WARSAW UNIVERSITY OF TECHNOLOGY

FACULTY OF MATHEMATICS
AND INFORMATION SCIENCE

MASTER THESIS

COMPUTER SCIENCE

Combining Information from Multiple Internet Sources

Author: Jakub Stadnik

Supervisor: dr Marcin Paprzycki

Warsaw, April 2008

…… ……

podpis promotora podpis autora

POLITECHNIKA WARSZAWSKA

WYDZIAŁ MATEMATYKI
I NAUK INFORMACYJNYCH

PRACA DYPLOMOWA MAGISTERSKA

COMPUTER SCIENCE

Łączenie informacji z wielu źródeł internetowych

Autor: Jakub Stadnik

Promotor: dr Marcin Paprzycki

Warszawa, Kwiecień 2008

…… ……

podpis promotora podpis autora

Combining Information from Multiple Internet Sources

Abstract:

This thesis compares three approaches for combining information retrieved from the Internet: *Game theory* based*, Auction* based and *Consensus* based. To compare those three methods an application was created to aid extraction of information from multiple Internet sources and to combine it using one of these methods. Application utilizes the *JADE* agent platform, to retrieve and combine the information extracted from different search engines.

Each of three methods under consideration provides a different way to combine results obtained from multiple sources to create a combined view presented as a single list of results. Main aim of the thesis was to test the three methods; see what results each of them can provide and to compare these results. The thesis presents the results; discusses their effectiveness and subjective opinion on their coherency.

Approaches were tested as follows: at first the results which were provided by those methods were compared to results of search engines which were used for extracting the data in the first place. The results were investigated to see what URLs contributed to each of the final answers, and from which search engines those URLs originated. Afterwards, the results of methods were compared with each other, by means of investigation of the resources which were pointed to by the URLs comprising the result sets returned by those approaches. This approach to testing provided enough insight into the results, to be able to check their quality.

Łączenie informacji z wielu źródeł internetowych

Streszczenie:

Niniejsza praca porównuje trzy sposoby na łączenie informacji uzyskanych z różnych źródeł internetowych: opartego na teorii *gier*, teorii *aukcji* oraz teorii *konsensu*. W celu porownania tych metod, stworzona została aplikacja, która pozwala na pozyskiwanie informacji z rożnych źródeł, a następnie na łączenie uzyskanych danych, przy użyciu jednej z trzech wymienionych metod. Aplikacja wykorzystuje platformę agentową JADE, jako narzędzie do pozyskiwania i łączenia wydobytych wyników przez różne wyszukiwarki internetowe.

Każdy z badanych sposobów daje inne możliwości konsolidacji informacji pochodzącej z wielu źródeł w celu utworzenia skonsolidowanego zbiór odopwiedzi przedstawionego jako jedna lista wyników. Głównym celem pracy było przetestowanie trzech sposobów, ocenienie wyników, ktorych są one w stanie dostarczyć oraz porównanie ich sensowności. Praca prezentuje wyniki dostarczone przez sprawdzane metody, efektywność owych metod oraz subjektywną ocenę przydatności uzyskanych wyników.

Powyższe metody były testowane w następujący sposób: jako pierwsze, wykonane zostały testy porównujące wyniki uzyskane przez te metody do wyników pojedynczych wyszukiwarek. Wyniki zostały sprawdzone, pod względem zawieralności się zbiorów dostarczonych przez metody i zbiorów pojedynczych wyszukiwarek. Następnie, zbiory wyników poszczególnych metod zostały porównane między sobą, poprzez sprawdzenie zawartości stron, na które wskazywały uzyskane wyniki. Podejście to, dało wystarczający obraz wyników, który pozwolił na sprawdzenie jakości otrzymanych rezultatów.

Table of contents

1. Introduction

Retrieval of information is a task that is very often performed by computer users when they seek to broaden their knowledge of a specific topic. The task becomes more complicated with multiple search engines available and multiple methods of information processing used within those. How to choose the best engine? Is there such a thing as the best search engine? How to know that the answer that was provided is valid? Those and many other questions arise when a user is searching for information. Furthermore, why not combine the powers of existing search engines? Why not filter obtained answers so that the user would not have to scroll through 5 computer screens to find answer that is the most relevant?

In recent years there were some attempts to solve the aforementioned problems. For example Menczer [11] created and implemented *MySpiders* - a multi-agent system for information retrieval. In his work he has shown that using adaptive and intelligent agents provides significant advantages Another examples of multi engine search can be the *MetaCrawler* (www.metacrawler.com) and *Dogpile* (www.dogpile.com) which utilize multiple search engines (Google, Yahoo!, Ask.com, MSN Search and more) to provide combined results. Those however unlike Menczer's approach do not process the results but rather display them sorted according to the number of occurrences in different search engines and average rank taken from ranks of each result in different engines. There are also tools which provide single site access to many search engines, however do not combine the information, just provide simple interface to query each engine. An example of such site is *iTools* (www.itools.com/search) which provides a single text box for the query, but user is still required to choose the engine (s)he wants to use. Another example of meta-search is *KartOO* (www.kartoo.com) search engine which provides results on a map of categories. It retrieves and ranks the information according to the search engines and also categorizes it. An interesting approach is used in the *ixquick* (www.ixquick.com) multi search, where the user is provided with the information instantaneously, because this search engine provides the links from the fastest search engines at a glance. In other words, the user does not have to wait for all the engines to finish their extraction tasks, but as he/she is browsing the already retrieved results, the list is updated with more entries as the other search engines finish processing. Those approaches however, apart from Menczer's one, do not use more advanced combination algorithms. The

application presented in this paper combines the retrieved results using advanced methods.

In this approach application utilizes agents as experts in knowledge extracted from the results provided by search engines. It uses *JADE* agent framework to create a multi-agent environment which aids information retrieval. It is interfaced through a web page when one can input a query and wait for the result. In the core of the system agents serve as miners of data from multiple search engines. Simply saying, they possess the knowledge that we are apparently looking for. So, when we ask several experts the same question we expect that they will provide us with the best results possible. Would not it be better if they arrived at some agreement about the answer thus giving us a single combined opinion? This thesis aims at answering those questions. It investigates coherency and performance of each of the three methods used for filtering the answers so that the filtered result sets are smaller but hopefully more meaningful. Those methods are then compared with each other; to see which one provides the most promising results.

There are three methods utilized in this work for yielding the final results: *Game theory*, *Auction* and *Consensus*. In the *Game theory* approach the process of combining results is compared to a game in which the game players are agents that decide about the information destination by either discarding or keeping it. An *Auction* based approach is almost similar to the "real-life" auction, but here agents decide about which information can be obtained for the lowest price. *Consensus* method uses more centralized approach since it takes the highest-ranked information from all result sets and checks how common is this result among those obtained from all search engines. All of those methods are described in the thesis; their implementation and effectiveness are presented.

While consensus algorithm was already used as a tool for combining information from search engines [1], two other algorithms: *Game theory* and *Auction* were used for a different task - as methods of negotiation in Agent-based system for classification tasks - the *NeurAge* system, described in [7], [8] and [9]. Since *Consensus* based approach had been used as a way for combination of information retrieved from multiple Internet sources, there was no need for adapting the algorithm to our needs - it was used in the same manner as in [1]. Furthermore, in the *AGWI* system search engines were selected randomly – there were more search engines than agents utilizing those. On the other hand, *Game theory* and *Auction* required adjusting to deal with data fundamentally different from what they were dealing with in their original version. Adjustment details are presented further in the paper; mainly in chapter 3.

1.1 Aim of the thesis

Are the *Game theory* based, *Auction* based and *Consensus* based approaches a good way of combining the information obtained from multiple Internet sources? By creating an application

which allowed testing of these there approaches; this question could be answered, at least to some extent. Approaches differ in ways of data processing and combination yet they could be brought to a form that is unified and thus comparable. This unification process is described in chapter 3 which presents implementation of those three approaches. Preliminary testing of those approaches was performed and the results are presented in further parts of the document. Tests which were conducted provide a view on the implemented information processing methods – how they perform when compared to single search engines and to each other; which approach yielded the "best" results and which has proven to be the "weakest" one.

1.2 Thesis outline

Chapter 2 presents the design of the tool which was used to help in testing those approaches. Chapter contains the description of all communication between parts of the system as well as the test application implementation details and the general application work flow.

Chapter 3 presents the three approaches: *Game theory* based, *Auction* based and *Consensus* based methods for information combining. Chapter presents their background, implementation and workflows.

Chapter 4 presents the tests that were conducted during work on this thesis. This chapter presents results of the three approaches and their comparison under certain conditions. Chapter also contains the conclusions describing the test results.

Chapter 5 presents final remarks on tests and possibilities of future work concerning combining information from multiple Internet sources.

2. Design of the tool used for testing

This chapter presents the design of the tool that was developed to aid information retrieval and to test the algorithms for query processing. This is the first main step in the thesis; development of a simple tool which would aid in comparing the methods of combining the information retrieved from the Internet. The tool is a web application allowing for inputting the queries and selection of the response-composition method. The main engine of the application is the answer processor – it contains implementations of all three methods and it retrieves the information from the Internet. It was developed to aid testing of *Game theory*, *Auction,* and *Consensus* based approaches and to aid in comparing the results yielded by the search engines.

By creating a web application, its main aim was to make it easy to use; as an ordinary search engine. The difference from ordinary search engines is as follows: since this application was built for the purpose of testing of the answer processing methods, a method must be selected when issuing the query. Method is selected to provide the information about which combining algorithm is going to be tested. When a query is input and an algorithm is selected the application starts the retrieval and combining of the data. Afterwards, when the answer processing is finished its results are displayed as URLs which can be accessed as a hyperlink. Sometimes feedback to the application must be provided so that it can learn which answer was considered the most valuable and also to rank the search engines. Ranking procedure is very simple, yet it was implemented to learn which engines provide most commonly used results.

This chapter of the thesis will present the design of the tool that can be divided into two parts: application part and database part. The application part is written in *Java* language, while the database is the *MySQL* database.

The application is very simple since it was created only for testing purposes. It was not the main aim of the thesis and that is why it was written in a simple way. However, it had to be implemented since without it tests would be impossible to conduct. The following part presents the application part of the test application.

2.1 Design of the application part

The application part is written using the *Java* language and utilizes the agent platform – *JADE* which is also written using *Java*. The web application hence it requires the *Java Servlet Container* – for this purpose *Apache-Tomcat* was used.

To create the testing tools the *JADE* Agent platform was utilized. Agents provide us with an abstract layer which can reflect the real-life situations. Agents play the role of experts at knowledge about given queries by hiding from the user the presence of the real search engines. Though the usage of software agents was not necessary it was an interesting and easy way to write an application; even such one which is written only to aid in testing something different than a multi-agent system. Agents provide an easy interface of communication between objects and modules and that makes developing applications relatively easy. Though there is a variety of agent platforms like IBM *Aglets* or *ZEUS* the decision was made to use *JADE* since it is still actively developed whereas other agent frameworks are not. Summarizing, the tool could be written without any agent platform, but development of agent based environments is fairly easy and, at the same moment, very elegant.

Next part presents the detailed design of the application. In general the application can be divided into two modules: *Client* and *Main*. The *Client* module is responsible for serving user requests and forwarding those to the web part of the application. *Client* module is responsible for interacting with the end-user; The *Main module* receives requests from the *Client* module and manages necessary agents for information retrieval and processing of the retrieved results. The *Main* module could also be accessed from any application (standalone/web based) as its entry point is started as a separate thread in the application.

Next part of the chapter describes the *Client* module design details. Its information flow, workflow and functionality will be described. Afterwards, in the next part of the chapter, in similar manner, the *Main module* will be presented.

