

Ατομική Διπλωματική Εργασία

Implementation of a Cognitive Travel Assistant

Tsinontas Nicholas

UNIVERSITY OF CYPRUS



COMPUTER SCIENCE CLASS

Μάιος 2017

UNIVERSITY OF CYPRUS

COMPUTER SCIENCE CLASS

Implementation of a Cognitive Travel Assistant

Tsinontas Nicholas

Supervising Lecturer

Antonis Kakas

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των
απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του
Πανεπιστημίου Κύπρου

Μάιος 2017

Acknowledgments

I am grateful to Google, for if it was not for its amazing developers my thesis would be impossible to complete. Google is mentioned all over my thesis and is even used in the implementation. There are people that I might be grateful for but the only thing I believe should be written here is for Google. Thank you.

Abstract

This Thesis is a research for the development of cognitive assistants. There is facts and opinions about the current state of today's cognitive assistants and how they are made. In addition, we get in depth on some algorithms to show just how a machine can learn. It is presented how some algorithms are more complicated than others and then which algorithm this thesis is working on.

Finally, the implementation of our own assistant is shown as well as the architectural design behind it. It is explained how each function works and for what purposes it might be developed.

To end, some user case scenarios are shown to explain and evaluate the final product. In conclusion, the project is criticised and given examples of how it could be developed further in the future.

Η ΑΔΕ αυτή είναι μια έρευνα για την ανάπτυξη ενός συστήματος γνωστικού βοηθού. Υπάρχουν γεγονότα και απόψεις για τα υπάρχον συστήματα και πως είναι φτιαγμένα. Επίσης, μπαίνουμε εις βάθος σε κάποιους αλγόριθμους έτσι ώστε να δείξουμε πως μπορεί να μάθει μια μηχανή. Παρουσιάζονται πως κάποιοι αλγόριθμοι είναι πιο περίπλοκοι από άλλους και με ποιους αλγορίθμους δουλεύει αυτή η ΑΔΕ.

Έπειτα δείχνουμε το πώς είναι φτιαγμένος ο δικός μας ο βοηθός όπως και η αρχιτεκτονική του. Εξηγείται πως δουλεύει ο κώδικας και για ποιους λόγους αναπτύχθηκε.

Στο τέλος, δείχνονται μερικά σενάρια χρηστών για να εξηγήσουν και να αξιολογήσουν το σύστημα. Συμπερασματικά, το πρόγραμμα κριτικάρετε και δίνονται εξηγήσεις στο πως μπορεί να εξελιχθεί στο μέλλον.

Contents

Chapter 1	Introduction.....	1
I.	Purpose of Thesis	1
II.	Description of the general problem	1
III.	Goals and Approach	2
Chapter 2	State of the Art.....	3
I.	Existing Systems	3
II.	Methods that are being used	5
Chapter 3	Customization.....	8
I.	Customization in the Modern Day	8
Chapter 4	Analysis-Design.....	10
I.	Architecture	10
II.	Interface	13
Chapter 5	Implementation.....	15
I.	Client Side	15
i.	General Execution	15
ii.	Personalised Execution	15
iii.	Input Processing	16
iv.	Result Processing	16
II.	Sever Side	16
III.	User's Manual	20
IV.	System Requirements	24
V.	Technologies	24
Chapter 6	Evaluation.....	26
I.	User Case Scenario	26
1.	Scenario 1	26
2.	Scenario 2	31
3.	Scenario 3	35
Chapter 7	Conclusion.....	41

I.	Strengths	41
II.	Weaknesses	42
III.	Future Work	42
Chapter 8 References		44
Appendix A Common Words		45
Appendix B DictionaryJSON		46
Appendix C ContradictionJSON		49
Appendix D LocationJSON		50
Appendix E Location Abbreviations		53

Chapter 1

Introduction

-
- I. Purpose of Thesis
 - II. Description of the general problem
 - III. Goals and approach
-

I. Purpose of Thesis

The purpose of the Thesis is the development of a cognitive assistant capable of helping a user at least in a single everyday task. We have chosen to build a trip assistant for it is a simple matter where we could implement the idea of a cognitive assistant. The point is to create a machine that can help someone find a place to go according to his personalised preferences. The user must be able to talk to the assistant as if he were talking to any other person and the assistant must have the ability to understand the context the user is trying to create.

II. Description of the general problem

In the world we live in machines are capable of incredible feats. They are programmed to learn and mimic, operate at microscopic levels and calculate immense amounts of data. Usually such machines are controlled by trained professionals while the common person struggles with understanding these scary and god like machines. What we need are machines that have the capacity to understand anyone, using any kind of dialect or language. Google is a company that is ahead in this matter and that is why we are using the help of Google to achieve our goals.

Due to extreme acceleration in the progress of technology many people are being left behind in the dust of technological evolution. Such people find it hard to catch up as the progress is exponential. This is why we need technology to understand human communications and adapt to it. Once that is achieved humans of all classes and education will be able to create an easier lifestyle for themselves. In the end anyone would program machines to their own liking making each life as easy and enjoyable as possible.

III. Goals and approach

Many programmers have played around with the idea of a cognitive assistant. Everyone dreams for a world where the machines are able to understand our every needs without us having to explain in their language what we want. Some use sample based learning techniques while others use preference learning languages such as prolog. We are going to use both technologies.

We collect a users preferences to assist the users input, and turn it to a closer version to the language the machine understands. It is enhanced also by words that would make it get a stronger meaning and then transformed in an advanced query for Google. The reason for that is, Google is the most advanced technology in human – machine communications. It uses a huge sample pool of queries plus it crawls through each page's metadata faster than any other search engine. In connection with booking.com, google returns to us the most suitable results, which we present to the user in a way that he understands.

Chapter 2

Sate of the Art

-
- I. Existing Systems
 - II. Methods that are being used
-

I. Existing Systems

As it was already pointed out lots of programmers play around with the idea of cognitive assistants. A more professional assistant is “Havyn, a cybersecurity assistant”[1]. It is a program that helps cybersecurity professionals to do their work in an augmented environment. The idea is that these kinds of professionals have too many aspects to look out for, so maybe a programmed assistant could help them spot weaknesses in their defences, and personalise itself with each company and their systems.

There is also a research being conducted to create a search engine for your memories. An inventor in IBM has already patented a cognitive assistant technology that learns about you and has the ability to remind you of simple every day information that slip your mind, for example, what is the name you forgot or where you put your keys and such trivial but annoying questions. He claims that it is a simple idea. “You monitor an individual's context, whether it’s what they’re saying or what they’re doing ... and you predict what comes next”[2]. He faces opposition, however, from some of his colleagues, who say that the human mind is unlike a computer brain and that we do not have pointers and addresses to look up information that we need.

Among the companies that work with AI, Slack is one of them and also it

is building a Cognitive Assistant for their app. Slack is a Conversation tool where you can have group chats and meetings, schedules and important business briefings. A researcher in IBM believes it is a good idea to create a cognitive agent to help users organise their schedules through experience learning.

Having said that, this project is named Watson and it would be able to be accessed by anyone and create other new powerful tools or simply use the already existing ones. “The first step, is combining Watson’s cognitive computing power with Slack’s digital workplace capabilities, starting with a Slack adoption of Watson Conversation for Slackbot. Once integrated, the bot will centralize communication and trouble-shooting in one channel. As the bot gathers data, it can more quickly and accurately address similar issues over time due to Watson’s machine learning technology”[4]. Therefore, this a great illustration of cognitive assistant usage. Moreover, Slack is being used by millions worldwide and a bot that learns to efficiently manage issues will be immensely valuable. “Over time, the value keeps growing. Every time an employee in your organization teaches Watson something, that learning is available throughout the system”[4]. A true case of “Knowledge is Power”. What's more, by feeding knowledge to the cognitive assistant it can then be used to help every other user or organization. Thus, this creates a community that helps itself, resulting in Watson having more value and reputation. “Shortly, IBM will release a Botkit Watson middleware plugin, that will allow the Watson Conversation service to talk to Slack. The companies will also release an Application Starter Kit for developers providing code and step-by-step instructions for integrating Watson on the Slack platform”[4].

Despite the most known projects on cognitive assistants, there are some less known ones and there are many of them. To give an illustration, Marcel Just developed “a cognitive architecture whose models can account for both traditional behavioural data and, more interestingly, the results of neuroimaging studies” [3]. In addition, Marcel Just's project on cognitive assistants is significantly important to help develop new technologies to be used in neurosciences. More specifically, 4CAPS project “is a hybrid architecture that

combines symbolic and connectionist mechanisms in a resource-constrained environment” [3] .

Furthermore we have SRI's project named CALO (Cognitive Assistant that Learns and Organizes). “The goal of the project is to create cognitive software systems, that is, systems that can reason, learn from experience, be told what to do, explain what they are doing, reflect on their experience, and respond robustly to surprise”[5]. This system will collaborate with its users while they admonish its performance. It will try to antagonise tasks that seemed impossible to be automatize. The researchers of this system were using it while it was still being developed. This was done to ensure that it satisfies the needs for privacy, security and trust.

We also see them every day while we surf the internet. Every time we are presented with advertisement, it goes through the decision of a cognitive assistant that decides what we would like to see. This happens automatically without us noticing. This might not seem like something we care about but it could prove useful.

Another key thing to remember, the area which was researched in the thesis mainly deals with some systems. However, these specific systems are not necessarily cognitive ones. To give an illustration, Trip Advisor is one most used websites among travellers. Most specifically, these particular non cognitive systems can be used to find locations to go on vacations alone or with company. Importantly, these are applications to help you find destinations but they are not personalised. In addition, if these applications are personalised, they are done by fixed programming and not by general and specific preferences. The system developed in the thesis combines fixed word connections with free speech and personalised data.

II. Methods that are being used

One method used for cognitive assistants is supervised learning.

Furthermore, “this algorithm consist of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these set of variables, we generate a function that map inputs to desired outputs. The training process continues until the model achieves a desired level of accuracy on the training data”[6]. All things concluded, supervised learning is used when we know the result that we desire thus being able to map the given inputs.

Another important method used is called unsupervised learning. More specifically, with “this algorithm, we do not have any target variable to predict. It is used for clustering population in different groups, which is widely used for segmenting customers in different groups for specific intervention”[6]. Hence, this is suitable for real world issues where we can't always predict the outcome, meaning that we can't supervise this kind of learning.

Reinforcement is a learning method of cognitive assistants. Moreover, by “using this algorithm, the machine is trained to make specific decisions. It works by exposing the machine to an environment where it trains itself continually using trial and error. This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions”[6]. Therefore, reinforcement learning has the ability to teach a system to collect past knowledge in order to develop better and more efficient decisions.

One of the simplest algorithms is a type of supervised learning algorithm “that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables. In this algorithm, we split the population into two or more homogeneous sets. This is done based on most significant independent variables to make as distinct groups as possible”[6]. This is learning is in the form of a tree, where learning comes through predefined properties which then help the machine calculate mathematically which decision to take.

The method we are using is Rule based decision making. The idea is

simple. You create a general rule. It has to have some attributes that define it. Then you initialize rules that might contradict the general rule and you set the level of priority. Some rules are allowed to coexist however they will still be of different priority. This kind of programming is used in logical programming and can be found in the Gorgias plug-in of Prolog.

In my opinion the best way to create an Artificial Intelligent System is by sample learning. An example of sample learning to reach sentient intellect is the human evolution. As we theorise we have a common ape ancestor. A certain lineage of that ancestors begun collecting clusters of knowledge by observing the environment and comparing it to other samples. Then that generation taught it to their descendants whom also gathered sampled information thus reaching the technologically advanced human kind we are to this day.

Chapter 3

Customization

I. Customization in the Modern Day

I. Customization in the Modern Day

As technology moves forward, machines get more and more personal. Nowadays everything on the internet collects data from the users to create a feeling of familiarity. Every search we make, tries to match our personal needs before presenting answers to us. However most of these assistants seem broken.

The reason for that is pretty simple. There is a term called “Data Gap”[0]. What that represents is the limitation in information gathering of the machines. A machine can “understand” your needs, based on its own terms. A system can only collect information about you through your feedback and it is very limited. So this “understanding” of your needs is only partial.

Another term that explains this is called “Computing Gap”[0]. This term is to describe the limits in computing and processing power. A person's needs are diverse and complex. The human brain is one of the most advanced machines to exist. There are too many variables to account for thus any man-made machine, even the fastest ones, become slow when try to process such extreme numbers of data that is the complexity of a person's needs.

Then we have a term that is called “Action Gap”[0] When you visit for example a site, you have at your disposal a certain amount of tools. Buttons, lists, forms and so forth. However you might want to use something that just is not there or you might want an

action to happen differently. However the functionality of a system cannot cover all possible ways one would want to use that system. Thus the user will not feel completely personal with that system and it will break the illusion.

The next term we will examine is “Interest Gap”[0]. In this day and age the real God is green. Everything is controlled by money and that means your needs fall on a low priority list. Business will pay to have their own content reach the masses, which means that even though you might not want to see something you will get it because someone is paying big money to have it published. That is the reason for all those “personalised” ads we see all the time. Even though we certainly do not want to see them we get bombarded by them because money is what pulls the strings.

Lastly there is a term that is probably the most common cause of unpersonalised personalization, the “Content Gap”[0]. There is a limit to how much information a system can hold. So that system may not be able to present to you the exact information you are looking for. Another case is it might not have a different one to present when you ask a second time. There is a limit on memory a server can have and a limit on the variety of information a server can gather. Thus you might not feel that you get what you need.

These are the most common difficulties in personalization and yet there are even more problems that a system can face.

