed to make, it can ignore everything more than that number of symptoms away.

So the reason why having 1,0000 vs. 1,000,0000 records causes such difference between

Table 8.2: STATS OF PERFORMANCE OF NEO4J OVER RELATIONAL DATABASE

	O.S.	CPU	MEMORY	NODES / Rows	LOOPS	QUERY TIME
Relational Databases	Linux	Core I3	2 GB	20	100	2000 ms
Neo4j Graph Database	Linux	Core I3	2 GB	20	100	15 ms
Relational Databases	Linux	Core I3	2 GB	50	200	5000ms
Neo4j Graph Database	Linux	Core I3	2 GB	50	200	20ms

a relational and a graph database is that if the number of records in relational database increases then performance is being decreased , while graph database performance depends on amount of connections between the records.

There are various cypher queries I have tried upon above graph structure to obtain results such as to obtain particular disease by firing symptoms in query or by mentioning the ratio of symptoms. I am writing down here some sample queries to demonstrate the use of Cypher Queries.

- **This query retrieves intersection of causes by fever and flu**

  `START flu=node` $(*), fever = node(*)$

  $MATCH\,(flu) - [:causes] \rightarrow (symptoms) \leftarrow [:causes] - (fever)$

  $WHEREflu.Name = "Flu" ANDfever.Name = "Fever"$

  $RETURNDISTINCTsymptoms.Name$[7]

- **This query retrieves nodes based on relationship properties**

  `START bodyTemp=node` $(*), disease = node(*)$

  $MATCH\,(bodyTemp) - [r:CAUSES] \rightarrow disease$

  $WHERER.low < 103ANDr.high > 103$

  $RETURNdisease$[7]

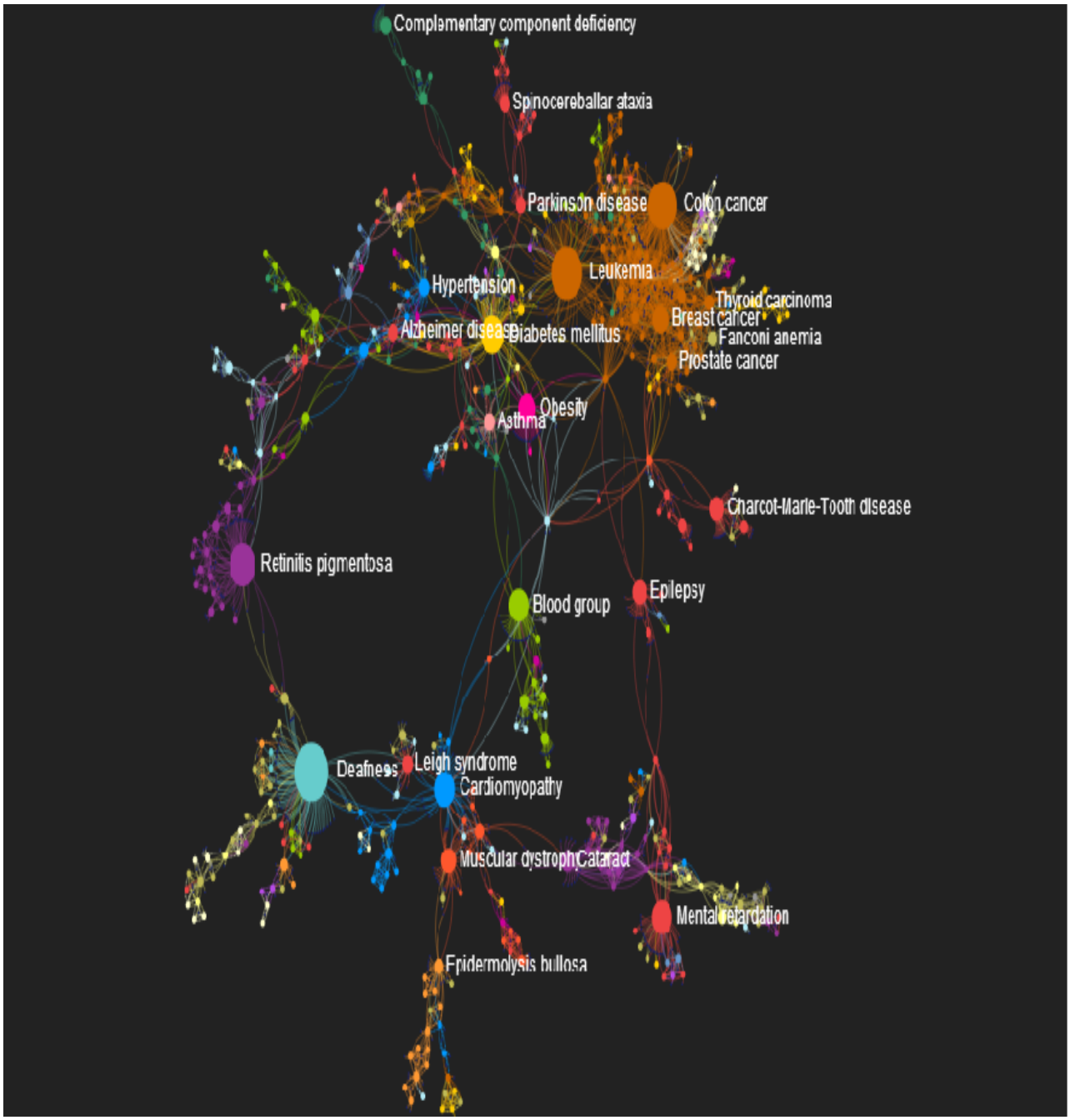

Figure 8.4: Graph Structure of Sample Medical Data
[6][9]

Chapter 9

Conclusion and Extension of Present Work

This report explains the benefits of Graph database over Relation databases in the applications where Data are very much in connected form with each other. This report also demonstrates the concepts behind graph database and key benefits of using Neo4j Graph database.It also explains various experimental results of graph algorithms on medical symptoms.

Future Work

- Considering sentence as a search parameter instead of just medical disease word to retrieve disease as result
- Taking Location , age and gender as search parameters
- Using visualisation tools such as Neoclipse application to interact with Neo4j .

Bibliography

[1] Wordnet : http://wordnet.princeton.edu/

[2] Thesaurus : http://thesaurus.com/

[3] G. Salton, A. Wong, and C. S. Yang. "A vector space model for automatic indexing" , 1975 Communications of the ACM , Volume 18 Issue 11

[4] Martin Wiesner, Stefan Rotter, Daniel Pfeifer . "Leveraging Semantic Networks for Personalized Content in Health Recommender Systems" ,2011 IEEE 24th International Symposium on Computer-Based Medical Systems

[5] Kei-Hoi Cheung, Andrew K. Smith, Kevin Y.L. Yip, Christopher J.O. Baker, and Mark B. Gerstein . "Semantic web approach to database integration in the life sciences" Book , 2007

[6] Neo4j Graph Database : "http://www.neo4j.org"

[7] Neo4j Cypher queyy language Manual

[8] Graph Database vs Relational Database "http://blog.octo.com/en/graph-databases-an-overview/"

[9] Webmd Search "http://search.webmd.com/"