2.2.1 Design of the *Client* module

2.2.1.1 General description of the *Client* module

The *Client* module consists of a web application, through which the queries can be submitted and processing results can be viewed. After submission of the query the web application creates a new *Main* module entry point and sends search parameters through the agent platform to the processing engine which starts the search process. Then the web browser waits for the search process and data combination to finish. After the search process is finished, the web application receives the results yielded by the selected algorithm and displays them. A list of results is then presented containing 10 results yielded by the algorithm selected at the beginning. Depending on the algorithm outcome it may happen that feedback will be required to finish the data processing. If that is the case, the application expects to receive the URL, which is chosen as the best answer from the set of answers returned by the processing engine. URLs can be viewed; the best one can be selected and therefore marked as feedback. Afterwards, when feedback is provided to the application it is ready to process another query. Providing feedback is not necessary – it is collected only to rank the search engines; which was not however main aim of this work.

The following Use Case diagram presents possibilities of the user and introduces essential components. It also presents the operations which can be performed by those components.

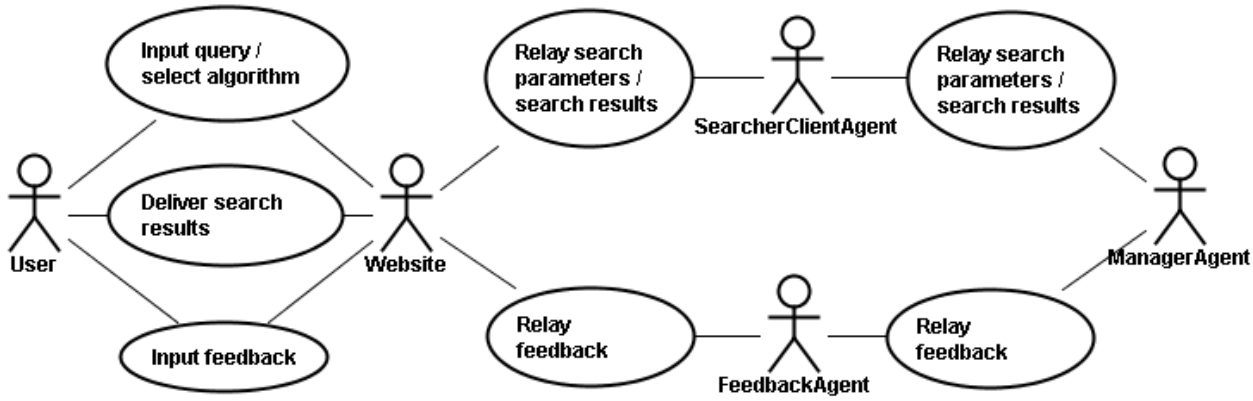

Fig 2.2.1.1 Use case diagram of *Client* module

Next part of the chapter presents the implementation details of the *Client* module. This part contains description of detailed implementation part and the workflow of this module. At the beginning of this section short descriptions of the components comprising this module are provided.

2.2.1.2 Implementation details of the *Client* module

The listing 2.2.1.1 presents the short descriptions of the components already introduced in the Use Case diagram (Fig. 2.2.1.1) and also some components that have not yet been described. Also the transfer objects which are utilized by the components are presented in this listing.

MAIN COMPONENTS:

GatewayServlet **–** this component is the backend of the web application. It is responsible for processing user requests and serves as a web application controller. Its purpose is also to forward user requests to the

SearcherClientAgent. This object is a derivation of the **HTTPServlet** class provided in the *Java Enterprise Edition API.*

SearcherClientAgent.– this component is responsible for the creation of the entry point to the *Main module*. It creates the necessary ***ManagerAgent*** as an entry point to the application and forwards user requests to it. It is also responsible for receiving result list after it is finished being processed. This object derives from the **GatewayAgent** class provided by the *JADE* framework.

FeedbackAgent – this component is responsible for sending user feedback (if such is required) to the application, after the results are presented. This object derives from the **GatewayAgent** class provided by the *JADE* framework.

ManagerAgent **–** this component is not a part of the *Client* module, however since it is known in the *Client* module, its purpose in this module will be described. It serves as an entry point to the *Main module* which in turn is responsible for the information retrieval and processing. From the *Client* module point of view it only receives query, returns results and sometimes receives feedback. This object derives from the **Agent** class provided by the *JADE* framework.

TRANSFER OBJECTS:

SearchParams **–** this transfer object serves as container to relay the search parameters provided by user into the lower parts of the application. This object contains the query and the algorithm name which are then relayed to the further parts of the application.

HTMLTagA – this transfer object is used as a container with which the feedback which user may provide is enclosed. This class is a simplified representation of

Listing 2.2.1.1 Components of the *Client* module

The following part presents the information flow and components interaction in a more detailed way. This section also presents the way in which the transfer objects are utilized in this

module. At the end of this section the sequence diagram which depicts the information contained in this section is presented.

As the first step of the process the user provides the query and the algorithm name as HTTP request parameters. Those are read and checked by the *GatewayServlet* object which displays a message if the parameters are invalid. Then the *GatewayServlet* wraps the parameters into a *SearchParams* object, sets those as an HTTPSession attribute and forwards session object to the *SearcherClientAgent*. The *SearcherClientAgent* then creates a new *Manager Agent* (*MA*) with a random name and stores its AID as a session attribute. Then it forwards the created *SearchParams* object to the *Main* module entry point - the newly created *MA*- and waits for the response from the agent. Afterwards, the *MA* receives *SearchParams* which is in turn unwrapped providing the query and the algorithm. Then search engines are queried and algorithm finishes processing the List of results is returned to the *SearcherClientAgent*. Depending on the case if the algorithm was able to yield the answer or not, the resulting webpage will contain buttons to provide feedback for specific URL. If the webpage contains no buttons for feedback, the process is finished. If, however, the webpage contains buttons for providing feedback, the user may view web pages under those URL and then provide feedback by clicking the button next to the URL he chooses as the best. Then the URL is forwarded to the *GatewayServlet* object as HTTP request parameters. The first parameter contains the link name and the second contains the actual URL. Those parameters are then wrapped into the *HTMLTagA* object, set as a session attribute and forwarded to the *FeedbackAgent*. *FeedbackAgent* forwards the *HTMLTagA* object to the *Main* module (*MA*) which can finish its processing tasks. After finishing processing tasks *MA* is still alive waiting for another request. However when the server session has ended the *MA* is destroyed immediately.

The following sequence diagram depicts the information described in the above part. This diagram can be viewed as a summary of what was described before as it presents the workflow and information flow between components of this module.

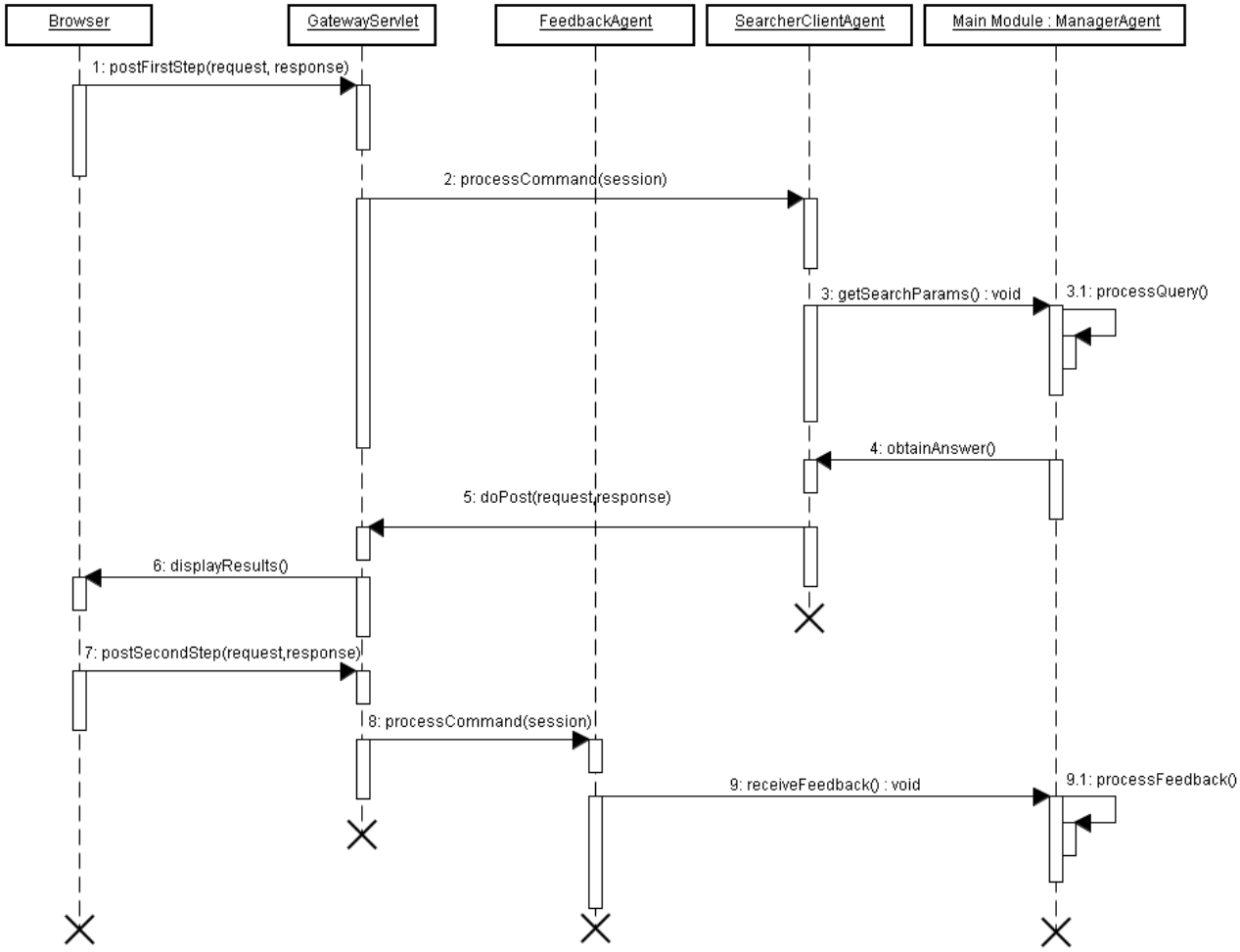

Fig 2.2.1.2 Sequence diagram of *Client* module

This diagram concludes the description of the *Client* module. Next part of the chapter presents the *Main module* of the application part.

2.2.2 Design of the *Main* module

2.2.2.1 General description of the *Main module*

The *Main* module is an application that processes user requests which are received from the *Client* module. Upon reception of request module performs search utilizing search engines and processes (combines) its results using one of the selected methods. This module consists mainly of two crucial components which are agents – *Manager Agent* (*MA*) and a set of *Search Agents* (*SA*). However those two utilize other, smaller components, which do not exist without agents. All of those components are described in the latter part of this section. The following part will present the general description of this module.

At the beginning there is only the *MA* created by the *Client* module. It waits for the input – a query and processing algorithm of choice. When the input is received the *MA* sets search engines up, to be ready for querying. Then several *SAs* are set up by the *MA*. *MA* sets up as many *SAs* as

there are set up search engines. Afterwards *MA* forwards the query to the each of the *SA*. Each *SA* is assigned a different search engine and *MA* controls engines assignment process.

Having the search engines assigned, the *SA* can start querying the engines. After search is finished, the *SA*s return their result sets to the *MA,* which starts processing results according to the algorithm chosen by the user at the beginning. When answer processing is finished, the *MA* sends the final results to the web application which displays them on the webpage. There are two possibilities depending on algorithm outcome. If the algorithm was able to find the best result, the result list is displayed and knowledge base is updated instantaneously. The search engine which yielded the final result is ranked as the best and other engines are ranked according to how close they were to this engine. If, however, algorithm was not able to yield one answer; application presents a list of possibilities and displays an option to provide feedback - selection of the answer that is the most valuable (subjectively, of course). After the feedback is received application ranks the engines according to it. Engines ranks are stored in the knowledge base – for each query, search engine and method of answer processing, there are engine ranks. After *MA* calculates the weights those are sent to the *SAs* which update the knowledge base with ranks (weights) of the engine they were assigned at the beginning and application can be issued another query.