Chapter 4

Analysis – Design

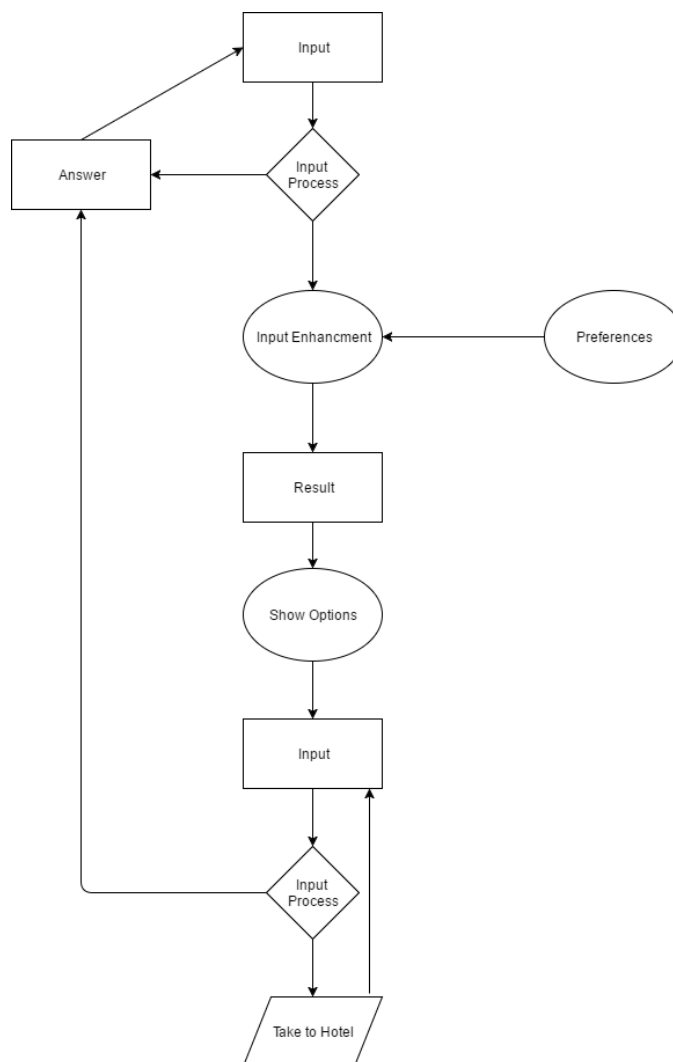
I. Architecture

II. Interface

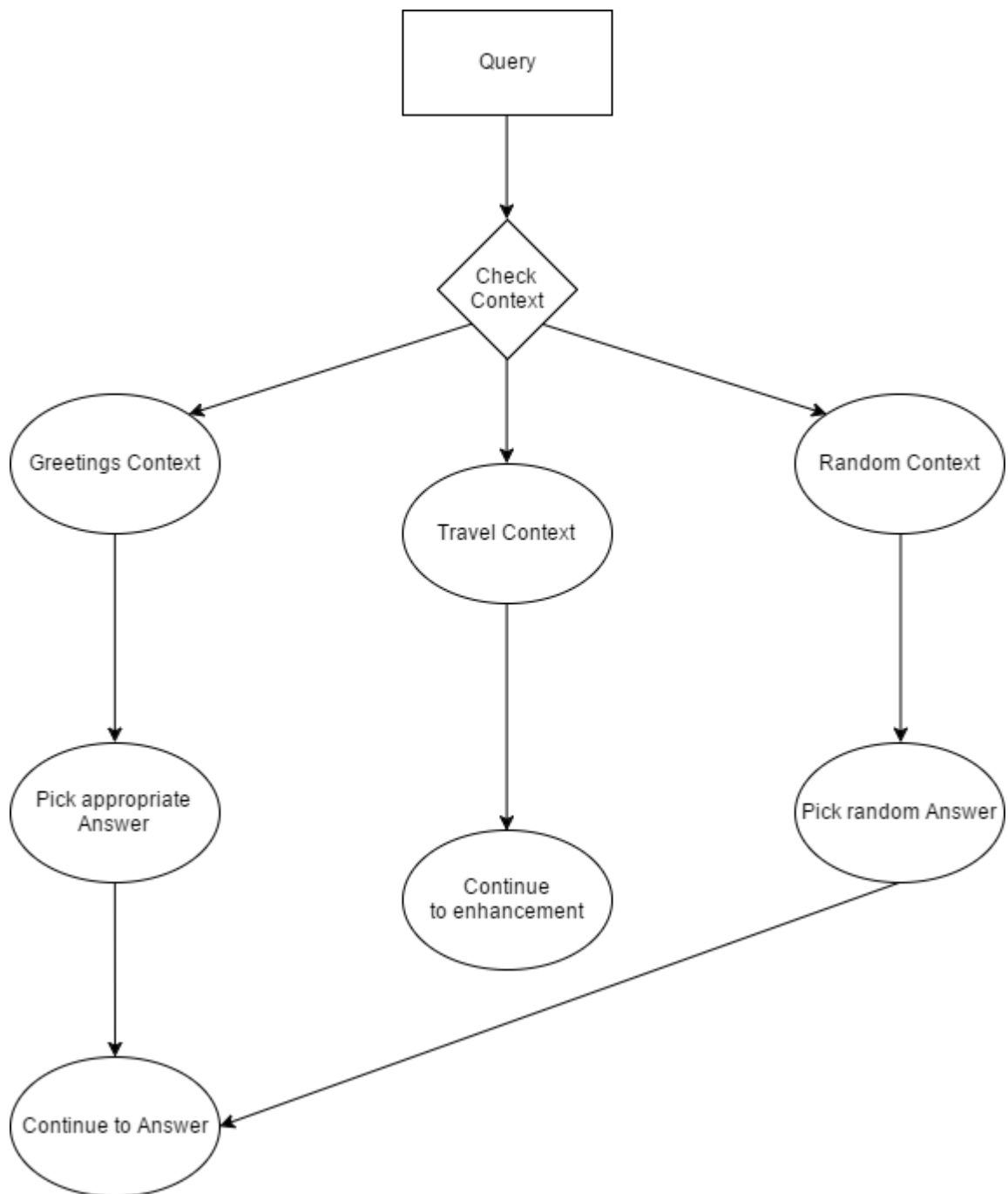
I. Architecture

The architecture is broken down to a few levels.

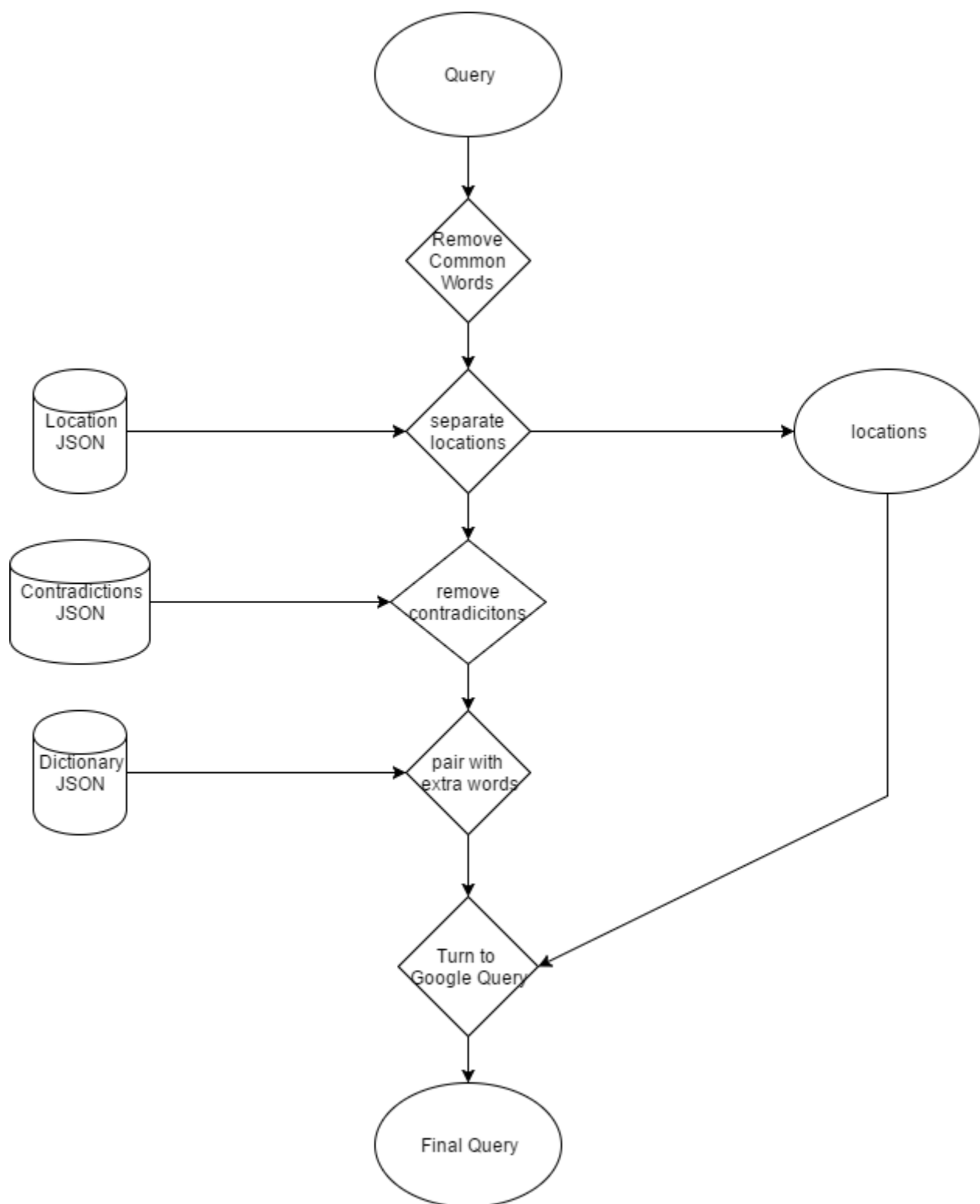
Level 0 is an abstract representation of how the system works.



Then we brake the input processing into steps.



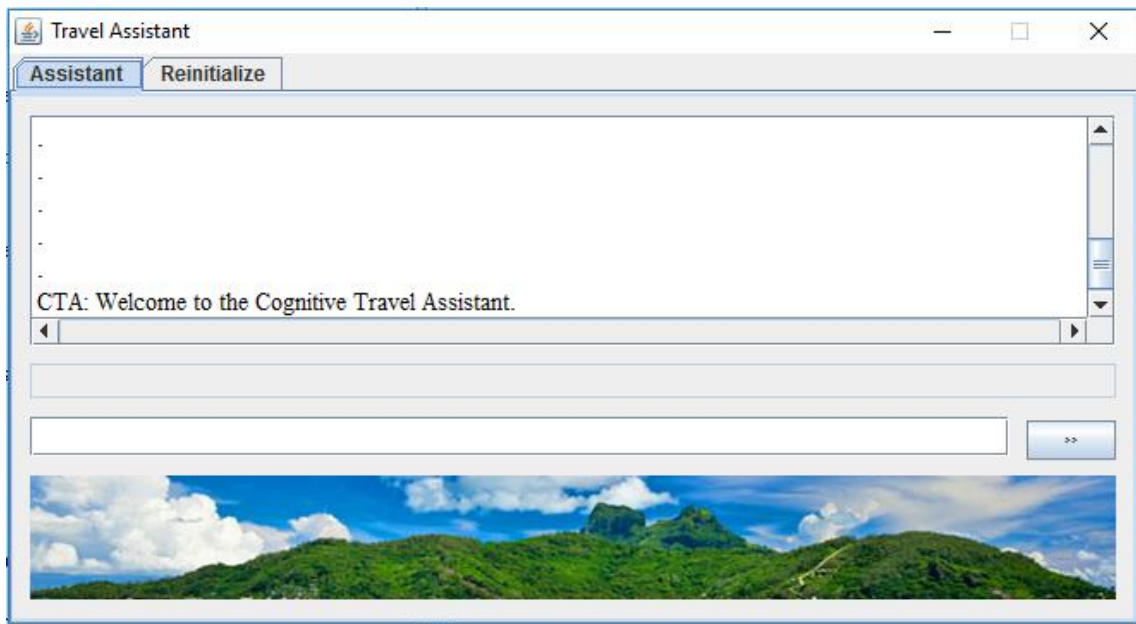
And this is how the server enhances the query



Then Google's answer is presented to the user in a user-friendly way, and he chooses to either pick an option or keep talking to the system.

II. Interface

The interface is built to be as simple as possible to be used by anyone. It is a mere chat box where the user can type any input and the machine will try to answer in a way it finds appropriate. The chat box is a scrollable text area where the user can see the up to now conversation. It is in html form so that we can present the hotels as hyperlinks and turn them into clickable text.



The tab named Reinitialize is where the users can redefine their preferences by just filling up some blank boxes. They put a rule and a preference. They can add more preferences and fill them in any way they see fit. Our machine will try and understand them to the best of its capabilities.

Travel Assistant

Assistant Reinitialize

Usually I like my trips to be	<input type="text"/>		
But if I am with	<input type="text"/>	I want them to be	<input type="text"/>
If I am with	<input type="text"/>	then they must be	<input type="text"/>
On the other hand if	<input type="text"/>	my trips should be	<input type="text"/>
If I am	<input type="text"/>	give me trips that are	<input type="text"/>

Add Pref Save Reset

Chapter 5

Implementation

-
- I. Client Side
 - i. General Execution
 - ii. Personalised Execution
 - iii. Input Processing
 - iv. Result Processing
 - II. Server Side
 - III. User's Manual
 - IV. System Requirements
 - V. Technologies
-

I. Client Side

i. General Execution

The systems can work without any personalization. The reason for this is, some people might not want something different each time. For these scenarios, the system will take the input without adding anything to it and process it. Depending on what words the user uses, then the systems will either give you results or answer back to them.

ii. Personalised Execution

On first use, the system will ask the user various questions so that it can profile them. Through the questions the system will see the user's preferences for different scenarios. For example, if you are with your family what kind of trip would you desire, or if you are with your work then you would want what.

After the first use to recreate a profile you just go to the Reinitialize Tab where you find some blank spots to fill in information that you would want.

iii. **Input Processing**

Before the input is sent to the server, it is processed by the client side code to understand the purpose of the input. If the user is looking for a place to go it is sent to the server. However the user has the ability to communicate with the system and the system will try to understand what the user is saying and answer accordingly. This doesn't have any real purpose but just to add a bit of flavour.

iv. **Result Processing**

The server answers in a form of (keyword OR keyword) AND (location). The system in the client side will take that and add it to the query “site:booking.com/reviews”. That is done in order to check reviews of people for the needs of our user. Then it takes the first three results. It takes the links and rearranges them in order to get the site that takes you to the hotels themselves. I also breaks those links in parts to present the user with the name of the hotel and its location. The location is found by comparing the short names of the locations to a list of locations we have gathered from Google. For example the short name “cy” will return “Cyprus”. The user then chooses which hotels he wants to check and is taken to that side. He can then choose another or keep talking. The chat remains there so if he needs to check another hotel after he resumed talking, he can click on one in the chat for they are in the form of hyperlinks.