In order to present to the user readable output, the vast majority of data is saved to local files. Only the top 10 results are displayed on the web page. The action of saving is done by *Manager Agent*. More precisely: information which is saved contains 10 top results from result sets of each search engine as well as the final result of answer processing algorithm.

Below is the use case diagram which presents what are the operations of the agents. It also depicts interaction with the application, however without getting into the details of the *Client* module.

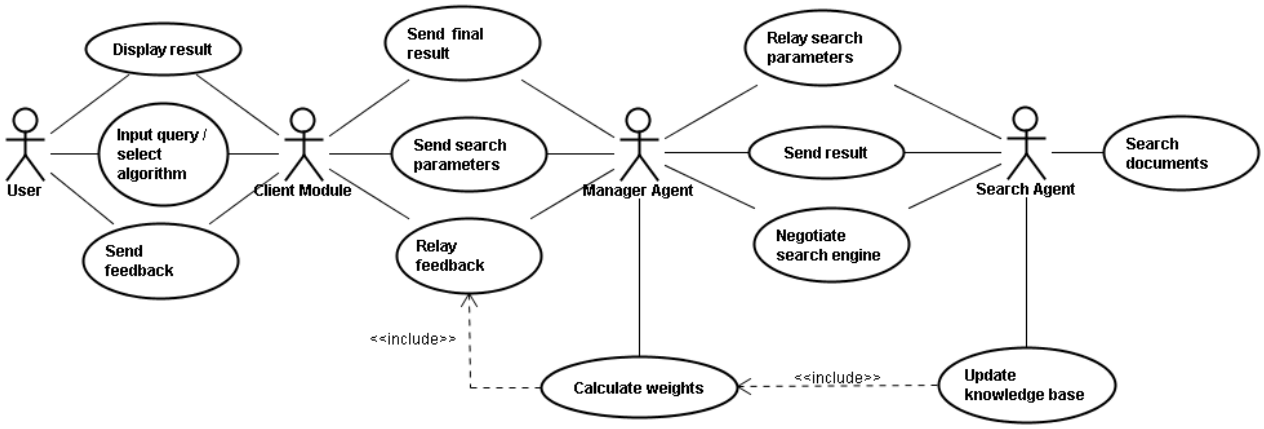

Fig 2.2.2.1 Use case diagram of *Main* module

2.2.2.2 Implementation details of the *Main* module

This part of this sub-chapter presents the implementation details of the *Main* module. Following listing presents the components comprising this module. This listing will introduce all of the utility classes and transfer objects as well, since those are essential to understand how this module works.

MAIN COMPONENTS:

ManagerAgent* –** this component is the main component of the *Main module*. It setups all ***SearchAgents, obtains results from the search engines and invokes methods necessary for results processing. It is also a only possible entry point for the Client application to process its requests. This object derives from the **Agent** class provided by the *JADE* framework.

SearchAgent* –** this component is responsible for proper querying the search engines using other smaller components. It is created by the ***ManagerAgent, so it cannot exist without one, but there is no direct association between those. Once ***SearchAgent*** is created it exists as almost independent entity. The only dependence is, however, when its creator (***ManagerAgent***) is destroyed he is destroyed at the very same moment.This object derives from the **Agent** class provided by the *JADE* framework.

TRANSFER OBJECTS:

SearchParams* –** this transfer object is received by the ***ManagerAgent at the very beginning of the search process. It contains the query and the result sets processing algorithm which are to be used during the search and results combination process.

SearchEngine – this transfer object represents the search engine which is used to extract the information from the Internet. This class contains several helper methods, however those are related to itself and do not process any other high level components and that is why it was classified as a transfer object. This object is assigned and passed by the ***ManagerAgent*** to the ***SearchAgents*** at the beginning of the search process.

HTMLTagA – this transfer object represents the HTML tag <a> and it is comprised of the most essential parts of this tag, that is its value (link name) and its href attribute (actual URL). This class is passed many times throughout the variety of objects, from the beginning of the search process to its very end (feedback).

Listing 2.2.2.1 Main components and transfer objects used in the *Main module*

UTILITY CLASSES:

AgentController – these objects are used to control the ***SearchAgents*** lifespan. The ***ManagerAgent*** controls the life of its ***SearchAgents*** through the objects of this class. Each ***AgentController*** object is responsible for controlling one corresponding ***SearchAgent***. Objects of this class can create, destroy and suspend the agent. This class is provided by the *JADE* framework.

SearchEngineExplorer – objects of this class are used to utilize ***SearchEngine*** classes during the search process. Those objects are responsible for filtering URLs which should not be considered as results when the search results are received.

LinkExtractor – objects of this class are responsible for retrieving the search results from the web. This is class is actually the class which queries the search engines and parses the result pages provided by the engines.

Listing 2.2.2.2 Utility classes used in the *Main module*

When the *MA* receives the *SearchParams* object from the web page it is unwrapped into two separate parameters. A *List<String>* is a query divided into single Strings where each is a phrase from the query and *String* which is the selected processing algorithm name. Having the search parameters unwrapped, *MA* gets the *SearchEngine* objects from the database. Each *SearchEngine* configuration is stored in the database since each search engine utilizes different search parameters, such as HTTP parameters for controlling its output, that is parameter controlling number of displayed results and parameter that states the actual query. Also to each of the search engines there is a list of URLs which should be ignored when processing given web page with results. Those URLs are resources which are related to the given search engine, but not directly to its search process. For instance *Google* search engine has many hyperlinks pointing to Maps search, Image search and other services. Those URLs should not be considered as a part of the search results and therefore should be ignored during the URL retrieval process. Information about which URLs should be ignored is stored in the database and is extracted during *SearchEngine* creation process so that each *SearchEngine* has its list of "to be ignored" URLs assigned. Later on, when setup of the *SearchEngines* is finished those are stored in memory of the *MA* since *SearchEngines* are needed in further steps. Afterwards *MA* creates multiple *SA*s. The creation process is dynamic: there are to be as many *SA* created as there were *SearchEngines* retrieved from the database. During this creation process *MA* creates *AgentController*s. Each *AgentController* is used to control on "real" *SA* and it immediately starts their lives. Afterwards, when *SAs* are started they wait for the query and

algorithm that will be used during search process. Then *SA* queries the database with the given query and algorithm to retrieve the weights set which are used during data processing by the algorithms. Weights are the ranks of the search engines; computed based on previous algorithm results. Their values vary from 0 to 1 depending on how the algorithm evaluated result set of some particular engine. If engine performed badly – results were not satisfactory in the sense of the algorithm; it is assigned a smaller weight than the engine which results were considered as better ones in the algorithm's sense. If this is a first time the application is issued a certain query ranks are set to 1. Those weights are used during ranking processes – it used to give better chances to the URLs which originate from engines, which contributed more to the previous results of the algorithm.[1]

To query a search engine the following process is performed:

1. *SA* creates a *SearchEngineExplorer* object that will create request to a search engine. *SearchEngineExplorer* creates specific query based on *SearchEngine* and phrases passed to its constructor. Every search engine has its unique URL so it is up to *SearchEngine* to create query provided the phrases. *SearchEngine* returns ready to use URL to which the *SearchEngineExplorer* indirectly connects.

2. *SearchEngineExplorer* creates a *LinkExtractor* that connects to the URL which was created by the *SearchEngine*. *LinkExtractor* job is to extract all HTML tags of the form AAA and translate those to the *HTMLTagA* objects. When those are returned as a *List* of *HTMLTagA* objects *SearchEngineExplorer* compares those versus the URLs that should be ignored. The URLs to be ignored are provided by *SearchEngine*. That is every URL that is on the ignore list should not be returned to the *SA*.

3. *SearchEngineExplorer* returns the *List<HTMLTagA>* to the *SA,* which in turn return answers to the *MA*.

4. *SA* waits for the weight from the *MA*.

5. After reception of answers list from all involved agents, *MA* runs answer processing algorithm selected at the beginning.

6. There are two possible outcomes of the algorithm:

   a) Algorithm finished processing and list of answers is present. The *MA* sends the list immediately to the web application and final set of answers is displayed. At this point *MA* writes information about the result sets to the local drives and sends weights to the corresponding *SAs* which in turn can update knowledge base. Afterwards, *MA* is ready to receive another query.

[1] Ranking of search engines was disabled during the tests which are described in chapter 4. It was implemented as a future possible extension of the application - which may include rankings of the search engines as a part of the tests.

b) If initial step of the algorithm is *null* (algorithm could find result to user query), *MA* creates a combined list of answers from the all result sets of all agents and sends it to the web application. After results are retrieved feedback to application should be provided so that *MA* can calculate weights and send those to the *SAs*. After feedback is received by *MA*; it calculates weights and send those to the corresponding *SAs* which in turn update the knowledge base.

7. It may happen however that *MA* receives the *SearchParams* object instead of the feedback which is an *HTMLTagA* object. Then *MA* immediately starts processing another query. *MA* is terminated the same moment the session was destroyed on the web server. Up to this moment *MA* can process multiple different queries.

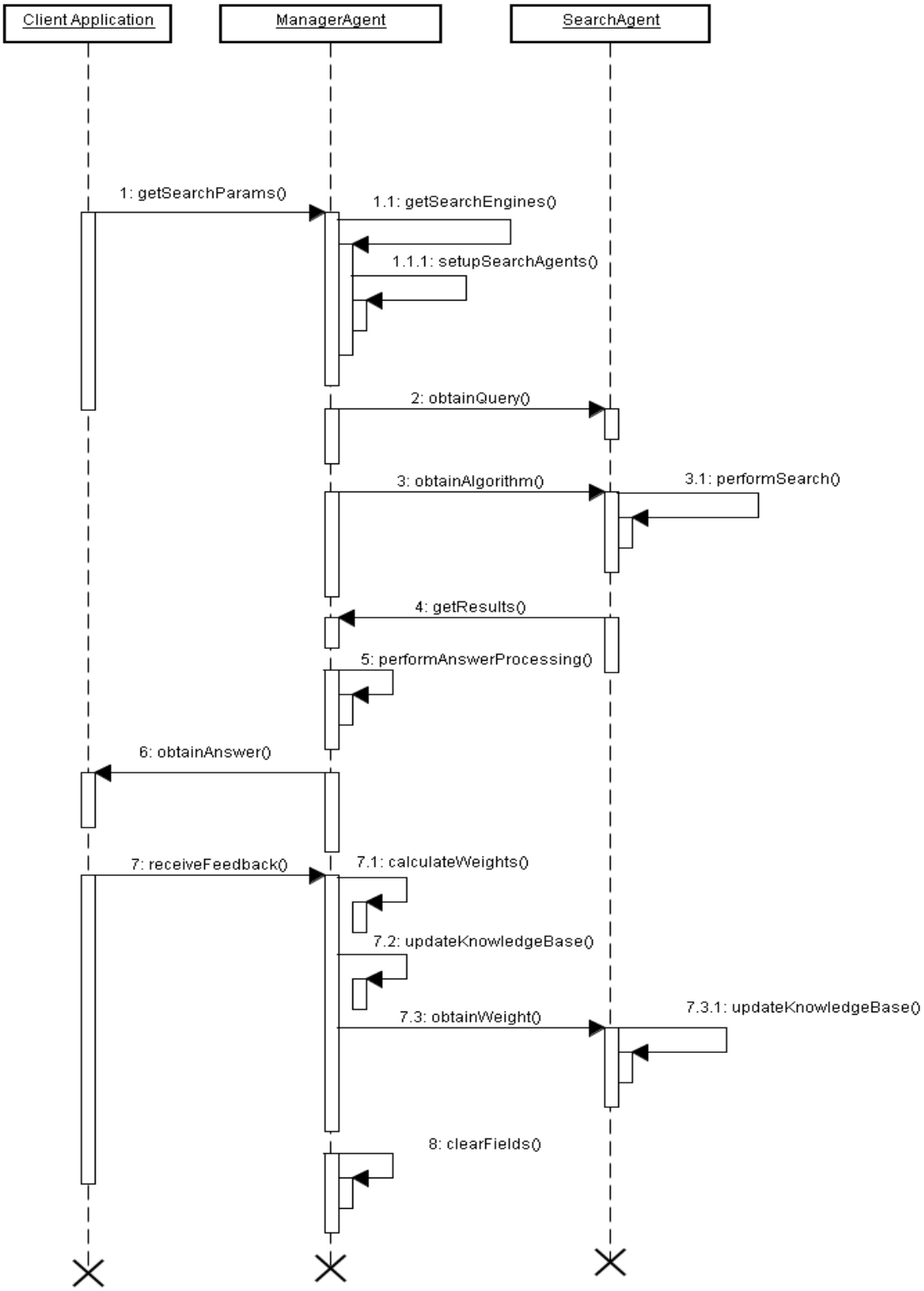

Fig 2.2.2.2 Sequence diagram of the *main* Module

Diagram in Fig 2.2.2.2 concludes the description of the *Main module* and the description of the application part. Next sub-chapter is database part description. Diagram presents database design and provides descriptions of the tables which are used in the database.