II. **Server Side**

The server takes a line of input and processes it. Firstly it removes any common

words to create a list of key words.

```
function removeCommon($query,$commonWords){
    $proQuery=array();
    $j=0;
    for($i=0;$i<count($query);$i++){
        if (strlen($query[$i])>0&&strpos($commonWords, $query[$i]) === false){$proQuery[$j]=$query[$i];$j++;}
    }
    return $proQuery;
}
```

The common words were found from Google's statistics. I removed some that had an impact to our field, but the rest are words like 'I they them me my' etc. For example the sentence “I want to take my partner somewhere romantic” would be turned into “partner romantic”.

Then it would find any words that indicated location.

```
function getLocations($l,$locations){
    $loc="";
    for($i=0;$i<count($l);$i++){
        if ($i<count($l)-1 && isset($locations->{$l[$i]." ".$l[$i+1]}) === true){$loc=$loc.$l[$i]."-".$l[$i+1]." ";$i++;}
        else if (isset($locations->{$l[$i]}) === true){$loc=$loc.$l[$i]." ";}
    }
    return $loc;
}
```

The list of countries were also gathered by Google. The system separates the list into two lists. Locations and none. For example the sentence “partner romantic spain” would be turned into “partner romantic | spain”.

After that, it takes the words and compares them while looking for contradictions.

```

function getContr($c,$contr){
    $flag=0;
    if(count($c)<=0) return "";
    $con="";
    $con.=$c[0];
    for($i=1;$i<count($c);$i++){
        $flag=1;
        for($j=$i-1;$j>=0;$j--){
            $found=0;
            if(isset($contr->{$c[$i]})===false){
                break;
            }else{
                for($k=0;$k<count($contr->{$c[$i]})[$k];$k++){
                    if(strcmp($c[$j],$contr->{$c[$i]})[$k])===0){
                        $flag=0;
                        $found=1;
                        break;
                    }
                }
            }
            if($found===1)break;
        }
        if($flag==1){
            if(count($con)>0)$con.=" ";
            $con.=$c[$i];
        }
    }
    return explode(" ", $con);
}

```

The point is to keep only the prioritised words. Priority is set from the client side, where they check the users preferences, then adding words at the start of the query arranged by most preferable. Thus the server takes priority from left to right and removes and contradictions. The contradictions are found by checking a JSON file that can be enhanced by any one that might be working on this subject.

Next step is to check any words that can be paired with predefined keywords that can enhance the query.

```

function getQR($q,$dictionary){
    if(isset($dictionary->{$q}) === false){return;}
    if($dictionary->{$q}==$q){
        if (strlen($q)>0&&strpos($GLOBALS['queryOut'], $q) === false)
            $GLOBALS['queryOut'].=" ".$q;
        return;
    }
    for($i=0;$i<count($dictionary->{$q});$i++)
        getQR($dictionary->{$q}[$i],$dictionary);
}

```


This words were collected through asking random people what they would search if they wanted to travel. This list, as can the rest, may be expanded by anyone at any given time, since it is not hard coded to the system. For example the sentence “kids cheap” would be turned into “kids cheap family”.

Finally it poly-morphs the sentence in an advanced version of Google queries.

This is done to create the best query so that we have the most fitting results to show.

For example the sentence “partner romantic | spain” would be turned into “(partner OR romantic) AND (spain)”. The Ors are placed because the user might use names or inexistent words in their query so we give Google the ability to ignore them.

Repositories:

1. Common Words

Google has collected statistics on each word that is being used by anyone search in the Google database. I used this collection to be able to process any sentence given to me by a user to keep only important key words so that I can create a context. Words like a, the, my, you and such are used in sentence but have zero impact on the context. Thus I completely ignore them. The collection by Google however had to be modified since it contained words that could give meaning to my context due to the subject of my project. For example the word “family” is a very commonly used word however it plays an immense role to travelling.

2. Location JSON

Unfortunately there is no API, from what I had found, that gives you a simple answer to the question “Is X a location”. That meant that I had to create a crawler that clustered all locations into a JSON file to be accessed by my server in order to separate locations from the rest of the key words. This json will be accessible by anyone so that it may be later used in other projects by anyone that needs a location finder. It will be found at http://*serverIP*/locationJSON.json

3. Contradictions JSON

Due to preferences and query enhancement sometimes sentences might contain contradicting arguments. For example let's suppose that you set that generally you like cheap holidays. Then you set as a specific preference that if it is work related trip you want it to be expensive. Finally you use as input “I want to go on a business trip”. The query will be transformed to “expensive cheap business”. That happens due to the fact that the sentence is checked against your preferences sorted by higher level priority. So since “business” will prompt the word “expensive” that will go to the head of the sentence. Then it will get the general preference and add next to the other “cheap”. Now this has created a contradiction. High priorities start first so I check backwards for contradictions and remove words. If something doesn't exist in my contradiction JSON then it will automatically be placed in the query. Because of this the contradiction JSON will be open to the public to be edited and added to. This will happen in future patches as this one will not have this possibility.

4. Query Enhancement

This is the repository that holds certain words with possible contexts. It will also be an open file so that anyone can add to it. Of course it won't be completely added but there will be an interface for it. A word in this file might be paired to multiple ones. It is visualised in a tree form. But it is used for enhancing from something more specific to something more general. For example the word “kids” adds the word “family”. However there are lots of ways to write a word that is why I want this to be open to anyone to add what they believe is relative. The reason we use this repository is so that we don't become too specific in our search since when someone is thinking about something, they are thinking about it in a more general way. For that we are using both words, the one entered and the one added, so that we can find the most preferable result.

III. User's Manual

The system is built in a way that even someone that doesn't use a computer in his everyday life can actually understand and use it. Since it is heavily relying on text boxes the user will only need to understand that he needs to type in those text boxes, if he cannot understand that that means he has never seen or used a computer before in his life and that is a very small percentage of the human population.

First of all the system wants an introduction. It will ask the user a series of questions on the big chat box. The user can answer accordingly to his understanding. After all questioning is done, it is the users turn to question the machine. The users can ask for anything they want, and can type absolutely everything. The machine will be trying to understand whether you are chatting or asking for recommendations which means the user is free to type anything. Then the user will press either [ENTER] or press the button with the arrow on it.

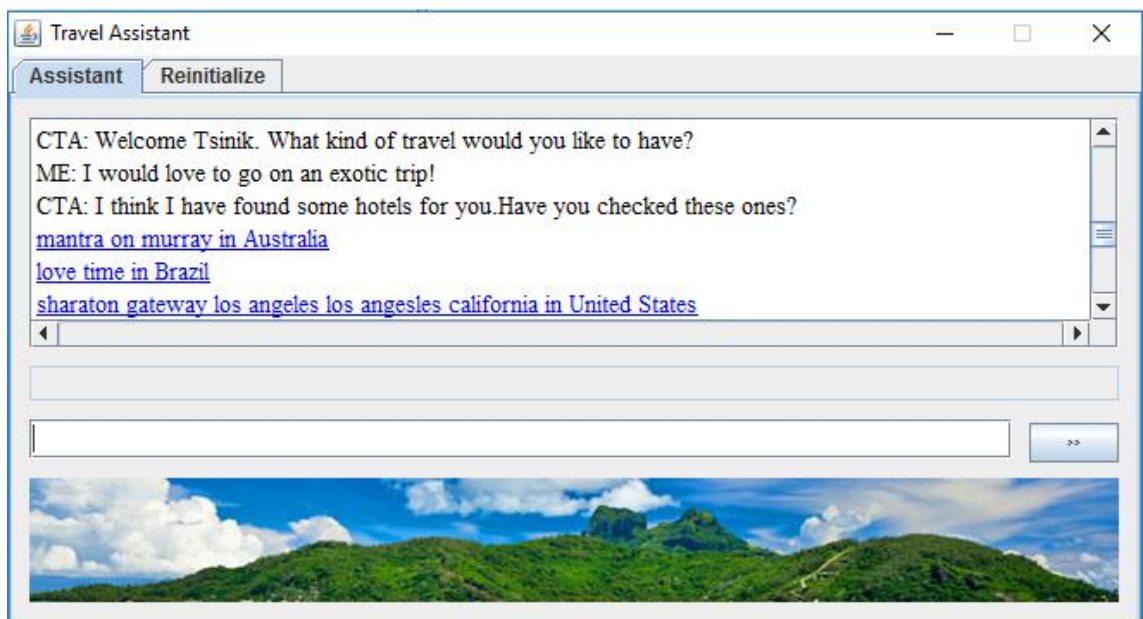
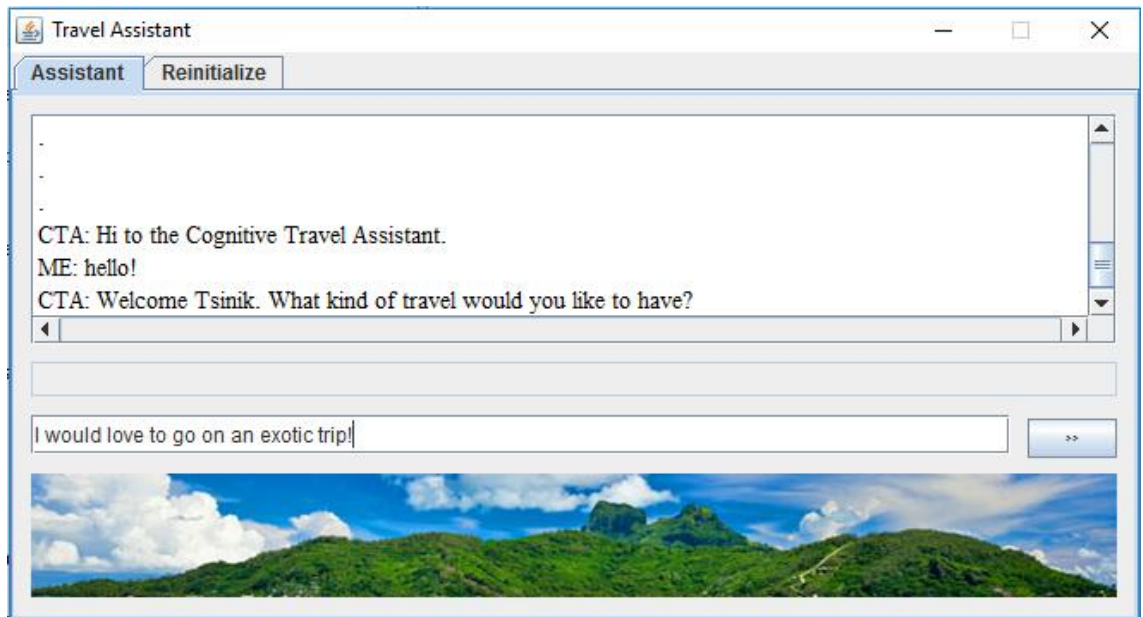
When the user asks for recommendations the machine will present up to 3 hotels that are pulled up from booking.com database. Those links are shown as hyperlinks thus the user can simply click on them and be taken to the site to check out the hotels. Then he is asked if he is happy with those answers. If the user answers in a negative context the machine will ask for further details to bring up other results. Otherwise the machine will wait for further input.

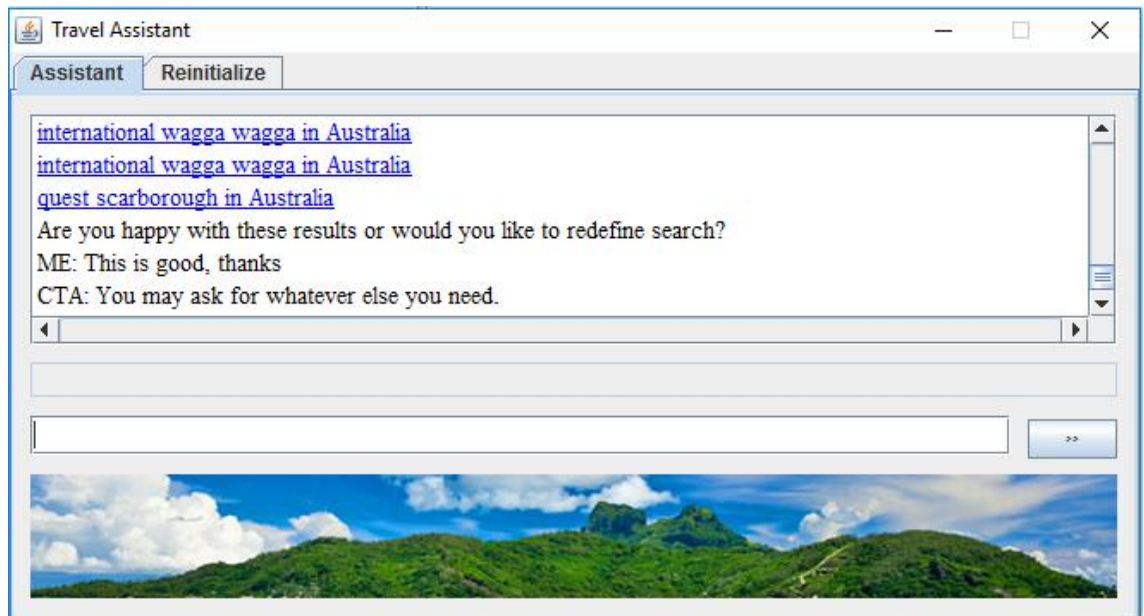
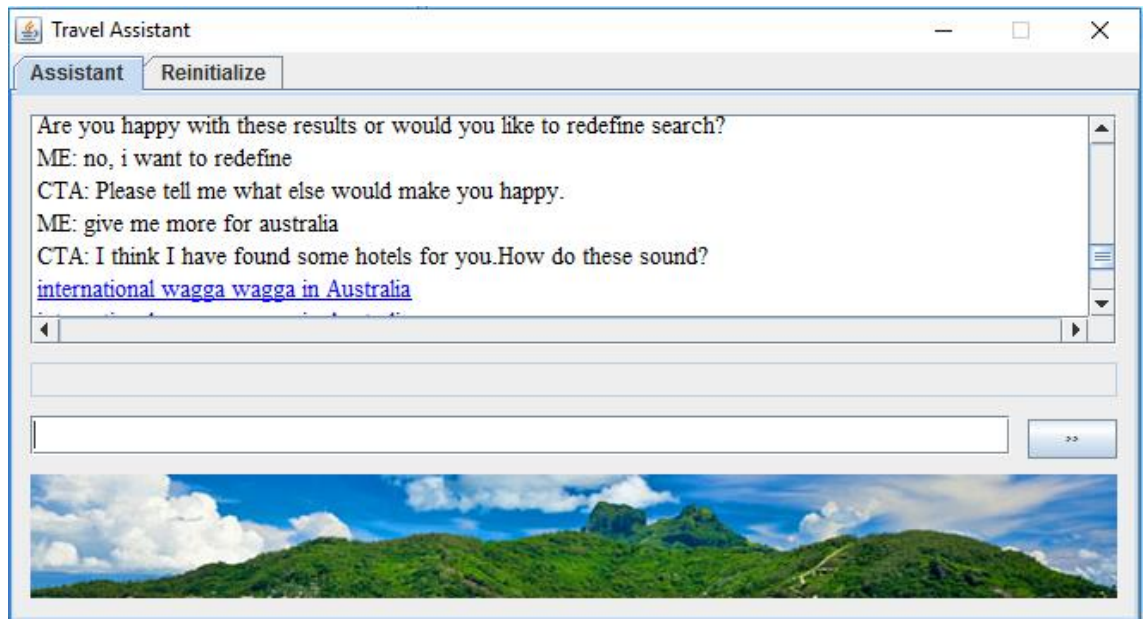
If the user is unhappy with the preferences that he initialized they can press the "Reinitialize" tab where they can see blank fields to complete. They are there to create new preference settings. The user fills the first box of each sentence with what kind of trip it will be and the second one with how they would want the trip to be in case it is that kind of trip. Once again they can type whatever they want in any way they would see fitting and the machine tries to understand and memorise it in a way that it understands.