2.3 Design of the database part

This chapter presents the database that is used to store the necessary data for the testing purposes. Database stores the information that is necessary for application configuration, such as configuration of the search engines and history of queries, which were issued together with engine rankings for specific queries and algorithms. Surely this data could be stored in the other way, in files for instance, but since this is not the most elegant, easy to modify and efficient way of information storage, *MySQL* relational database was used. Also, any data modifications are easier than in case of using plain files.

Database schema of the application is not complex, but still this is enough to store necessary data. Storage is accessed from *Java* application, namely the *Main module* through the *JDBC* drivers. Schema is constructed in very simple way; there are no computations performed on it, there are no stored procedures defined in the schema. Those are not needed as application deals with very simple data. Instead of stored procedures plain SQL and DDL statements are used to retrieve and update the data contained in it. Next part of the sub-chapter presents the database schema implementation and descriptions of the tables in the schema.

As stated earlier this schema is very simple and consists only from four tables. Furthermore there are no stored routines implemented. This makes this schema simple, yet robust enough to store information necessary for the testing purposes.

Fig 2.2.1 depicts the database schema diagram (ERD – Entity Relationship Diagram). It presents the tables contained in the schema and relations between these (foreign keys).

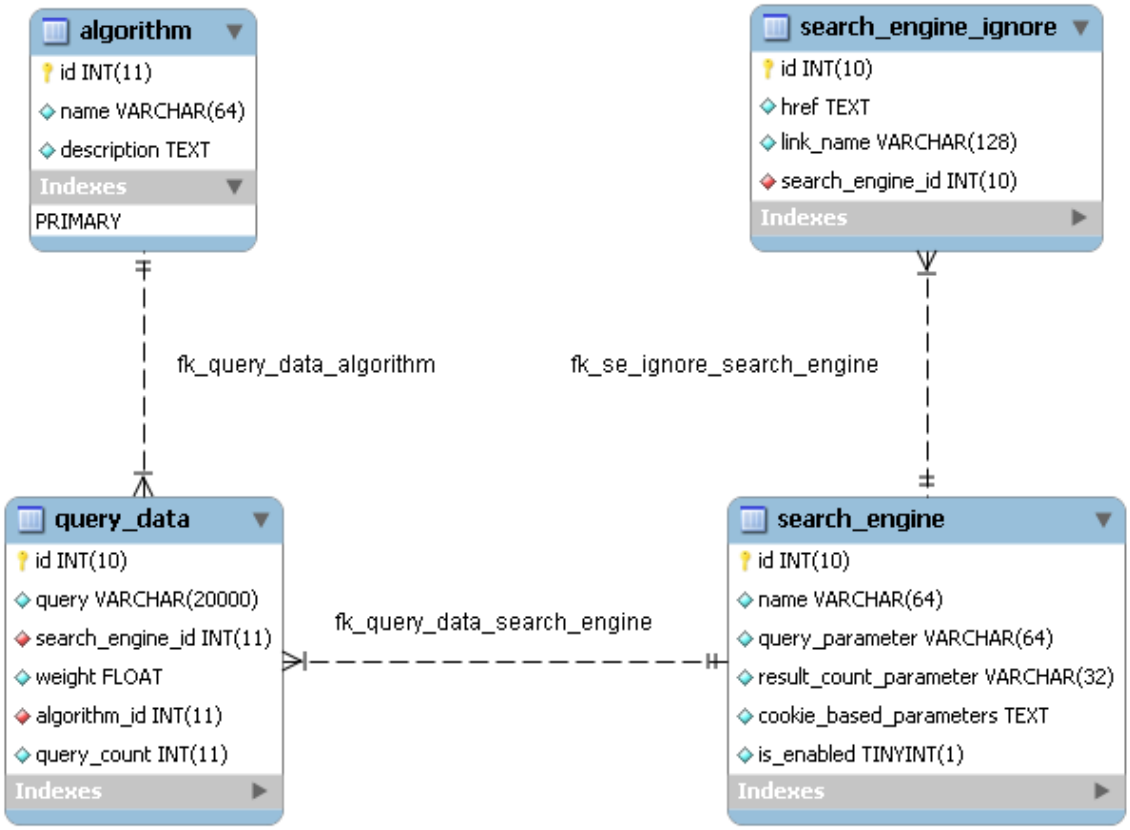

Fig 2.2.1 Database schema diagram

Listings below present short descriptions of tables contained in the database. Each table has its columns described as well as its purpose. Relations between tables can be seen on the above diagram, so those are not mentioned directly in the database tables' descriptions.

Table *algorithm*:

This table is a dictionary table. It contains algorithms names and descriptions. It is also used to relate the query history with a particular algorithm.

Columns:

- ***id*** – Primary Key
- ***name*** – Name of the algorithm
- ***description*** – (Optional) Description of the algorithm

Listing 2.2.1 *Algorithm* table description

Table *search_engine*:

This table contains definitions of search engines. It contains all necessary data to construct specific queries which in turn can be used to extract answers from the search engines.

Columns:

- ***id*** – Primary Key
- ***name*** – Name of the search engine
- ***query_parameter*** – Contains base URL of the search engine, used when constructing queries
- ***result_count_parameter*** – Contains name of the HTTP request parameter used to manipulate number of results displayed per page
- ***cookie_based_parameter*** – Some of the search engines store user preferences in cookies rather than, for instance, allow the user to supply HTTP request parameters to manipulate the results

Table *search_engine_ignore*:

This table contains lists of pairs (Link Name, HREF) which should be ignored when parsing page of search engine with results on given query. When parsing the HTML document system should not consider all buttons, URLs, etc. which are not connected to the query (for instance on Google page one can find hyper links to Google Maps, Google News, etc. which should not be considered as a result)

Columns:

- ***id*** – Primary Key
- ***href*** – URL to be ignored (if empty system will ignore all pairs with the given link name)
- ***link_name*** – Name of a link to be ignored (if empty system will ignore all pairs with given HREF)
- ***search_engine_id*** – (Foreign key) Specifies which search engine should use the given pair

Listing 2.2.2 *Search_engine* and *search_engine_ignore* tables descriptions

Table *query_data*:

In fact whole knowledge base is stored here. This table contains a search engine ranking based on previous system inputs.

Columns:

- ***id*** – Primary Key
- ***query*** – Query that was input by user
- ***search_engine_id*** – (Foreign key) Specifies which search engine yielded this result
- ***weight*** – A ranking parameter – higher the better results the search engine provided
- ***algorithm_id*** – (Foreign key) Specified which algorithm was used to yield this specific result
- ***query_count*** – Number of updates on the specific configuration (algorithm, SE, query); it is used to average weights that are submitted on the specific configuration

Listing 2.2.3 *Query_data* table description

3. Algorithms

This part of the thesis provides the detailed description of the three information combination approaches. This chapter also describes any helper routines that are used by those approaches. Each of the main algorithms for obtaining the final answer has its pseudo code included as well as the activity diagrams.

3.1 *Game theory* method

This sub-chapter presents the *Game theory* method. This algorithm was used before in the *NeurAge* system [9] and had to be adapted to suit the purpose of combination of the data retrieved from the Internet. In its original form agents were supposed to vote for certain classes of data; here, they are voting for certain URLs. The confidence values from the original algorithm have been replaced by the URL ranks according to the algorithm described in section 3.4.1. Also, in its original form agents were yielding one class as the final answer. In the adapted form agents are returning 10 URLs in sequence, where any next iteration starts the whole process from the beginning, however without processing the URL which was already selected. This does not violate the main assumptions of the algorithm and this was stated by Edyta Szymańska of Emory University, Atlanta by means of personal communication.

In general, a game is defined as follows: it consists of set of players, set of moves (strategies) and specifications of payoffs for each combination of moves. In case of algorithm which will be presented in further section game is a normal form game that is defined as follows:

There is a finite set P of players, which we label $\{1,2,...,m\}$.

Each player k has finite number of pure strategies (moves)

$$S_k = \{1, 2, ..., n_k\}.$$

A pure strategy profile is an association of strategies to players, that is *m*-tuple

$$\vec{\sigma} = (\sigma_1, \sigma_2, ..., \sigma_m)$$

such that

$$\sigma_1 \in S_1, \sigma_2 \in S_2, ..., \sigma_m \in S_m$$

Let strategy profiles be denoted by Σ

A payoff function is a function

$$F : \Sigma \to \Re$$

whose intended representation is the award given to a single player at the outcome of the game. Accordingly to specify a game the payoff function has to be specified for each player in the player set $P = \{1,2,...,m\}$.

Definition. A **game in normal form** is a structure

$$(P, S, F)$$

Where $P = \{1,2,...,m\}$ is a set of players, $S = (S_1, S_2, ..., S_m)$ is a *m*-tuple of pure strategy sets, one for each player and $F = (F_1, F_2, ..., F_m)$ is a *m*-tuple of payoff functions.

Listing 3.1.1 Definition of the normal form game

In this game components are as follows: players are agents, possible moves are change or keep the URL; payoffs for those moves are defined as a 2x2 matrix. Each agent is assigned two values: one for the keeping the aforementioned URL and one for changing the selected URL. Those values may or may not change each round of the game, depending on the previous round outcome.

At the beginning of the process, the results obtained by *Manager Agent* from *Search Agents* are filtered, ranked and updated according to the algorithm from section 3.4.1. The URL ranking represents how the agents are confident about a certain URL. From this point the game starts.

The game proceeds as follows. In each round there are two agents selected. Those two agents are those, which were assigned the result set with the highest ranked URLs. The highest ranked URL is found as follows: if there is an URL which has, for instance, rank equal to 20 and there are no URLs with higher rank (taking into account all result sets) then this is a highest ranked URL. After first agent is found we search for the second agent which has the second highest ranked URL, but this time omitting the result set which is assigned to previously selected agent. Selected agents present their highest ranked URLs and have two possibilities: either to keep their answer or to change it. If the keep action has higher value than the change action, the agent will be assigned the action to keep its URL for the next round. If, however, the agent is assigned the action to change its URL and the second agent is assigned the action to keep its URL, the latter is considered a winner of the round and the former is considered to be the loser – it and its result set are discarded from further considerations. Then the next round starts (without the agent, which was removed in previous round - that implies removing the result set assigned to it) and so on, until there is only one agent with his assigned URL. After that the game is restarted – every agent takes part in the negotiation process once more, however the URL that was selected as the winning in previous negotiation is removed from the further consideration from all result sets. Negotiation is performed in the same manner – round by round agents are removed – but this time they play without the URL that was selected in the previous "big" round. Process of game restarting continues until there are 10 URLs selected. That is; there are 10 "big" rounds; each being a separate negotiation of one URL; resulting in 10 URLs being selected and ordered.

The following listing presents one "small" round of negotiation process. That is the negotiation between two agents. This example presents calculations which are performed during each round of negotiation process, and what the possible outcomes of the algorithm are; when approaching this particular situation.

Example:

Let us consider the following initial ranking values:

Answer	Agent 1	Agent 2
A	35	20
B	10	30

Agent 1 is assigned URL A as the highest ranked. Agent 2 is assigned URL B as the highest ranked.

Then the keep payoff matrix would look like following:

Agent 1	Agent 2
35 – 10 = 25	30 – 20 = 10

And the change payoff matrix:

Agent 1	Agent 2
(35 + 10) / 2 = 22.5	(30 + 20) / 2 = 25

So in the following situation the Agent 1 is assigned keep action whereas Agent 2 is assigned change action. Therefore Agent 1 is winner in this round and Agent 2 is loser. Thus Agent 2 and its result set are removed from the further consideration until game restarts for the next “big” round.

After that Agent 1 updates the rank of the URL it was assigned for this round with the keep action payoff. That is URL A is no longer ranked with 35 but with 25 which was the keep action payoff.

At this point it may happen that this URL is no longer in the two of the highest ranked URLs. In this case it is now the second highest ranked one:

Listing 3.1.2 Example of *Game theory* round flow process

Example cont:

Let us consider the following ranking values in the next round:

Answer	Agent 1	Agent 3
A	25 (keep payoff from the previous round)	20
B	10	23

Agent 1 is assigned URL A as the highest ranked. Agent 3 is assigned URL B as the highest ranked.

Then the keep payoff matrix would look like following:

Agent 1	Agent 2
25 – 10 = 15	23 – 20 = 3

And the change payoff matrix:

Agent 1	Agent 2
(25 + 10) / 2 = 17.5	(23 + 20) / 2 = 21.5

So in the following situation the Agent 1 is assigned change action and so is Agent 3. This situation results in the draw – however to yield a winner the initial rankings (the ones from the beginning of the algorithm) of the chosen URLs for this round are compared. Agent's 1 ranking of its assigned URL is equal to 35. Agent's 2 ranking is equal to 23; which results in the following situation: Agent 1 is winner in this round and Agent 3 is a loser. Similarly as in previous round; where the Agent 2 was removed; Agent 3 and its result set are removed from the further consideration until game restarts for the next "big" round.

After that Agent's 1 rank of its assigned URL is updated with the keep action payoff. So Agent 1 is entering the next round with the URL A is ranked with 15 which was the keep action payoff.

As in previous case; at this point it may happen that this URL is no longer in the two of the highest ranked URLs. Then Agent 1 may not necessarily take part as negotiator in the next round of the process which continues until there is only one agent with its assigned URL remaining.

Listing 3.1.3 Continuation of the example of *Game theory* round flow process

During algorithm processing there is no direct participation of the *Search Agents* (*SA*s) in the game itself. The whole game logic is performed by the *Manager Agent (MA). MA* invokes all necessary methods to perform the game. *SAs* in this game are used as grouping factor for result sets – each agent corresponds to the search engine which returned a particular result set; and then result set is assigned to an agent. Algorithm was centralized for unification purposes (*Consensus* method described in 3.4 is also highly centralized algorithm) and also to achieve greater reliability and speed that could be seriously lowered due to the communication overhead or communication failures. In fact *SAs* are not necessary; those were used as a somehow interesting way to deal with the information retrieval task.

Below the pseudo code for the main game part is presented. This part is started after the algorithm from section 3.4.1 is finished and the result sets have been processed.

***Input*: Map containing URL rankings**
***Output*: 10 URLs**

BEGIN

1. repeat until there are 10 **URLs** in **answer list**
2. repeat until one agent remains
3. find agent whose **URL** is the highest ranked **URL**, find also the aforementioned **URL** – let those be **FA** (first agent) and **FAU** (first agent URL)
4. find agent whose **URL** is the second highest ranked **URL**, also find the aforementioned **URL** – let those be **SA (second agent)** and **SAU (second agent URL)**
5. construct keep and change payoff values as follows:

$$FA_{keep} = rank(FA, FAU) - rank(FA, SAU)$$

$$SA_{keep} = rank(SA, SAU) - rank(SA, FAU)$$

$$FA_{change} = \frac{rank(FA, FAU) + rank(FA, SAU)}{2}$$

$$SA_{change} = \frac{rank(SA, SAU) + rank(SA, FAU)}{2}$$

   determine agent actions by comparison of their values – the action with higher value is the chosen action
6. determine round winner:
   - if action assigned to one of agents (**FA**, **SA**) is keep action and other is change the one that selected keep is marked as winner, the second one is marked as loser and is discarded from further game
   - if both of them are assigned the same action their **URL** ranks are replaced by the values of the chosen action; if this situation occurs second time the following takes place:

     Depending on the initial ranks of the URLs assigned to the agents the one with the higher ranking is considered to be a winner of the round, and the second one is loser. Then the loser and its result set is discarded from the next rounds of the negotiation until the game is restarted (2.)
7. add **URL** to **answer list**
8. remove the **URL** from further evaluation
9. go to **2** (next round)

END

Listing 3.1.4 *Game theory* main algorithm

It may happen that the algorithm from listing 3.1.4 will not be started at all; in case of disjoint result sets. If it is so, *Manager Agent* creates the combined result set from all result sets (without repetitions) and such is returned to the web application. *Manager Agent* iterates through each of the initial result sets, takes every URL which is not already in the "big result set" and populates the set with this URL. Such situation occurs very rarely, but if one takes only 2 engines – one of Polish origin and one of English and issue a specific query such situation may happen – the result sets will be totally disjoint. If such situation happens, feedback should be provided so that the application can calculate the weights (listing 3.4.2) according to the URL provided as feedback. Weights are then used to rank the search engines. Each of the URL presented can be opened, look through its content and then finally the URL can be marked as the best one. Afterwards the weights are calculated using the URL which was marked as the feedback and which is an anchor to the weights calculation algorithm.

It may also happen that algorithm from 3.4.1 will remove some result sets from the evaluation. The result sets which were completely disjoint with other result sets are removed as being unsuitable for negotiation. This process needs an opinion of every search engine on each URL that is a part of the competition. If, however, the sets are joint in even one URL they are considered suitable for negotiation and algorithm 3.4.1 will update those with the URLs those are missing. The missing URLs are taken from the result sets of other search engines. Then the main part deals only with the result sets that were left. After the main part is completed, the search engine ranking can be performed by invoking the algorithm from listing 3.4.2 for weights calculation, taking the first yielded answer URL as an anchor point for the calculations. Then the weights are sent to the corresponding *Search Agents* which in turn update the knowledge base with weight of the his search engine selected at the beginning and for this particular query. Simultaneously, the final result set containing 10 selected URLs is returned.

Below is an activity diagram that represents the algorithm processing flow. This diagram presents processing flow as a whole. The diagram is divided into three parts in order to expose application responsibility during the process.

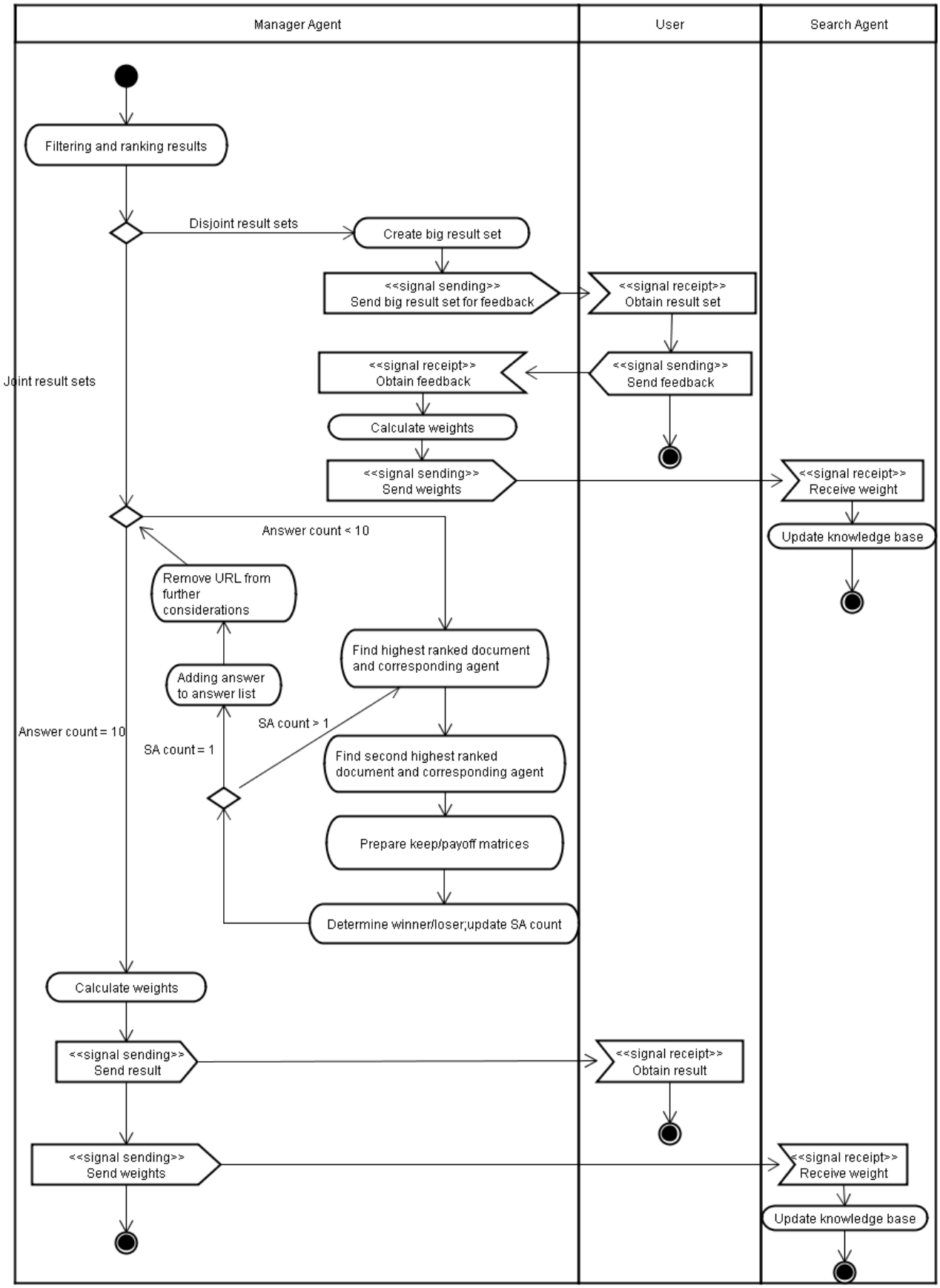

Fig 3.1.1 *Game theory* Method Activity Diagram

3.2 Auction method

Auction method as its name tells is an auction adjusted to be used in the thesis. Unlike real life auction the one which is implemented consists only of buyers. They try to reach an agreement on price of the commodity (URL) – select the product with lowest price.

This approach was also used before in the *NeurAge* system [9], and has been adapted to be usable for purpose of this thesis. Like in the algorithm described previously, the *Auction* method in its original form was about agents voting about the classes of data. In this adaptation classes were replaced by URLs and agents are supposed to vote about those. This method also returns 10 distinct URLs like the *Game theory* method with the same assumptions concerning correctness.