The users have the ability to add more preferences by pressing the ADD button and is given the right to fill up to 10 different preferences. When the users believes those are the preferences that suit them they must click the button

SAVE to implement those preferences.

Below you can see a Demo of how it can be used:





Travel Assistant

Assistant Reinitialize

Usually I like my trips to be

But if I am with

I want them to be

If I am with

IV. System Requirements

The system is to be given input in natural language that are about traveling. The input can contain any word the user might want to use. When the system asks the user something the user must answer to it. The results that any input should produce are three links to three hotels. The more information the user gives from that point on, the more links the system should provide the user with.

V. Technologies

Java:

High-level programming language. I used it to build the interface and implement all the client side code.

XAMPP:

Program that allows you to create a local server to test your files. I used it to run the server side code.

PHP:

Server programming language. I used it to implement the server side code. I chose to use this for it is simple to use and has an amazing JSON parser.

JSON:

Text type files where you can parse them with certain languages. It is used to create objects and arrays in an organized matter. I used it to be able to check words for query enhancement and have an organized dictionary.

Google:

It is a worldwide used search engine. I use it to crawl through booking.com in the most efficient fashion.

Booking.com:

It is a site that has a huge library of hotels across the globe with open reviews. I use it to find destinations to suggest to my clients.

Evaluation

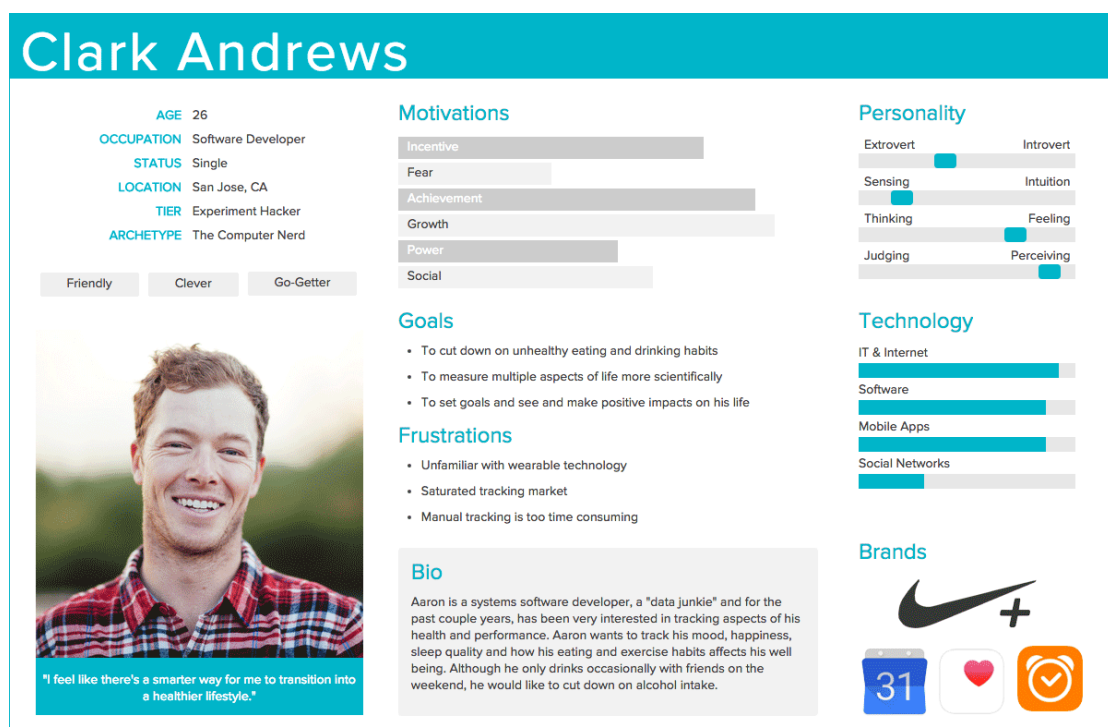
I. User Case scenarios

1. Scenario 1
2. Scenario 2
3. Scenario 3

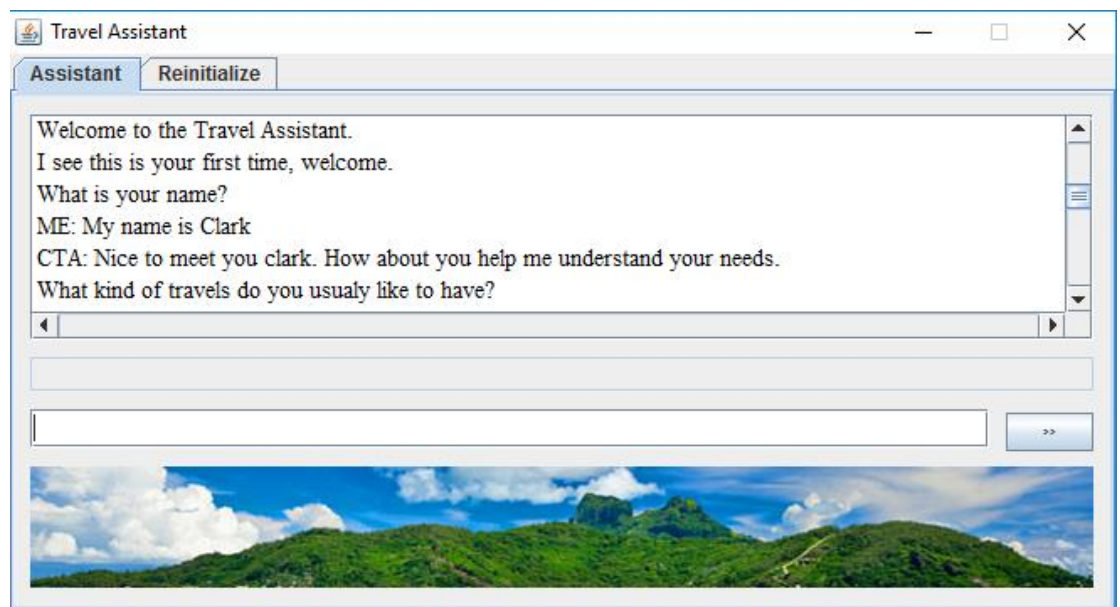
I. User Case Scenarios

*For the evaluation we will be using fake personas while trying to simulate possible scenarios that may happen in real life. Furthermore here, we will use input that seem logical for each persona and expect output which that persona would find more appealing.

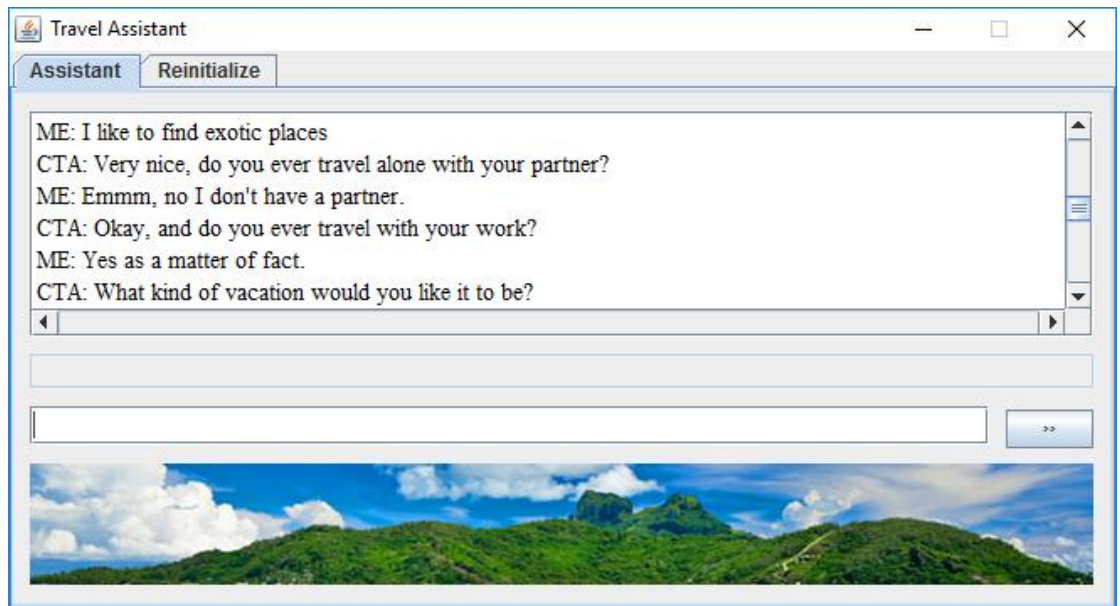
Scenario 1:



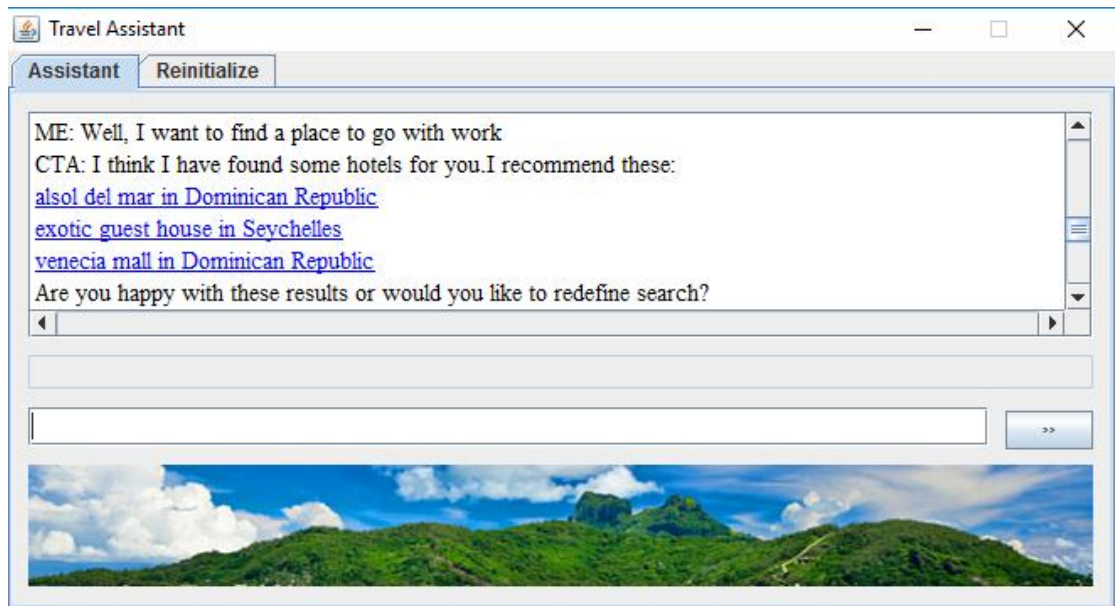
Meet Clark Andrews. He is a Software developer who lives a good life in San Jones, California. He has a well-paying job however it is a very stressful one. These past 6 months Clark was responsible for creating a system for one of the wealthiest companies in California, hence that created a lot of stress. He is an extrovert however he didn't have any time to go out and enjoy his time as he usually likes. More than that, he is a man of feelings rather thoughts and like so he started to break down. Once the project had come to a finish, his boss was extremely satisfied with his work, since they made a million dollars agreement. He then offered Clark paid vacation anywhere in the USA. Despite Clark's excitement he didn't know where to go. On the other hand, he heard of a system from a travelling buddy of his that suggested hotels to go according to your own preferences and likings. Thereupon he decided to check it out. He introduced himself to the assistant and claimed that he generally likes exotic places.



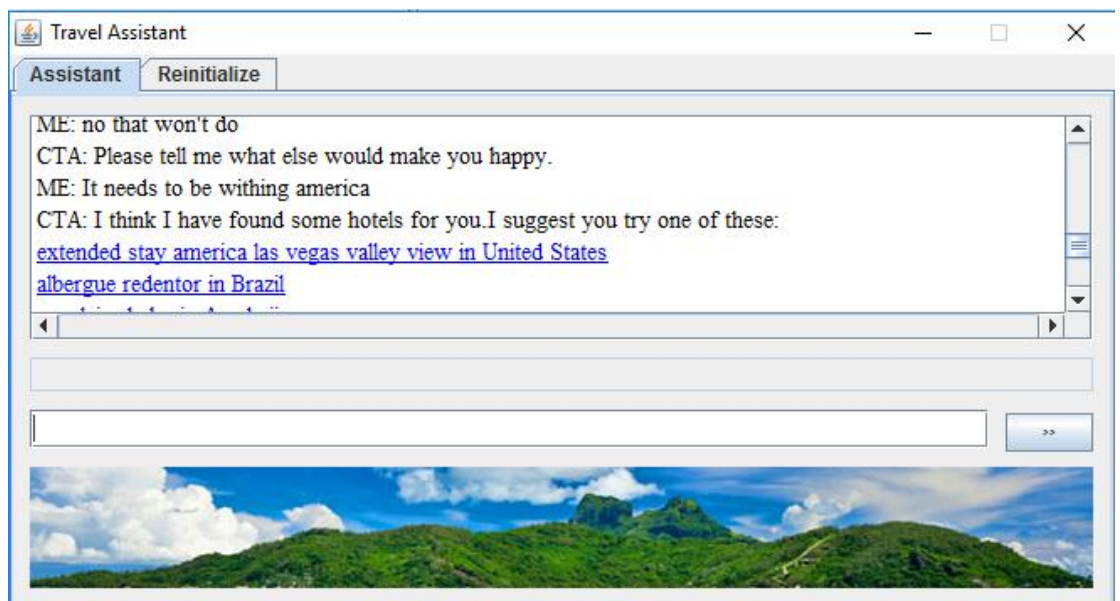
He then was asked about more specific desires and was inclined to say that he wants something expensive when with work.

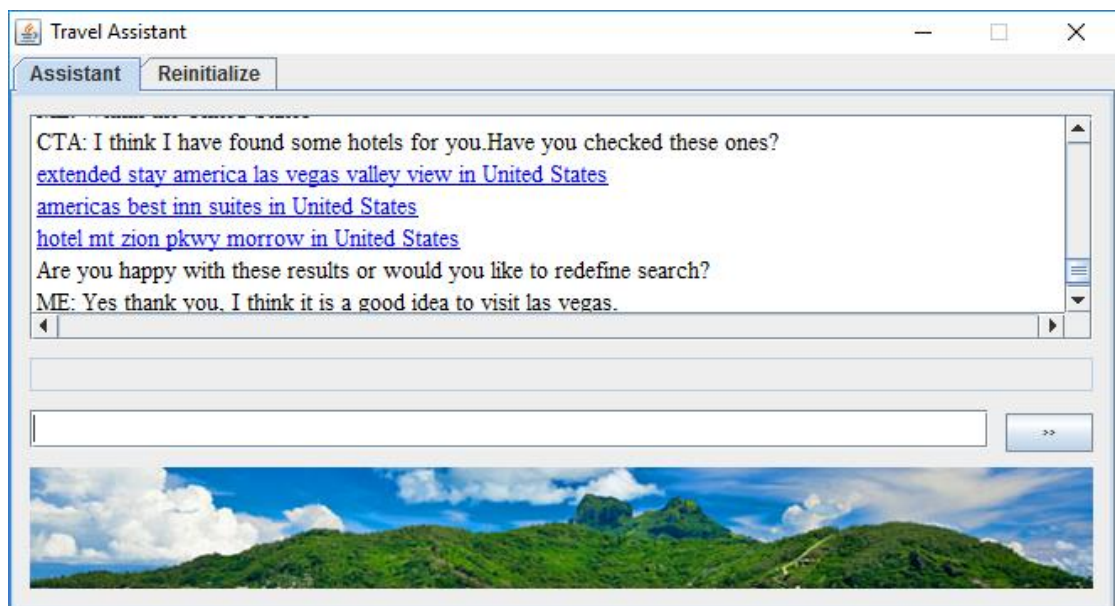
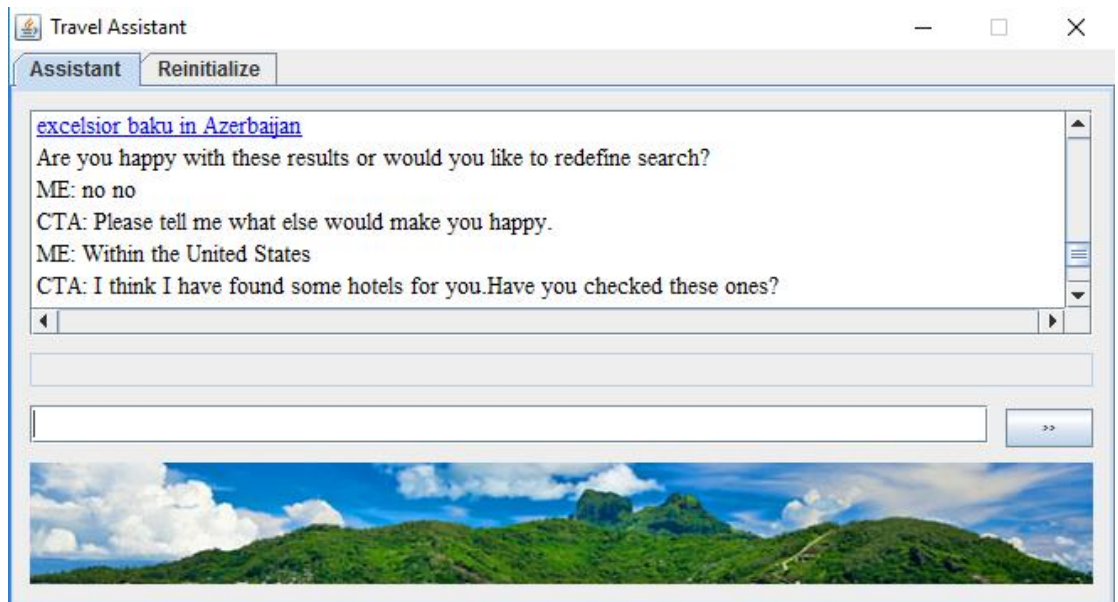


Once the introductions had reached an end he said that he needs a recommendation for a place to go with work. The system responded with a few suggestions.



He was not satisfied however because he was only allowed in the United States. So he tried to explain this matter to the system.





Finally, he did not reserve at the hotel he was suggested to, however he took the assistant's advice and visited Las Vegas. He enjoyed that he could simply talk to the system however he felt like it since he is a developer and understands coding limitations, and thinks about using it again when he gets the chance.

We tried to compare the results with what would happen if this person tried to just search on google:

30 Places We Want to Work | GOOD

<https://www.good.is/slideshows/30-places-we-want-to-work> ▼

Nov 11, 2010 - We've compiled a list of 30 of the companies that, if we worked there, would have us excited to get out of bed each morning.

27 Great Jobs for People Who Love to Travel - Business News Daily

www.businessnewsdaily.com/2389-jobs-travel-lovers.html ▼

Mar 30, 2017 - Do you want to get paid to see the world? ... Best jobs for travel lovers ... want to spend your time exploring the world instead of working 9-to-5 ...

15 Ways To Work Overseas And Keep Traveling the World | Thought ...


thoughtcatalog.com/.../03/15-ways-to-work-overseas-and-keep-traveling-the-world/ ▼

Mar 27, 2014 - Finding work overseas is not like finding a job in the United States. ... Make sure you get to your destination well before the season starts to secure a ... Many of the easy low-wage jobs usually go to people from ... Websites like Craigslist and Gumtree are two places to advertise your abilities and find work.

We can clearly see that it thinks, this person is looking for a job. That is not what our user would want thus our system provided him with a viable result.

Scenario 2:

Bananios Kyriakidis



"I like blue colours even though I am colorblind."

Age: 22
Work: Topographer
Family: Forever Alone.
Location: Nicosia
Character: Fire

Personality

Introvert	Extrovert
Analytical	Creative
Conservative	Liberal
Passive	Active

Huge Personality

Knows the whole Alphabet

Understands numerals within the limits of 0 to Million even in their negative form

Goals

- Germany - Brazil 7-1.
- I would like to achieve greatness, reaching higher than anyone else.
- I would like to build the next big MOBA type franchise game.

Frustrations

- I don't like pineapple on pizzas.
- I hate crowded places.
- I feel annoyed when my team doesn't do well in projects.
- I despise political debating.

Bio

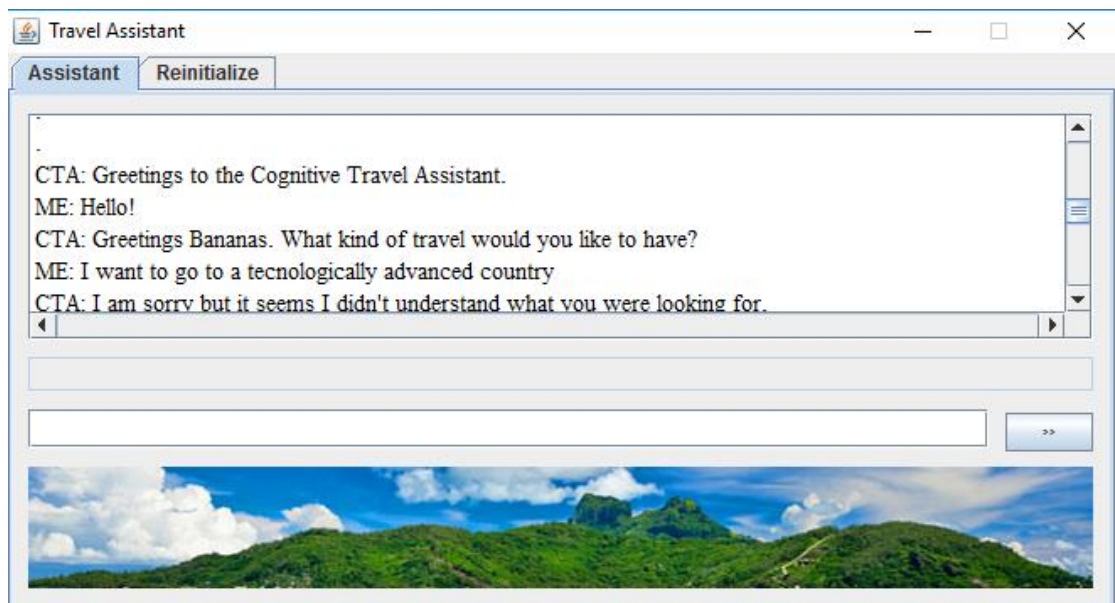
I was once a student interested in Computer Science. Now my sight has become wide, and I am looking to expand cities and villages into becoming amazing marvels of road work.

Motivation

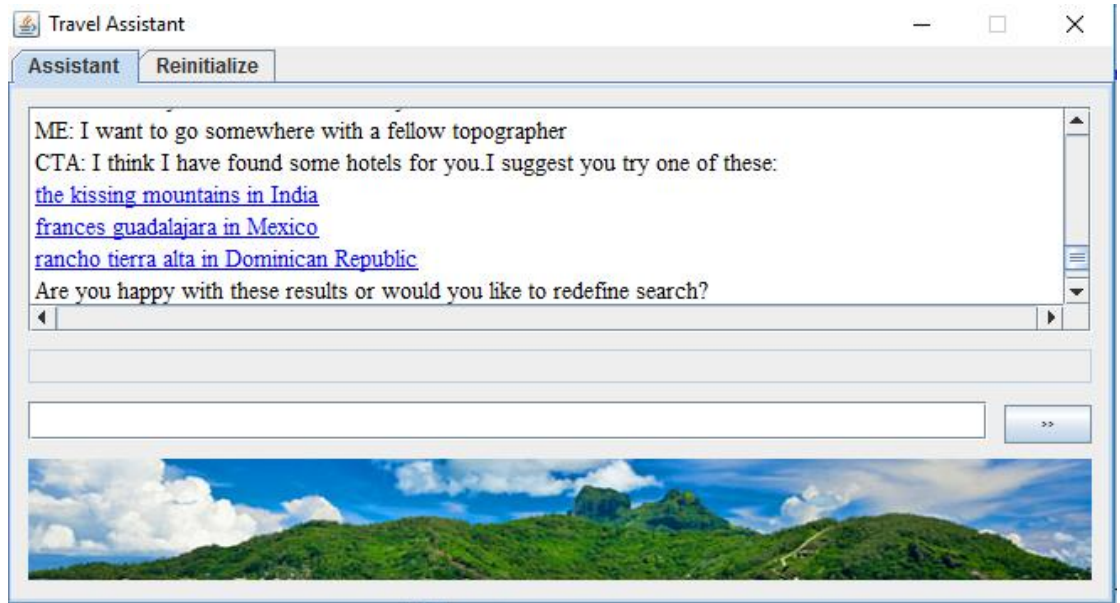
Incentive	
Fear	
Achievement	
Growth	
Power	
Social	

Bananios is one of the head topographers of Cyprus Road Construction National. He is a passionate man that does not accept failure. This persona gets paid to take other employs from other companies to business trips. In view of that we see that he uses the application often since he needs to be abroad all the time. He doesn't have much of an interest to where specifically he wants to be thus he uses our system to be recommended of hotels and just pick one that seems nice. Budget is of no issue since he works for a very important company and he will not let money cloud his way to greatness. That is if he is on business trips. Whenever he is forced to go alone, since he doesn't take vacation by choice, he goes small and cheap holidays just so that his boss can't force him to go again for some time. For instance Bananios has to take a fellow topographer anywhere and show him a good time in order to sweeten the deal on the new road that is being constructed at the Aglantzia street. This is his experience with the system.

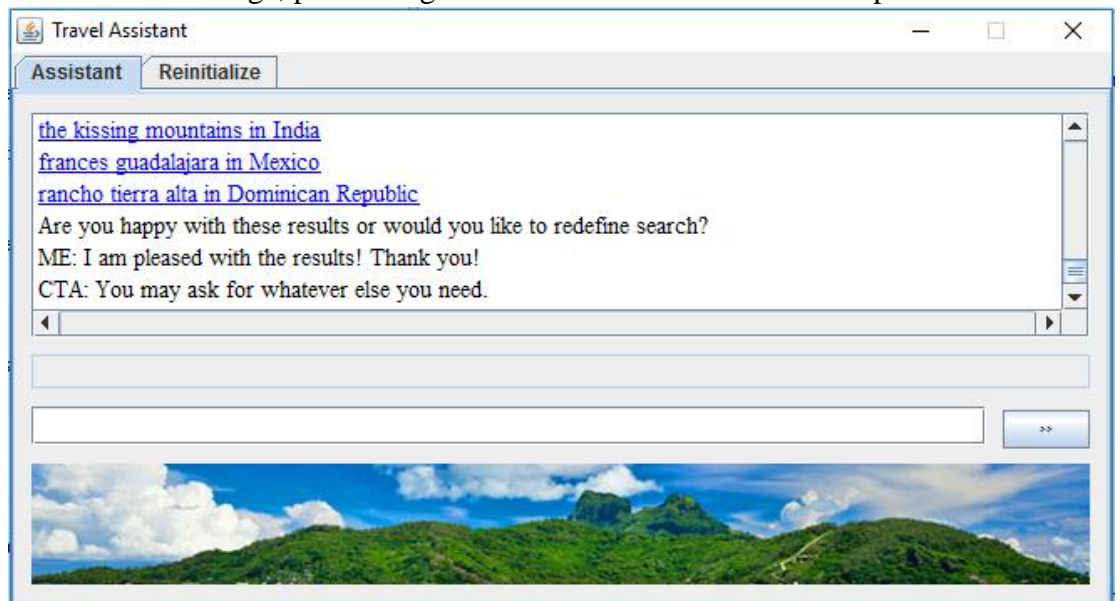
Firstly, it is important to point out that the system does not try to reinitialise its knowledge since this is not Bananios first time using the program. Notably, we can see that Bananios is trying to greet the program to feel a little more familiarity from it.