In each round of the auction each agent has its product (URL) assigned. Afterwards, the "cost" for each assigned URL is calculated. Costs are compared and the agent with the highest cost is considered to be a loser. Afterwards, the confidence values for selected URLs are updated by subtracting the cost from their value. Henceforth, the next round takes place. If the agent that was marked before as a loser loses again, it and its result set are discarded from further negotiation process - each agent has two chances before being removed. After removal, the process enters its next round, and so on until one agent remains with his selected answer. This is repeated 10 times and therefore it presents the same as the *Game theory* based approach way of evaluation of its final answer. As in the *Game theory* algorithm; each time the URL that was selected already as a part of the final answer, is not included in next "big" rounds of the *Auction* process.

In this algorithm (as in the previous one – the *Game theory*) the URL ranking is used as a base for all calculations. Therefore at the beginning of the main process we apply the algorithm from section 3.4.1. The main process may start or may not, depending on outcome of this algorithm. *Auction* method also requires that search engine has opinion on every URL that takes part in the negotiation process, just like the *Game theory* approach. The result sets which do not contain a particular URL are updated with the aforementioned URL. If the results sets are completely disjoint the combined result set created from all result sets without URLs repetition will be returned. In this case feedback to the application should be provided so that the search engines can be ranked. If it is not the situation, then the main part of the *Auction* starts and afterwards 10 results will be yielded at its conclusion.

Listings 3.2.1 and 3.2.2 present the example of flow of the one round which is a part of *Auction* method. All calculations that take place in the round of the *Auction* are presented in this example; on real numbers. This example also shows how the algorithm behaves in particular situations.

Example:

Consider the following initial ranking

Answer	Agent 1	Agent 2	Agent 3
A	35	20	25
B	10	30	15
C	20	25	30

Agent 1 is assigned URL A as the highest ranked. Agent 2 is assigned URL B as the highest ranked. Agent 3 is assigned URL C as the highest ranked.

Below the table with costs is presented. Costs are on its diagonal:

	Agent 1	Agent 2	Agent 3
Agent 1	((35-10)+(35-20))/10=4	35-10=25	35-20=15
Agent 2	30-20=10	((30-20)+(30-25))/10=1,5	30-25=5
Agent 3	30-25=5	30-15=15	((30-15)+(30-25))/10=2

As one can see in this round Agent 1 is considered as a loser. The new ranks for answers are:

(Agent 1, A) = 35 - 4 = 31
(Agent 2, B) = 30 - 1,5 = 28,5
(Agent 3, C) = 30 - 2 = 28

Those ranks are updated and put into overall ranking. At this point agents can change their favored answer but in this case this is not happening since still the updated ranks are higher than other ones.

Listing 3.2.1 Example of *Auction* method flow process

Example cont:

Then agents enter the next round with their URLs ranked as following:

Answer	Agent 1	Agent 2	Agent 3
A	31 (subtracted cost)	20	25
B	10	28,5 (subtracted cost)	15
C	20	25	28 (subtracted cost)

Agent 1 is assigned URL A as the highest ranked. Agent 2 is assigned URL B as the highest ranked. Agent 3 is assigned URL C as the highest ranked.

Below the table with costs is presented. Costs are on its diagonal:

	Agent 1	Agent 2	Agent 3
Agent 1	((31-10)+(31-20))/10=3,2	35-10=25	35-20=15
Agent 2	30-20=10	((28,5-20)+(28,5-25))/10=1,2	30-25=5
Agent 3	30-25=5	30-15=15	((28-15)+(28-25))/10=1,6

As one can see in this round Agent 1 is considered as a loser. Its cost is 3,2 which is the highest one. Since it happened second time in a row – it and result set assigned to it are removed from the further part of the negotiation. The new ranks for answers are:

(Agent 1, A) = 31 – 3,2 = 27,8
(Agent 2, B) = 28,5 – 1,2 = 27,3
(Agent 3, C) = 28 – 1,6 = 26,4

Then process continues in this way until there is only one agent with his URL assigned. Then, afterwards next “big” round is started.

Listing 3.2.2 Continuation of the example of *Auction* round flow process

In *Auction* method as in the *Game theory* method there is no direct participation of *Search Agents* in the process. *Search Agents* are just for grouping purposes and could not be used at all.

Listing 3.2.3 presents the pseudo code for the main process of the *Auction* method.

Input: Map containing URL rankings.

Output: 10 URLs.

BEGIN

1. repeat until there are 10 URLs in answer list
2. repeat until one agent remains
3. find highest ranked URLs for all agents and pair them like $\left(A^{(i)},U^{(i)}\right)$
4. calculate costs for each agent:

$$\cos t\left(A^{(i)}\right)=\frac{\sum_{i=1,i=j}^{i=m} rank\left(A^{(i)},U^{(i)}\right)-rank\left(A^{(i)},U^{(j)}\right)}{10}$$

where $U^{(i)}$ is URL from pair $\left(A^{(i)},U^{(i)}\right)$ (highest ranked URL for agent $A^{(i)}$ and $U^{(j)}$ is a highest ranked URL for agent $A^{(j)}$

5. find agent with highest cost – he is a loser
   - it may happen that all agents have the same costs – if it occurs twice the agent which is assigned the URL initially ranked as the lowest is considered a loser and thus removed from further negotiation, if it so go to 7.
6. if the agent is a loser twice in a row remove him from further auction
7. update URL rankings for all agents with following values:

$$rank\left(A^{(i)},U^{(i)}\right)=rank\left(A^{(i)},U^{(i)}\right)-\cos t\left(A^{(i)}\right)$$

where $\left(A^{(i)},U^{(i)}\right)$ is the pair found at the beginning; at this point the winning URL can be changed

8. add URL to answer list
9. remove the URL from further evaluation
10. go to 2

END

Listing 3.2.3 *Auction* method main algorithm

Following activity diagram presents the *Auction* method workflow. Like in the previous case all objects in the process are shown on this diagram so that it can be easily seen who is responsible for certain parts of the process.

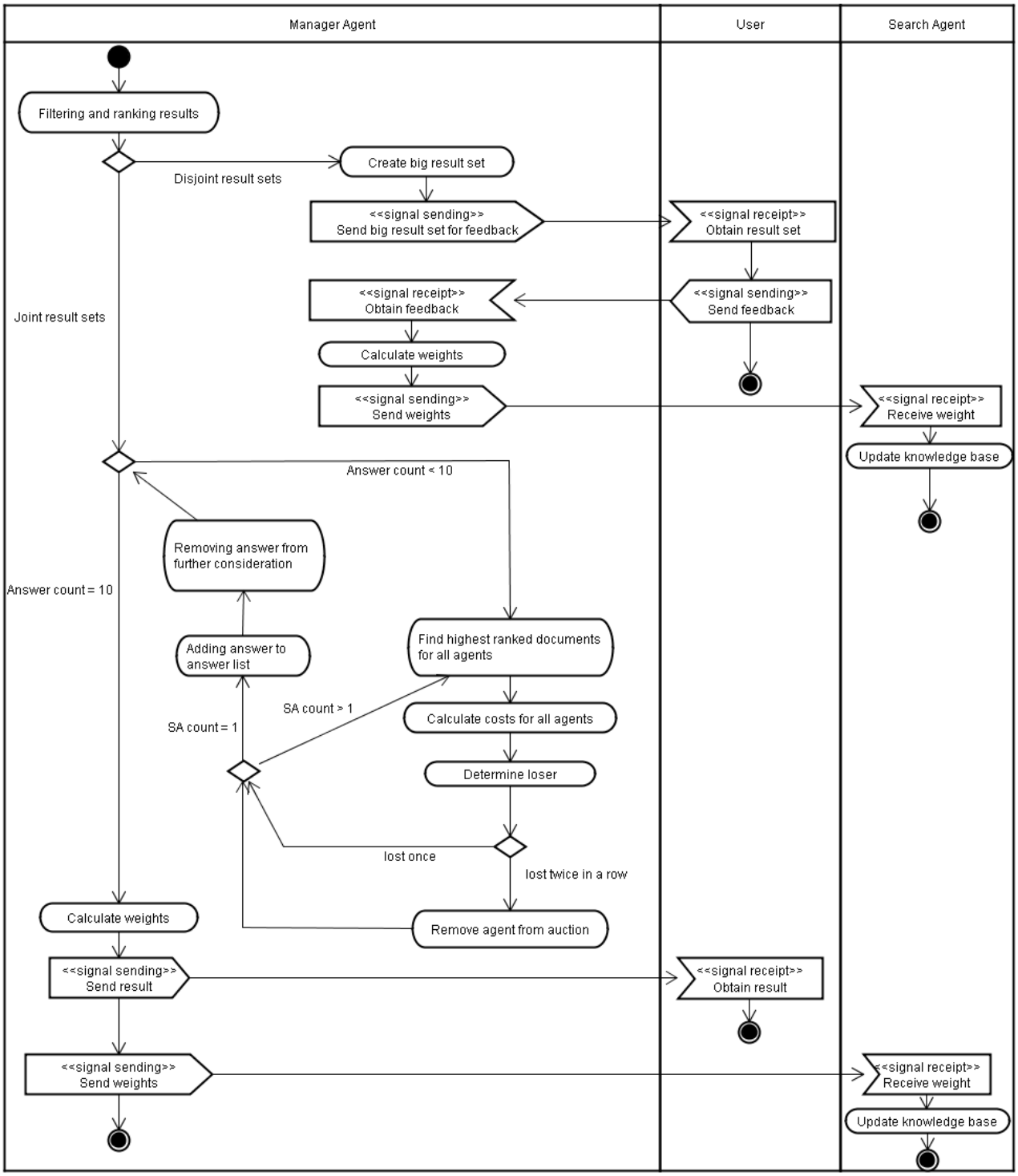

Fig 3.2.1 *Auction* Method Activity Diagram

3.3 *Consensus* method

The *Consensus* method was used previously in the *AGWI* system [1]. The *Consensus* approach for conflict solving has been widely described by Nguyen N.T. in [4]. Its main aim is: given a set of answers reach the common agreement on what the final combined answer should be. It has been applied to be used in the application under considerations, however with slight differences. The main assumptions of this approach were not altered – consensus answer is created at the beginning and then its consistency is evaluated. The consistency part has been slightly modified. One of steps performed in this algorithm is measuring of distances between result sets. Modification changes the way; the evaluation of distances takes place. Another difference lies in the method of choosing of the search engines for results retrieval. In the *AGWI* system there were more search engines than there were *Search Agents*. In this case there are as many *Search Agents* as there are search engines which are to be utilized.

First, the result sets are evaluated. A combined result set (without repetition of URLs) from all result sets is created. Then for each URL its average position in result sets is calculated. After that the combined result sets is sorted according to the average positions. The consensus answer is found. Afterwards, it remains to check its consistency.

Listing 3.3.1 presents the pseudo code of algorithm for finding the consensus answer.

Input: Map of results $\langle a^i, r^i \rangle$ provided by m Search Agents – each in the form $r^i = \langle U^i{}_1, U^i{}_2, ..., U^i{}_n \rangle$ where $U^i{}_1, U^i{}_2, ..., U^i{}_n$ are URLs. Map containing weights for result sets.
***Output*: Consensus answer**

BEGIN

1. create set $URLS$ from all URLs from all result sets (without repetitions)
2. for each $U \in URLS$
   - create array $\langle t_1, t_2, ..., t_n \rangle$ where t_i is position on which U appears in $r^{(i)}$;
   - if U does not appear in $r^{(i)}$ then set t_i as the length of the longest ranking increased by 1
   - divide each t_i by $weight(r^{(i)})$; if $weight(r^{(i)}) = 0$ divide by 0.01
   - calculate average $t(U)$ of values $t_1, t_2, ..., t_n$
3. consensus answer is obtained by ordering elements of $URLS$ according to values $t(U)$

END

Listing 3.3.1 *Consensus* method main algorithm

Having found the consensus answer; algorithm must check its consistency. To check consistency of consensus answer the average of distances between result sets and average of distances between each of result set and the consensus answer must be evaluated. Before performing the calculation, however, the result sets and consensus are normalized; only a specific number of top URLs are incorporated into the answer. This number is of size of the smallest non-zero result set. Afterwards application calculates averages, and checks if the average of distances is bigger than average of distances of result sets to the consensus. If it is so, then consensus answer is consistent; if not the consensus answer is not consistent.