He then seems that our system didn't find any results so it says that it didn't understand what he meant. That means Bananios had to rephrase his needs.



He was happy with the results as it suggested 3 different countries, which meant he had options. And since he doesn't really mind where to go he will not search any further. Once again trying to make the experience a little more lifelike he puts emotion into his response and even tries to thank the agent. To his bad misfortune the agent is not yet developed in a stage where it can freely respond to anything, which hopefully will be added in a later stage, proceeding to answer back in a more cold response.



The hotels the assistant had suggested were found due to the previously set preferences of our client.

Travel Assistant

Assistant Reinitialize

Usually I like my trips to be

But if I am with I want them to be

If I am with then they must be

Add Pref Save Reset

What our client doesn't understand is since he did not say that he doesn't care about money in the more specific preferences our agent takes his general preference and adds it to the main query. Which then will search for something in the mountains that is cheap. Thankfully this hasn't prevented our client from using our system again and again. He believes it's an easy solution.

Once again we try to compare the results with google and we see that it suggests 62 places that the user has to look through instead of suggesting only a few. This shows efficiency from our system.

The 30 Cheapest Places To Travel In 2017 - Forbes

<https://www.forbes.com/sites/.../2016/11/30/30-cheapest-places-to-travel-in-2017/> ▼

Nov 30, 2016 - Read on to get their favorite picks for the most affordable destinations to visit next year. And if you want more great affordable travel ideas, ...

Missing: fellow topographer

12 of the world's cheapest holiday destinations for 2017 - Skyscanner

<https://www.skyscanner.net › Travel News> ▼

Jan 23, 2017 - Here are 12 of the cheapest countries to visit in 2017. ... Even if you've managed to bag cheap flights for your holiday, no-one wants to have to ...

Missing: fellow topographer

Budget Trips: 20 Of The Cheapest Places To Travel | Rough Guides

<https://www.roughguides.com/gallery/budget-trips-20-of-the-cheapest-places-to-travel/> ▼


Apr 25, 2017 - Here are our top 20 cheapest places to travel around the world. ... You'll find all the information you need to plan a budget trip in our Snapshot ...

Missing: fellow topographer

Scenario 3:

Tsinik

Xtensio



Trait

Trait

Trait

Trait

Goals

- Travel the world.
- Become known.

Frustrations

- Idiocracy.

Bio

I am a computer science student. I live my life to the fullest and I try to keep moving in order to avoid time. The only thing that scares me in life is inactiveness.

"Life without living is simply surviving."

Age: 23
Work: Student
Family: None.
Location: Nicosia
Character: Energetic

Motivation

Incentive	80%
Fear	10%
Achievement	85%
Growth	80%
Power	80%
Social	85%

Personality

Introvert	Extrovert
Analytical	Creative
Conservative	Liberal
Passive	Active

This is Tsinik. He is studies computer science in the University of Cyprus. He feels tired after four years of studies so he believes it is time for a rest. He is planning to work hard all summer to collect enough money to travel across Europe and see as many countries as the money will get him. He is a passionate person where anything he does seem eccentric and sometimes annoying. That is because he is hyperkinetic and can't sit still. That's why on this trip he doesn't care about planning and wants to go wherever the road takes him. That is why he will be using the program in every country just to find any location to stay. He needs it to be cheap and doesn't care about personalisation so he uses the system in a more general way. Only thing that he cares is that he can find cheap hostels to sleep at night. Then move onwards towards completing his adventure.

He had already set his preferences to cheap hostels since he wants his later inputs to be as simple as possible.

The screenshot shows a window titled "Travel Assistant" with two tabs: "Assistant" and "Reinitialize". The "Assistant" tab is active. It contains a form with the following labels and input fields:

- "Usually I like my trips to be" followed by a text box containing "in cheap hostels".
- "But if I am with" followed by an empty text box.
- "If I am with" followed by an empty text box.
- "I want them to be" followed by an empty text box.
- "then they must be" followed by an empty text box.

At the bottom of the window, there are three buttons: "Add Pref", "Save", and "Reset".

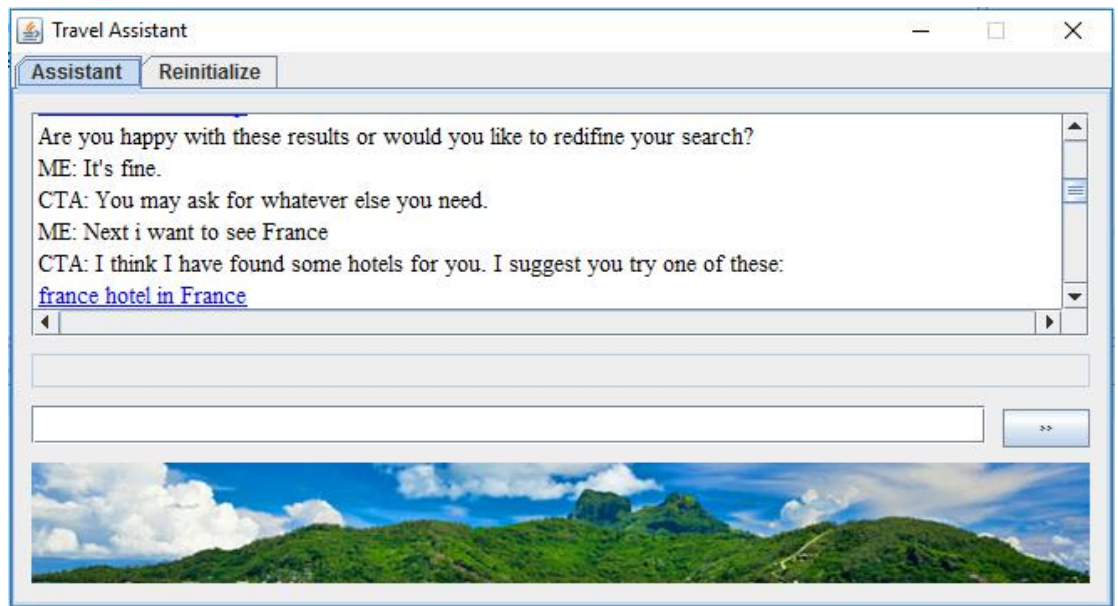
First stop is Italy. Tsirik doesn't want to start in Greece since it doesn't feel like a European Country.

The screenshot shows the same "Travel Assistant" window, but now the "Assistant" tab displays a conversation log. The log contains the following text:

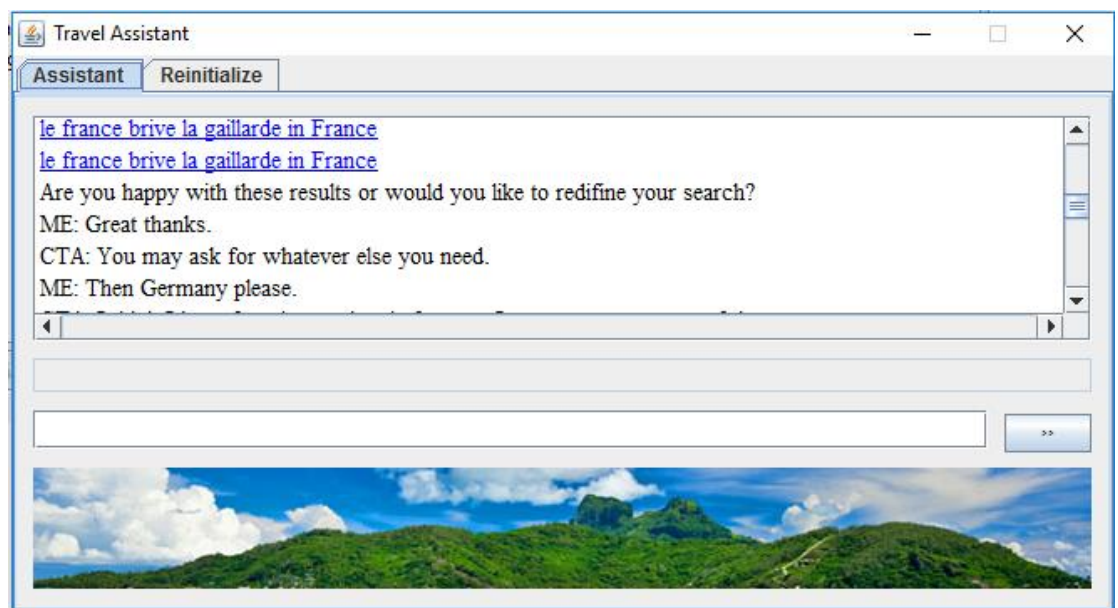
CTA: Greetings to the Cognitive Travel Assistant.
 ME: First stop is Italy
 CTA: I think I have found some hotels for you. I suggest you try one of these:
[la luna blu in old town in Italy](#)
[chroma octho in Italy](#)
[cadorna suites in Italy](#)

Below the log, there is a horizontal scrollbar and a text input field. At the bottom of the window, there is a wide landscape image showing green hills under a blue sky with white clouds.

Since he found a hostel to stay that helps him see in which city he will stay.



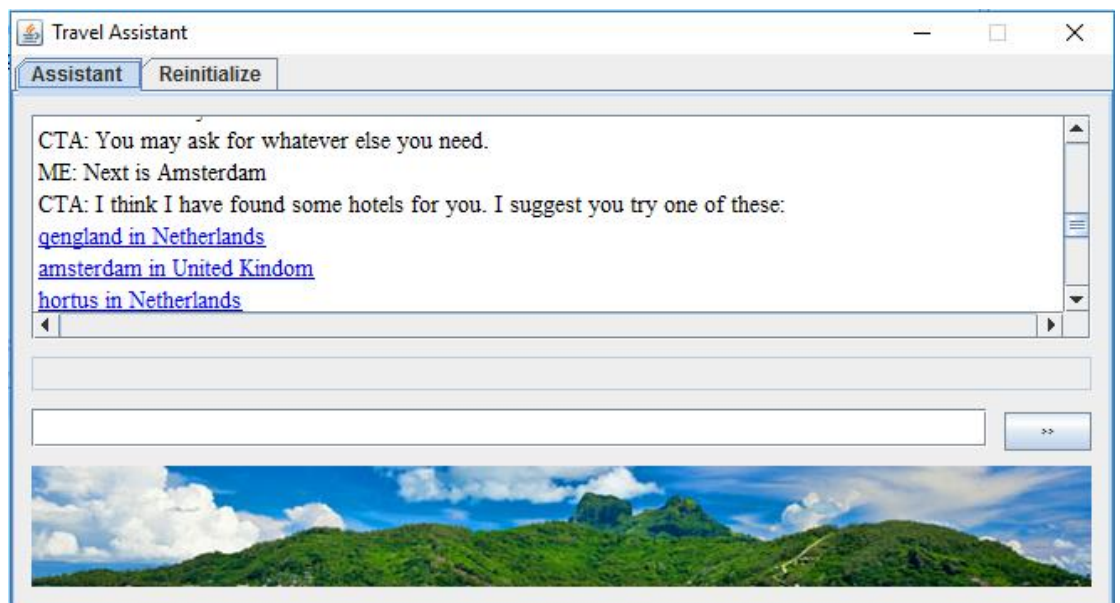
After Italy he decides to relax somewhere in France and enjoy some of that famous cuisine. Hopefully there won't be any trouble and that it won't be too expensive as his budget is limited.



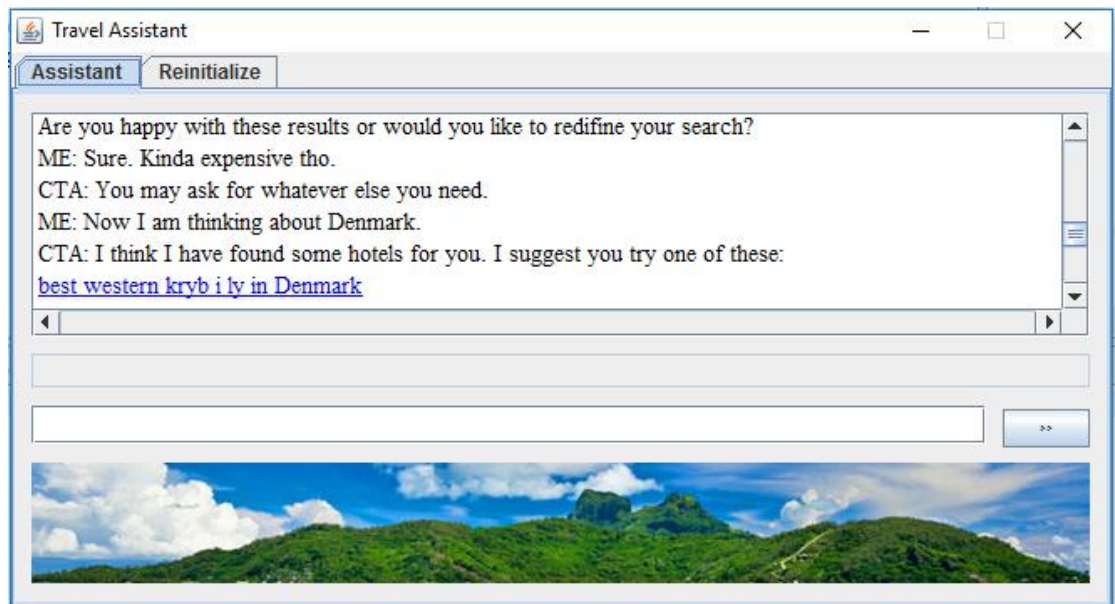
Even though he doesn't really like Germany he has to pass through there so that he will reach countries he will like.



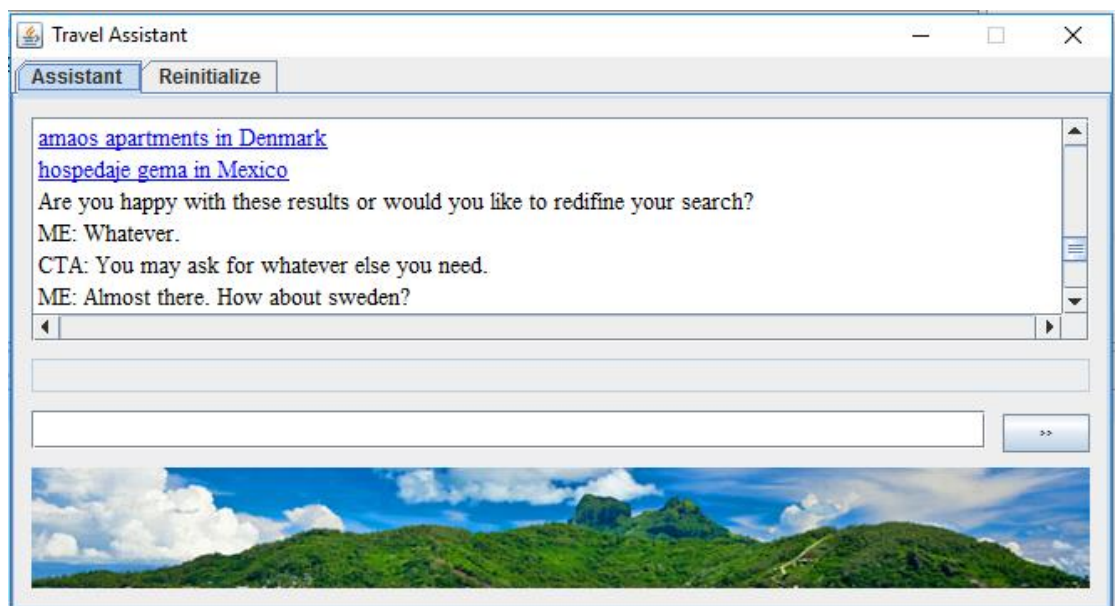
Finally he can go to somewhere he has been looking forward to.
Amsterdam.



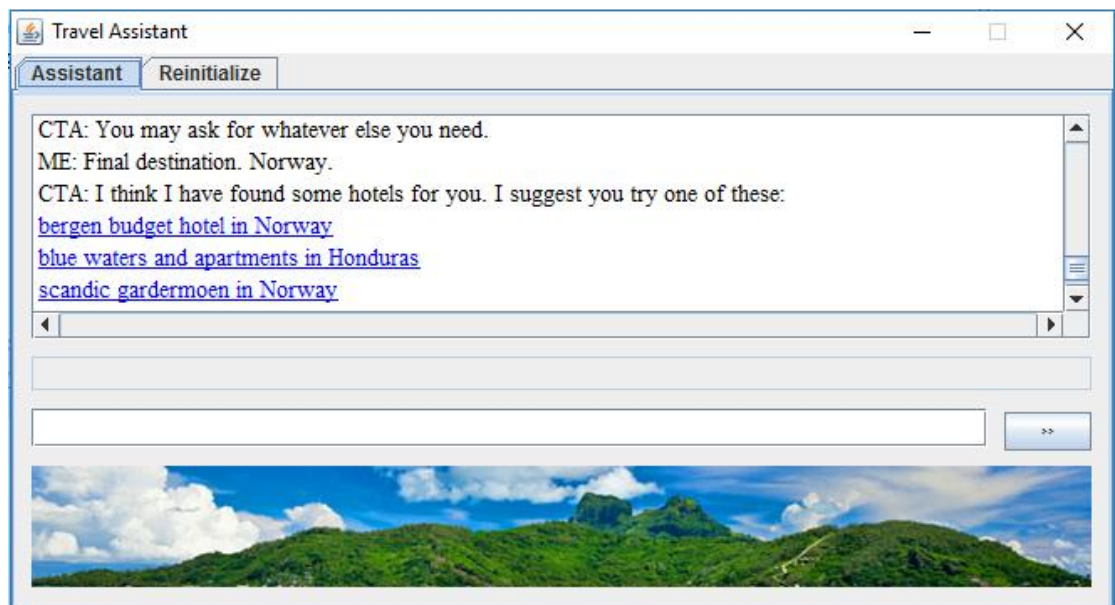
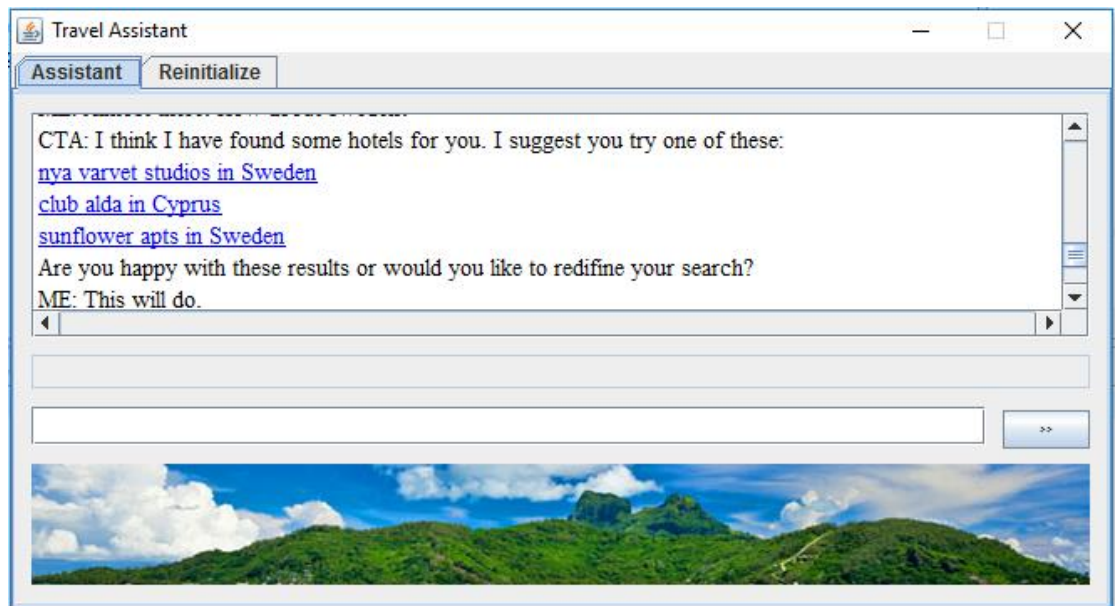
Unfortunately for him Amsterdam is a bit expensive so he won't be taking
the assistants recommendations, however he will try to find something of
his own accord.



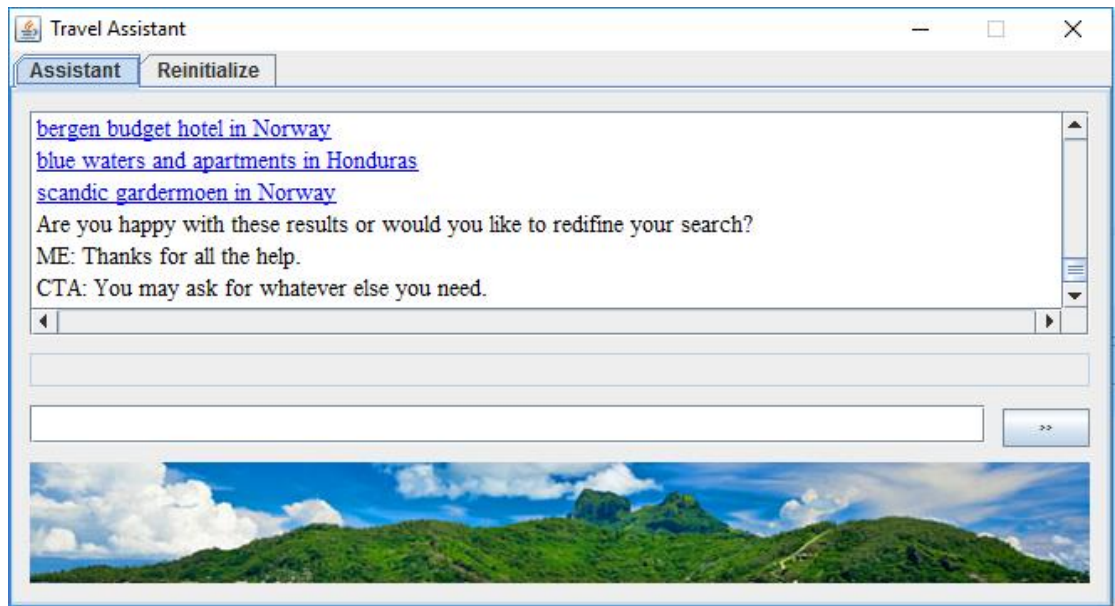
The system located only two results in Denmark but sees fit to present on from Mexico. That happens due to the fact that it searches reviews and someone must have said something about Denmark in the reviews or there are no more Denmark hotels in the booking.com database.



Furthermore he starts climbing in the ice Sweden. Once again the results do not contain solely Swedish hotels. Unfortunately the system can't give correct results all the time.



In the end, Tsinik decides to finish off his quest in the top of Europe, in Norway. He found some hotels however the results don't really match his desires. This show how limited the database we are using is.



In this run we can clearly see the non-lifelikeness of the system. This is due to the fact that this user tried to get a lot of queries and lost from the illusion. However it can still be used in this scenario and help our aspiring traveller.

This time when we tried to compare with what the user would have asked for, it suggests tools that help you find what you look for. Looking deeper in to this we can see that some of the tools suggested can help the user find exactly what he looks for.

Hostels in Europe from \$3 - Europe's Best Hostels - hostelworld.com

www.hostelworld.com/

4.7 ★★★★★ rating for hostelworld.com

Book your hostels in Europe with Everything you need for a Great Stay!

Destinations: London, Amsterdam, Berlin, Barcelona, Rome, Paris

Amenities: Private Rooms, Air Conditioning, Bar, Free WiFi, 24 Hour Reception, 24 Hour Security,...

[Hostels Hostels](#) · [Cheap Hotels Hostels](#)

Cheap Europe Hostels - Incredibly Low Prices - tripadvisor.com

www.tripadvisor.com/

TripAdvisor Searches Up to 200 Sites to Find You the Lowest Hotel Prices.

Styles: Romantic, Family, Great Value

Amenities: Free WiFi, Swimming Pool, Free Breakfast

[Best Value Hotels](#) · [Find Hotels](#) · [Travel Forum](#) · [Cheap Flights](#) · [Area Map](#) · [Plan Your Trip](#)

[Romantic Hotels](#) - from US\$247.00/day - [Compare Prices](#) · [More](#)

Better Than Hostels - Rent Amazing Properties - airbnb.com

www.airbnb.com/

4.4 ★★★★★ rating for airbnb.com

Search Unique Rentals Worldwide. Save Money & Book on Airbnb Today!

Over 3,000,000 listings · 24/7 customer service · \$1,000,000 Host Guarantee · Travel like a local

2015 Innovative Brand of the Year – Marketing Magazine

[Learn how Airbnb Works](#) · [Signup Now](#) · [View most Popular Rentals](#) · [What is Airbnb™?](#)

In conclusion we can see that a user using Google will get too many suggestions and not exactly what he is looking for. Since this system is explicitly for this subject, it can actually give more preferable results.

Chapter 7

Conclusion

-
- I. Strengths
 - II. Weaknesses
 - III. Future Work
-

I. Strengths

The point of this thesis was to create an agent that can be used by anyone in order to help them find preferable locations as to where to go and spent a pleasant holiday or a proper business trip. In the end we have created such a system. Any user of any social class can have access to this system and use it without any kind of computer or programming knowledge. All they have to know is how to use their keyboards and mice.

The system developed in this thesis, is also flexible as in the fact that it can be updated extremely easily. Since it reads from external files, all anyone needs to do is update those files and you have a more advanced version already. Furthermore if the users are given some access to the mentioned files they could even alter their own experience through their own point of view.

To name another important factor, uses already proven to be great technologies. As it is mentioned this system takes advantage of Googles great crawling benefits as Google is the fastest and most accurate crawler known so far. And since it is using Google's systems to create a more precise search, we can see that the results the system provides are more preferable.

To summarise this, thesis was more of an implementation rather than a research. In the end we were able to develop a semi-interactive system capable of understanding some of the user's needs, which then has the ability to guide you towards a preferable goal. It is a non-absolute system which can be used in

different scenarios and produce different results according to the user and the case at hand.

II. Weaknesses

This thesis discusses the abilities and the developments of cognitive assistants. Even at the start of the assignment it was discussed for which subject the cognitive assistant would be assigned. Even though the trip assistant was chosen, the main subject should have been generally cognitive assistants. Through the research I have realised that cognitive assistants may be obsolete as they are very limited. In seldom cases a cognitive assistant presents the user with results of maximum potential, however, most cases the assistants are limited.

In the same way, the algorithm we have picked for our assistant is one of the most limited since it is bound by rules. Having to create ruling based on general direction and then create levels of preference, may give you a head start, but in spite of that it can reach a certain height. The true way to knowledge is free learning. By using ruling you predefine your results to a certain aggregate. However if you predefine several actions and then let a system understand the weight of each action, this will conclude in a system with an almost clear understanding of its existence. Through this research I have come to believe that this is not the true way to technological advancement but through sample based training. Something that is used by most AI researching facilities at the moment.

Complementary to this, the certain subject we are working on, is also extremely limited. Even though it has the ability to advance it is limited to the database of booking.com and the limits of Google's permissions. If a user were to keep asking the system for locations, without any pause, Google would shut them out for a period of time. That would mean that, anyone trying to find mistakes in the system would be able to find some easily.

III. Future Work

As it was already discussed this system has the capability to advance and become a better assistant. The only thing needed is for the JSON files to be added to.

Firstly the contradiction JSON must be filled. With more contradictions we can get more clear and correct queries. Contradictions can take many forms as each word has many synonyms. Meaning each synonym of a word has to have all of the contradictions of its synonyms and each contradiction must be recorded with its own synonyms. For this to be done a simple website can be created where any user can add a word and a contradiction. However this creates the problem of intentionally submitting false information and ruining the assistant's knowledge.

Likewise a negation function can be created to understand when a user does not want something. The system can scan for negative words and then replace the pair of negative word and preference with a contradiction from the contradiction JSON thus creating a more accurate result. The reason this needs the contradiction JSON and not just adding Google's negation symbol is because that would create, as already tested, lots of problems.

Furthermore, the dictionary JSON could also be upgraded at any time. Once again through a simple website with a few blank spaces for filling, users can create a better word relation tree. Not so many words are needed to understand a travel context however those words can be written in a lot of ways and present a lot of meanings. Users can submit pairs or groupings of their own preferences and backgrounds. This unfortunately also risks users intentionally inserting incorrect data. It is a risk, I believe, worth taking.

Finally the conversation mechanism can also be upgrade to add a bit of life to the assistant and enhance the illusion of augmentation. Before sending each input to be tested by the server the client side might be able to check for certain keywords and pick from a cluster of answers to create a real life atmosphere.