Listing 3.3.2 presents the algorithm for evaluating the consensus consistency:

Input: Map of results $X = \langle a^i, r^i \rangle$ provided by m Search Agents – each in the form $r^i = \langle U^i{}_1, U^i{}_2, ..., U^i{}_n \rangle$ where $U^i{}_1, U^i{}_2, ..., U^i{}_n$ are URLs; consensus answer found earlier.

Output: TRUE or FALSE

BEGIN

1. trim result sets and consensus to the smallest non zero result set
2. calculate:

$$\hat{d}(X) = \frac{\sum_{x,y \in X} d(x,y)}{m(m+1)}$$

where $d(x,y)$ is the distance between two result sets (Levenshtein distance – section 3.4.3)

3. calculate:

$$\hat{d}_{\min}(X) = \frac{\sum_{x \in X} d(x,C)}{m}$$

where $d(x,C)$ is Levenshtein distance between consensus and result set

4. if $\hat{d}(X) \geq \hat{d}_{\min}(X)$ then return TRUE. Else FALSE

END

Listing 3.3.2 Algorithm evaluating consensus consistency

Having checked the consistency algorithm, now decides on the next step. In case the consistency of the answer is low, the answer is returned containing all results and feedback to the application should be provided. If the consistency is high, 10 first URLs from consensus answer are presented.

Depending on the outcome of the consistency check the different entry point is used for the weight calculation algorithm. If the consistency of the consensus was high, the agent whose result set has the smallest distance to the consensus is selected as the agent whose weight will be equal to 1 and the algorithm in listing 3.3.3 does not require the feedback URL as an input – step 1 is omitted. If the consistency was low, the first step of the algorithm must be performed to find the agent.

Input: Map of results $\langle a^i, r^i \rangle$ provided by m Search agents – each in the form $r^i = \langle U^i{}_1, U^i{}_2, ..., U^i{}_n \rangle$ where $U^i{}_1, U^i{}_2, ..., U^i{}_n$ are URLs; feedback URL

Output: Set of weights with corresponding agents

BEGIN

1. find the agent whose result set contains URL from feedback and is closest to consensus, set his weight to 1
2. for all other agents:

   $find\ d(r^{(i)}, C)$

   $$W[i] = \frac{|r^{(i)}| - d(r^{(i)}, C)}{|r^{(i)}|}$$

   where $d(r^{(i)}, C)$ is the Levenshtein distance
3. return weights

END

Listing 3.3.3 Weights calculation algorithm for *Consensus* method

Those weights are used as ranking modifiers of the results provided by the search engines, when application is issued the same query for this algorithm. When the weights are calculated *Manager Agent* sends those to the corresponding *Search Agents*. Depending on the distances between results sets provided by *Search Agents* weight may vary from 0 to 1. Weight will be equal to 0 when a result set has maximal distance to the anchor result set. Afterwards, when weights are already calculated those are stored in the database in case the query is issued once more. Then during main algorithm, which yields the consensus answer, those are used as URL ranks modifiers – the positions of URLs are divided by those. This results in moving a certain URL to the bottom of the list if the weight of the result set from which the URL originates is close to zero. If the weight is equal to 0, URL position is divided by 0.01.[2]

[2] Like stated before, this way of ranking search engines was not tested and was disabled during tests which are described in chapter 4. Weights of all search engines were equal to 1 – URL position was not altered. However, it was implemented for future possibility of including rankings of the search engines in the process of answers processing.

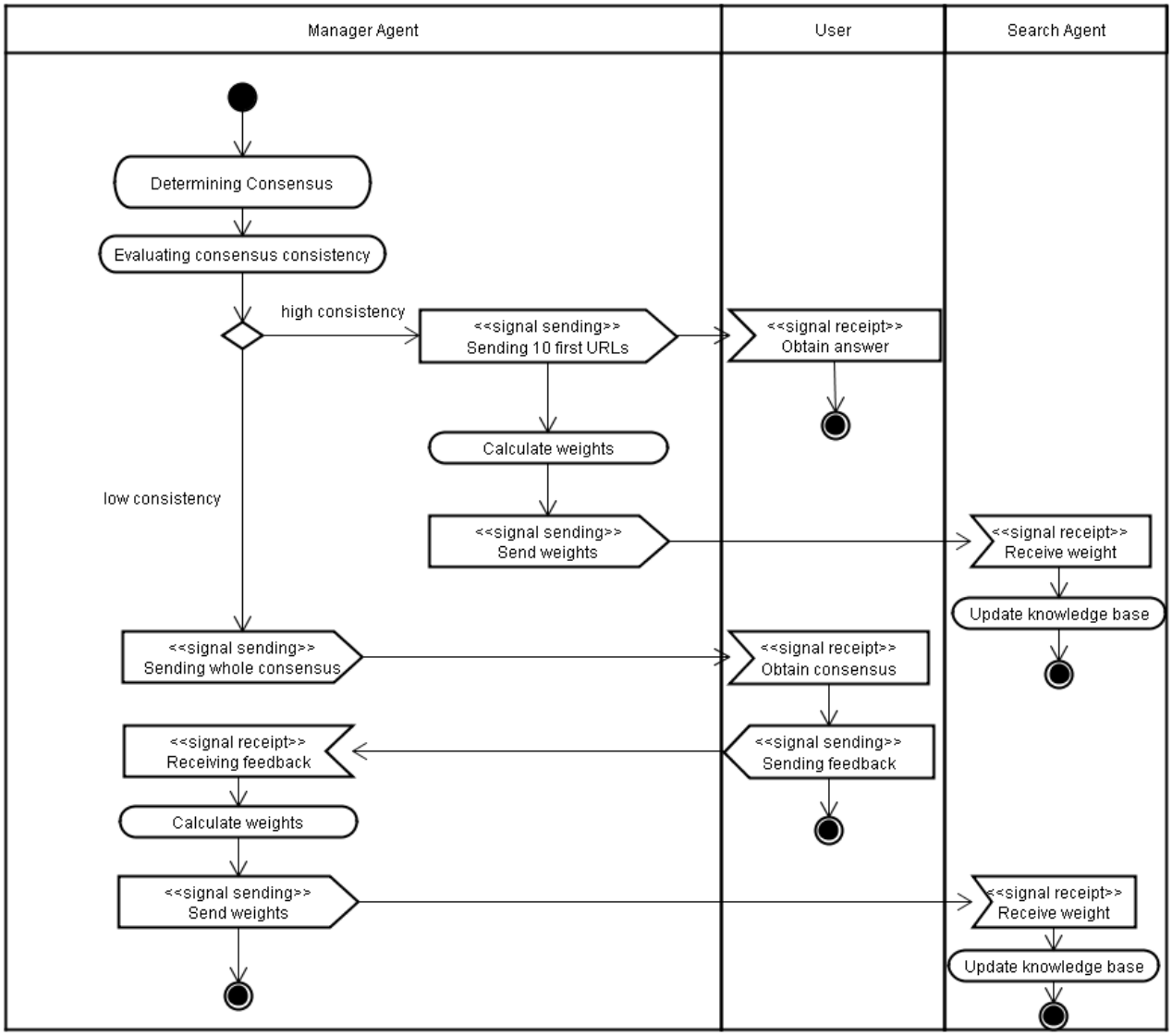

Fig 3.3.1 *Consensus* Method Activity Diagram

3.4 Common algorithms

This part of chapter 3 presents algorithms that are commonly used throughout the application. This chapter presents the purposes of the algorithms and their pseudo codes. Also a short description of each algorithm is provided.

3.4.1 Ranking algorithm

Listing 3.4.1 presents the pseudo code of the algorithm for the initial URL ranking. This initial ranking is being performed before the *Game theory* and *Auction* methods (not the *Consensus* method) can start their main computational parts. Its purpose is to calculate the confidence values of the *Search Agents* about a certain URL. The confidence value in general is calculated as follows: $|result\ set\ of\ agent| - position\ \ of\ the\ URL\ in\ the\ result\ set$. However the *Game theory* and the *Auction* methods require that each of the result sets contain the same URLs, not necessarily at the same places. In other case the algorithm breaks, since agent may now nothing about a certain URL and therefore the comparison of ranks of this certain URL cannot be performed. This algorithm also insures that this assumption is fulfilled by updating the result sets with missing URLs. Algorithm also determines if the main computational parts of the two aforementioned approaches can be even performed. The rule is as following: if for all pairs of result sets say $A\ and\ B$ the $A \cap B = \varnothing$ then the main part of the *Game theory* and *Auction* can not start. If there is a result set say that has no common URL with any other result sets it is removed from the process at the very beginning as being not suitable for the algorithms which require every URL to be in every result set.

Input: **Map of results** $\langle a^i, r^i \rangle$ **provided by m Search Agents – each in the form** $r^i = \langle U^i{}_1, U^i{}_2, ..., U^i{}_n \rangle$ **where** $U^i{}_1, U^i{}_2, ..., U^i{}_n$ **are URLs. Map containing weights of the result lists.**

Output: **Map containing URL rankings**

BEGIN

1. for each agent in map:
   - check if other agents result sets contain any of the URLs of the agent
   - construct matrix representing how many URLs of the agent are contained in the each result set of other agents
2. check if each agent has at least one common URL with another if not – remove him from the further process
3. if result set of every agent is disjoint with each result set of every other agent – stop algorithm
4. for each agent in map:
   - for each URL in agent result set
     - rank the URL as following: $rank(U^i) = (|r| - i) * weight(r)$ where *i* is a position of URL in *r*
     - find agents which result set does not contain the URL, update their rankings: $rank(U^i) = 1.0 * weight(r)$ (weights calculation – listing 3.4.2)
5. return ranking

END

Listing 3.4.1 URL ranking algorithm for *Game theory* and *Auction* methods

3.4.2 Weights calculation for *Game theory* and *Auction* methods

Following listing presents the pseudo code for the weights calculation algorithm. Weights calculation is performed after *Game theory* and *Auction* methods finish their main negotiation parts. This algorithm is to rank the search engines according to how the URL from a given engine was evaluated in the final answer of the algorithm. The topmost URL is chosen to be the feedback result and other result sets are weighted accordingly to the number of URLs overlapping with the result set which provided the URL. After this part is finished ranks are stored in the knowledge base for further use. The weights are used as follows: when issuing the query for the second time for a particular method (*Game theory* or Auction in this case) the weight of the result set is used to diminish the rank of the URL which originates from this result set. The rank of such URL is multiplied by this weight thus, if it is less than 1 it is being diminished. This process gives handicap to URLs which are returned by the search engines with low weights – those contributed in small extent to the previous algorithm results for a particular query. If the weight is equal to zero the rank

is multiplied by 0.01.[3]

If the algorithms could not be started; the application creates a combined result set from all result sets without URL repetitions and such large set is displayed with a possibility to provide feedback. The URL which is considered to be the best can be marked as feedback and it is sent to the application, which uses it as an anchor to start weights calculation process.

***Input*: Result from feedback; initial result sets**

***Output*: Map of weights with corresponding agents**

BEGIN

1. find the agent whose result set contains the result from feedback, set his weight to 1
2. for all other agents:

   $find\ d\left(r^{(i)}, r^{w}\right)$

   $$W[i] = \frac{\left|r^{(i)}\right| - d\left(r^{(i)}, r^{w}\right)}{\left|r^{(i)}\right|}$$

   where $d\left(r^{(i)}, r^{w}\right)$ is the number of different URLs between the result set of agent *i* and the "winner " agent (note that those in case of ad joint result sets will be equal to zero)
3. return weights

END

Listing 3.4.2 Weights calculation for *Game theory* and *Auction* methods

3.4.3 Adapted Levenshtein distance

Next listing presents the adapted algorithm for finding Levenshtein distance. An adaptation of this algorithm was used in the application for calculating distances between result sets. The algorithm is simple but at the same moment it is very fast and provides well and easily interpretable results.