References

- [0] Jarno M. Koponen “The Future Of Algorithmic Personalization”
<https://techcrunch.com/2015/06/25/the-future-of-algorithmic-personalization/>
- [1] “Havyn: a cognitive assistant for cybersecurity – Cognitive Voices – Medium”
<https://medium.com/cognitivebusiness/havyn-a-cognitive-assistant-for-cybersecurity-e6580898f49e>
- [2] Adrienne Lafrances “A Search Engine for Your Memories”
<https://www.theatlantic.com/technology/archive/2016/01/sorry-dave-afraid-i-cant-do-that/431559/>
- [3]<http://www.ccbi.cmu.edu/4CAPS/index.html>
- [4] TC Currie “IBM Watson to Bring ‘Cognitive Assistant’ Capabilities to Slack”
<https://thenewstack.io/slack-teams-watson-give-im-cognitive-abilities/>
- [5] William Mark , Raymond Perrault “Cognitive Assistant that Learns and Organizes” <http://www.ai.sri.com/project/CALO>
- [6] Sunil Ray “Essentials of Machine Learning Algorithms”
<https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/>

Appendix A

Common Words

about above across after again against all almost alone along already also although
always among an and another any anybody anyone anything anywhere are area areas
around as ask asked asking asks at away back backed backing backs be became because
become becomes been before began behind being beings best better between big both
but by came can cannot case cases certain certainly clear clearly come could did differ
different differently do does done down down downed downing downs during each
early either end ended ending ends enough even evenly ever every everybody everyone
everything everywhere face faces fact facts felt few find finds first for four from full
fully further furthered furthering furthers gave general generally get gets give given
gives go going good goods got great greater greatest group grouped grouping groups
had has have having he her here herself high high high higher highest him himself his
how however if important in interest interested interesting interests into is it its itself
just keep keeps kind knew know known knows large largely last later latest least less let
lets like likely long longer longest made make making man many may me member
members men might more most mostly mr mrs much must my myself name necessary
need needed needing needs never new new newer newest next no nobody non noone not
nothing now nowhere number numbers of off often old older oldest on once one only
open opened opening opens or order ordered ordering orders other others our out over
part parted parting parts per perhaps place places point pointed pointing points possible
present presented presenting presents problem problems put puts quite rather really right
right room rooms said same saw say says second seconds see seem seemed seeming
seems sees several shall she should show showed showing shows side sides since small
smaller smallest so some somebody someone something somewhere state states still still
such sure take taken than that the their them then there therefore these they thing things
think thinks this those though thought thoughts three through thus to today together too
took toward turn turned turning turns two under until up upon us use used uses very
want wanted wanting wants was way ways we well wells went were what when where
whether which while who whole whose why will with within without would year years
yet you younger youngest your yours

Appendix B

Dictionary JSON

```
{
  "rain" : "rain" ,
  "windy" : "windy" ,
  "humid" : "humid" ,
  "heat" : "heat" ,
  "sunny" : "sunny" ,
  "cold" : "cold" ,
  "cloudy" : "cloudy" ,
  "snow" : "snow" ,
  "continent" : "distant" ,
  "worldwide" : "distant" ,
  "international" : "distant" ,
  "kid" :
    [
      "family" ,
      "fun"
    ] ,
  "teen" :
    [
      "family" ,
      "fun"
    ] ,
  "family" : "family" ,
  "relax" :
    [
      "quiet" ,
      "spa"
    ] ,
  "quiet" : "quiet" ,
  "spa" : "spa" ,
  "pets" : "pets" ,
  "adventure" :
    [
      "tropical" ,
      "exotic"
    ] ,
  "forest" : "forest" ,
  "tropical" : "tropical" ,
  "exotic" : "exotic" ,
  "partner" :
    [
```

```

        "romantic" ,
        "anniversary"
    ],
    "romantic" : "romantic" ,
    "anniversary" :
    [
        "romantic" ,
        "expensive"
    ],
    "expensive" : "expensive" ,
    "cheap" : "cheap" ,
    "local" : "local" ,
    "fun" : "fun" ,
    "activities" :
    [
        "activities",
        "Fun"
    ],
    "party" : [
        "fun" ,
        "party"
    ],
    "geolocation" :
    [
        "island" ,
        "countryside"
    ],
    "island" :
    [
        "beach" ,
        "mountain"
    ],
    "countryside" : "mountain" ,
    "mountain" : "mountain" ,
    "beach" : "beach" ,
    "personal" :
    [
        "friends" ,
        "alone" ,
        "Partner"
    ],
    "friends" : "fun" ,
    "alone" :
    [
        "fun" ,
        "culture" ,
        "skill" ,
        "health"
    ],
    "promotion" : "business" ,

```

```
"business" : "business" ,
"proposal" : "proposal" ,
"culture" : "culture" ,
"skill" : "skill" ,
"health" : "health" ,
"medicine" : "health" ,
"spiritual" : "health" ,
"work" : "business" ,
"pleasure" :
    [
        "exotic" ,
        "relax"
    ] ,
"cozy" :
    [
        "warm" ,
        "relax"
    ] ,
"silent" : "quiet" ,
"boutique" : "boutique" ,
"sandy" : "beach" ,
"calm" : "relax" ,
"near" : "local"
}
```

Appendix C

Contradiction JSON

```
{  
  "expensive" : ["cheap"],  
  "cheap" : ["expensive"],  
  "mountain" : ["beach"],  
  "beach" : ["mountain"],  
  "quiet" : ["party"],  
  "relax" : ["party"],  
  "party" : ["quiet", "relax"],  
  "exotic" : ["city"],  
  "tropical" : ["city"],  
  "forest" : ["city"],  
  "city" : ["exotic", "tropical", "forest"]  
}
```


Appendix D

Location JSON

The file is too big so here is a sample:

```
{  
  "africa" : "africa",  
  "europe" : "europe",  
  "usa" : "usa",  
  "asia" : "asia",  
  "america" : "america",  
  "afghanistan" : ["charikar" , "ghazni" , "herat" , "jalalabad" , "kabul" , "kandahar" ,  
  "kunduz" , "mazari sharif" , "puli khumri" , "sar-e pol" , "sheberghan" , "taloqan"] ,  
  "charikar" : "charikar" ,  
  "ghazni" : "ghazni" ,  
  "herat" : "herat" ,  
  "jalalabad" : "jalalabad" ,  
  "kabul" : "kabul" ,  
  "kandahar" : "kandahar" ,  
  "kunduz" : "kunduz" ,  
  "mazari sharif" : "mazari sharif" ,  
  "puli khumri" : "puli khumri" ,  
  "sar-e pol" : "sar-e pol" ,  
  "sheberghan" : "sheberghan" ,  
  "taloqan" : "taloqan" ,  
  "albania" : ["durrësi" , "shkodër" , "tirana"] ,  
  "durrësi" : "durrësi" ,  
  "shkodër" : "shkodër" ,  
  "tirana" : "tirana" ,
```

"algeria" : ["ain beida" , "algiers" , "annaba" , "bab ezzouar" , "baraki" , "barika" ,
 "batna" , "biskra" , "blida" , "bordj bou arrǧdj" , "bordj el kiffan" , "bou saⵔ" , "bⵝar"
 , "bⵝⵓ" , "chlef" , "constantine" , "djelfa" , "el eulma" , "el khroub" , "el oued" ,
 "ghardaⵓ" , "guelma" , "jijel" , "khenchela" , "laghouat" , "m'sila" , "medea" , "messad" ,
 "mostaganem" , "oran" , "ouargla" , "relizane" , "saida" , "setif" , "sidi-bel-abbⵝ" ,
 "skikda" , "souk ahras" , "tebessa" , "tiaret" , "tlemoen" , "tougourt"] ,
 "ain beida" : "ain beida" ,
 "algiers" : "algiers" ,
 "annaba" : "annaba" ,
 "bab ezzouar" : "bab ezzouar" ,
 "baraki" : "baraki" ,
 "barika" : "barika" ,
 "batna" : "batna" ,
 "biskra" : "biskra" ,
 "blida" : "blida" ,
 "bordj bou arrǧdj" : "bordj bou arrǧdj" ,
 "bordj el kiffan" : "bordj el kiffan" ,
 "bou saⵔ" : "bou saⵔ" ,
 "bⵝar" : "bⵝar" ,
 "bⵝⵓ" : "bⵝⵓ" ,
 "chlef" : "chlef" ,
 "constantine" : "constantine" ,
 "djelfa" : "djelfa" ,
 "el eulma" : "el eulma" ,
 "el khroub" : "el khroub" ,
 "el oued" : "el oued" ,
 "ghardaⵓ" : "ghardaⵓ" ,
 "guelma" : "guelma" ,
 "jijel" : "jijel" ,
 "khenchela" : "khenchela" ,

"laghouat" : "laghouat" ,

"m'sila" : "m'sila" ,

....

Appendix E

Location Abbreviations

ac-Ascension Island

ad-Andorra

ae-United Arab Emirates

af-Afghanistan

ag-Antigua And Barbuda

ai-Anguilla

al-Albania

am-Armenia

an-Netherlands Antilles

ao-Angola

aq-Antarctica

ar-Argentina

as-American Samoa

at-Austria

au-Australia

aw-Aruba

ax-Åland

az-Azerbaijan

ba-Bosnia And Herzegovina

bb-Barbados

be-Belgium

bd-Bangladesh

bf-Burkina Faso

bg-Bulgaria

bh-Bahrain

bi-Burundi
bj-Benin
bm-Bermuda
bn-Brunei Darussalam
bo-Bolivia
br-Brazil
bs-Bahamas
bt-Bhutan
bv-Bouvet Island
bw-Botswana
by-Belarus
bz-Belize
ca-Canada
cc-Cocos (Keeling) Islands
cd-Congo (Democratic Republic)
cf-Central African Republic
cg-Congo (Republic)
ch-Switzerland
ci-Cote D'Ivoire
ck-Cook Islands
cl-Chile
cm-Cameroon
cn-People's Republic of China
co-Colombia
cr-Costa Rica
cu-Cuba
cv-Cape Verde
cx-Christmas Island
cy-Cyprus
cz-Czech Republic

de-Germany
dj-Djibouti
dk-Denmark
dm-Dominica
do-Dominican Republic
dz-Algeria
ec-Ecuador
ee-Estonia
eg-Egypt
er-Eritrea
es-Spain
et-Ethiopia
eU-European Union
fi-Finland
fj-Fiji
fk-Falkland Islands (Malvinas)
fm-Micronesia, Federated States Of
fo-Faroe Islands
fr-France
ga-Gabon
gb-United Kingdom (no new registrations, see also UK)
gd-Grenada
ge-Georgia
gf-French Guiana
gg-Guernsey
gh-Ghana
gi-Gibraltar
gl-Greenland
gm-Gambia
gn-Guinea

gp-Guadeloupe
gq-Equatorial Guinea
gr-Greece
gs-South Georgia And The South Sandwich Islands
gt-Guatemala
gu-Guam
gw-Guinea-Bissau
gy-Guyana
hk-Hong Kong
hm-Heard And Mc Donald Islands
hn-Honduras
hr-Croatia (local name: Hrvatska)
ht-Haiti
hu-Hungary
id-Indonesia
ie-Ireland
il-Israel
im-Isle of Man
in-India
io-British Indian Ocean Territory
iq-Iraq
ir-Iran (Islamic Republic Of)
is-Iceland
it-Italy
je-Jersey
jm-Jamaica
jo-Jordan
jp-Japan
ke-Kenya
kg-Kyrgyzstan

kh-Cambodia
ki-Kiribati
km-Comoros
kn-Saint Kitts And Nevis
kr-Korea, Republic Of
kw-Kuwait
ky-Cayman Islands
kz-Kazakhstan
la-Lao People's Democratic Republic
lb-Lebanon
lc-Saint Lucia
li-Liechtenstein
lk-Sri Lanka
lr-Liberia
ls-Lesotho
lt-Lithuania
lu-Luxembourg
lv-Latvia
ly-Libyan Arab Jamahiriya
ma-Morocco
mc-Monaco
md-Moldova, Republic Of
me-Montenegro
mg-Madagascar
mh-Marshall Islands
mk-Macedonia, The Former Yugoslav Republic Of
ml-Mali
mm-Myanmar
mn-Mongolia
mo-Macau

mp-Northern Mariana Islands
mq-Martinique
mr-Mauritania
ms-Montserrat
mt-Malta
mu-Mauritius
mv-Maldives
mw-Malawi
mx-Mexico
my-Malaysia
mz-Mozambique
na-Namibia
nc-New Caledonia
ne-Niger
nf-Norfolk Island
ng-Nigeria
ni-Nicaragua
nl-Netherlands
no-Norway
np-Nepal
nr-Nauru
nu-Niue
nz-New Zealand
om-Oman
pa-Panama
pe-Peru
pf-French Polynesia
pg-Papua New Guinea
ph-Philippines, Republic of the
pk-Pakistan

pl-Poland
pm-St. Pierre And Miquelon
pn-Pitcairn
pr-Puerto Rico
ps-Palestine
pt-Portugal
pw-Palau
py-Paraguay
qa-Qatar
re-Reunion
ro-Romania
rs-Serbia
ru-Russian Federation
rw-Rwanda
sa-Saudi Arabia
uk-Scotland
sb-Solomon Islands
sc-Seychelles
sd-Sudan
se-Sweden
sg-Singapore
sh-St. Helena
si-Slovenia
sj-Svalbard And Jan Mayen Islands
sk-Slovakia (Slovak Republic)
sl-Sierra Leone
sm-San Marino
sn-Senegal
so-Somalia
sr-Suriname

st-Sao Tome And Principe
su-Soviet Union
sv-El Salvador
sy-Syrian Arab Republic
sz-Swaziland
tc-Turks And Caicos Islands
td-Chad
tf-French Southern Territories
tg-Togo
th-Thailand
tj-Tajikistan
tk-Tokelau
ti-East Timor (new code)
tm-Turkmenistan
tn-Tunisia
to-Tonga
tp-East Timor (old code)
tr-Turkey
tt-Trinidad And Tobago
tv-Tuvalu
tw-Taiwan
tz-Tanzania, United Republic Of
ua-Ukraine
ug-Uganda
uk-United Kingdom
um-United States Minor Outlying Islands
us-United States
uy-Uruguay
uz-Uzbekistan
va-Vatican City State (Holy See)

vc-Saint Vincent And The Grenadines

ve-Venezuela

vg-Virgin Islands (British)

vi-Virgin Islands (U.S.)

vn-Viet Nam

vu-Vanuatu

wf-Wallis And Futuna Islands

ws-Samoa

ye-Yemen

yt-Mayotte

za-South Africa

zm-Zambia