In its original version it is an edit distance – measure of distance between strings. It finds how many basic operations are needed to transform one string into another. "Basic operations" mean the following:

- deletion of a character from the string
- insertion of a character to a string
- substitution of a character with another character

This distance was applied to measure the distance between result sets. Adaptation of this distance was as following: strings became result sets; characters became URLs. Having this translation, one

[3] As for the previous case – this functionality was disabled for the tests presented in chapter 4. Weight of every search engine was equal to 1 – URL ranks were not altered.

could interpret it as number of basic operations (in the sense defined above) to unify two different result sets.

Following example illustrates how the distance between two result sets can be evaluated.

Example:

Let us consider following result sets:
RS1 = (a, b, c)
RS2 = (b, c, a)

Then the distance between those result sets is equal to 2.

To obtain RS2 from RS1 one is required to do:
1 deletion – remove a from the beginning
1 insertion – add a at the end

It gives following 2 alignments:

1. **(a, b, c)**
   (b, c, a)
2. **(a, b, c, –)**
   (–, b, c, a)

What corresponds to lowest cost path from (–1, –1) to (2, 2)

		–1	0	1	2
			b	c	A
–1		0	1	2	3
0	a	1	1	2	2
1	b	2	1	2	3
2	c	3	2	1	2

Listing 3.4.3 Example of variation of algorithm for Levenshtein distance

This distance is used in *Consensus* Method. It is used during the main algorithm part and also during weights calculation after it. The following listing presents pseudo code of dynamic programming version of the variation of this algorithm.

```
Input: Two lists with URLs
Output: Levenshtein distance between lists
int LevenshteinDistance(List<HTMLTagA> list1, List<HTMLTagA> list2)
      declare int d[list1.size() + 1, list2.size() + 1]

      for i from 0 to m
            d[i, 0] := i
      for j from 0 to n
            d[0, j] := j

      for i from 1 to m
            for j from 1 to n
                  if list1[i-1] = list2[j-1] then
                        cost := 0
                  else cost := 1
                  d[i, j] := minimum(
                        d[i-1, j] + 1,     // deletion
                        d[i, j-1] + 1,     // insertion
                        d[i-1, j-1] + cost   // substitution
                        )
return d[list1.size(),list2.size()]
```

Listing 3.4.4 Pseudo code of variation of algorithm for Levenshtein distance

Next chapter presents conducted tests of the three methods. Each of the methods was compared to search engines and then methods were compared between themselves. Next chapter presents those results and contains comments on those.

4. Tests of the three approaches

This chapter presents the tests of the three approaches: *Game theory, Auction* and *Consensus*. There were three queries issued for the testing purposes: `consensus decision making`, `consensus decision making for conflict solving` and `is consensus decision making for conflict solving good enough or maybe Game theory or auction is better`. The idea was to take three queries which relate to the same topic; however first was to be simple, second more complex and third was to be very complex, while retaining coherence.

There were 5 search engines queried. Four of them were English-language-based: *Google, Ask.com, Live, Yahoo!* and one of Polish origin – *Interia*, which in fact is a *Google* based engine; however very often it produces results which differ from its parent engine. Search engines were set up to return 20 results for each query. This means that as input to tested algorithms there were 5 result sets provided; each comprising of 20 URLs. This allowed for fast algorithm processing.

The first phase of result evaluation was to compare the result sets of each of the three tested approaches against result sets produced by each search engine individually. There are two measures of comparison: ***Set Coverage*** and ***URL to URL*** coverage. ***Set Coverage*** measures how many URLs from the result of the algorithm is contained in the result set returned by the search engine regardless of the position of the URL. ***URL to URL*** measures how many URLs were at the same position in both results – of the algorithm and that of the search engine. Those measures however; were taken only for the 10 top results returned by each search engine. This means that in the algorithms result sets there may be answers which are not shown in the result set of any search engine. Afterwards, algorithms, for each query, were compared with each other and then with *MySpiders* system [11].

4.1 Tests with simple query

Following section will present the results for query: `consensus decision making`. The section is organized as follows: first the results of each algorithm vs. search engines will be presented, afterward the comparison of the methods vs. *MySpiders* will be presented and then at the section conclusion the comparison of the algorithms' results will be provided.

The following table presents results of the *Auction* method and the 10 top URL from result sets of each search engine.

Auction method vs. Search Engines

#	**Auction**	**Google**
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://en.wikipedia.org/wiki/Consensus	http://en.wikipedia.org/wiki/Consensus
3	http://www.zmag.org/forums/consenthread.htm	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://www.casagordita.com/consensus.htm	http://www.npd-solutions.com/consensus.html
5	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.seedsforchange.org.uk/free/consflow.pdf
6	http://www.ic.org/pnp/ocac/	http://www.seedsforchange.org.uk/free/consens
7	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.casagordita.com/consensus.htm
8	http://www.npd-solutions.com/consensus.html	http://www.ic.org/pnp/ocac/
9	http://www.seedsforchange.org.uk/free/consens	http://globenet.org/horizon-local/perso/consent.html
10	http://web.mit.edu/hr/oed/learn/teams/art_decisions.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Ask.com**	**Live**
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.casagordita.com/consensus.htm	http://en.wikipedia.org/wiki/Wikipedia:CON
3	http://www.npd-solutions.com/consensus.html	http://www.npd-solutions.com/consensus.html
4	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
5	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.consensus.net/
6	http://www.zmag.org/forums/consenthread.htm	http://www.casagordita.com/consensus.htm
7	http://en.wikipedia.org/wiki/Consensus_decision-making	http://www.reclaiming.org/resources/consensus/invert.html
8	http://www.spokane-county.wsu.edu/family/consen.htm	http://vagreenparty.org/consensus.html
9	http://www.msu.edu/~corcora5/org/consensus.html	http://www.nato.int/issues/consensus/index.html
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Yahoo**	**Interia**
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus
3	http://www.zmag.org/forums/consenthread.htm	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://en.wikipedia.org/wiki/Consensus	http://www.npd-solutions.com/consensus.html
5	http://www.casagordita.com/consensus.htm	http://www.zmag.org/forums/consenthread.htm

6	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.seedsforchange.org.uk/free/consens
7	http://lefh.net/pcpo/CONSENSUSSteps.pdf	http://www.seedsforchange.org.uk/free/consflow.pdf
8	http://www.npd-solutions.com/consensus.html	http://www.ic.org/pnp/ocac/
9	http://www.reclaiming.org/resources/consensus/invert.html	http://www.casagordita.com/consensus.htm
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://globenet.org/horizon-local/perso/consent.html

Table 4.1.1 Results of *Auction* method and search engines for simple query

Auction	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	60%	40%	70%	60%	70%
URL to URL	0%	10%	20%	30%	20%

Table 4.1.2 Coverage of results of *Auction* method with the search engines for simple query

The table above presents how the auction algorithm performs compared to the result sets of each search engine. It can be observed that the *Auction* method result set covers in 70% result sets of two search engines: *Interia* and *Google*. As stated before, *Interia* utilizes *Google* search engine to provide its results, so the situation with the similar coverage is understandable. However when comparing result sets in position-wise fashion, the result set of *Auction* method is in 30% similar to *Yahoo!* search engine and only in 20% similar to result sets of *Google* and *Interia* which provide best ***Set Coverage*** with this method result set. One more thing to note is that *Ask.com* search engine is set-covered in 60%. However, its ***URL to URL*** coverage is 0%. This means that there were no URLs on the same positions in the result set of *Auction* method and in the result set provided by the *Ask.com* engine. Result set of *Live* search engine is set-covered in 40% and only one URL from this set is at the same position in the result set of the *Auction* method.

Auction method as the two top most URLs returned those which are also two top most in the *Google* and *Interia*. Third URL is also third URL in the *Yahoo!* engine but it is also present in the *Ask.com* engine on the 5th place. However some URLs present in all search engines were not returned. This situation happens due to the nature of the algorithm: during costs evaluation, the result sets which contained those at the top most positions were the ones with the highest cost. It happened due to the lower rank of such URL in other result sets and thus resulting in high cost value. This leads to following: the rank of such URL was seriously diminished in the next round and thus it was not selected as the highest ranked. This shows that A*uction* method may not necessarily return URL which is contained in all result sets. However sometimes this works the other way around: result set of *Auction* contains URLs which are not in the top most results of every search engine. But those were ranked as the top most URLs in some of the search engines and thus resulting in low cost of such URL during processing and leading to small decrease in its rank every next round.

The results described above lead to following conclusion: *Auction* method provides results which are highly dependent on results “featured” by each individual search engine and that are not dependent on the search engines treated “as a whole.” In other words: the presence of the URL on

top enough positions in all result sets of search engines, may not necessarily be a factor which decides if the URL will be taken as a part of the final result.

The following table presents results of the Game theory method and 10 top most URLs from each search engine. Results of the search engines are – obviously – the same as before and are presented here only to simplify digestion of results.

Game theory method vs. Search Engines

#	**Game theory**	**Google**
1	http://en.wikipedia.org/wiki/Consensus	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus
3	http://www.npd-solutions.com/consensus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://www.casagordita.com/consensus.htm	http://www.npd-solutions.com/consensus.html
5	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.seedsforchange.org.uk/free/consflow.pdf
6	http://www.consensus.net/	http://www.seedsforchange.org.uk/free/consens
7	http://en.wikipedia.org/wiki/Consensus_decision-making	http://www.casagordita.com/consensus.htm
8	http://www.zmag.org/forums/consenthread.htm	http://www.ic.org/pnp/ocac/
9	http://www.seedsforchange.org.uk/free/consens	http://globenet.org/horizon-local/perso/consent.html
10	http://www.reclaiming.org/resources/consensus/invert.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Ask.com**	**Live**
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.casagordita.com/consensus.htm	http://en.wikipedia.org/wiki/Wikipedia:CON
3	http://www.npd-solutions.com/consensus.html	http://www.npd-solutions.com/consensus.html
4	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
5	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.consensus.net/
6	http://www.zmag.org/forums/consenthread.htm	http://www.casagordita.com/consensus.htm
7	http://en.wikipedia.org/wiki/Consensus_decision-making	http://www.reclaiming.org/resources/consensus/invert.html
8	http://www.spokane-county.wsu.edu/family/consen.htm	http://vagreenparty.org/consensus.html
9	http://www.msu.edu/~corcora5/org/consensus.html	http://www.nato.int/issues/consensus/index.html
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Yahoo**	**Interia**
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus
3	http://www.zmag.org/forums/consenthread.htm	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://en.wikipedia.org/wiki/Consensus	http://www.npd-solutions.com/consensus.html
5	http://www.casagordita.com/consensus.htm	http://www.zmag.org/forums/consenthread.htm
6	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.seedsforchange.org.uk/free/consens
7	http://lefh.net/pcpo/CONSENSUSSteps.pdf	http://www.seedsforchange.org.uk/free/consflow.pdf
8	http://www.npd-solutions.com/consensus.html	http://www.ic.org/pnp/ocac/

9	http://www.reclaiming.org/resources/consensus/invert.html	http://www.casagordita.com/consensus.htm
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://globenet.org/horizon-local/perso/consent.html

Table 4.1.3 Results of *Game theory* method and search engines for simple query

Game theory	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	60%	60%	70%	80%	60%
URL to URL	30%	10%	0%	10%	0%

Table 4.1.4 Coverage of results of *Game theory* method and search engines for simple query

The table above presents how the *Game theory* based algorithm performs compared to the result sets of each search engine. It can be observed that the *Game theory* method result set covers in 80% result sets of the *Yahoo!* search engine. The next engine in line is *Interia* search engine, with 70% ***set-coverage*** (but note the 0% of ***URL to URL*** coverage!). The highest ***URL to URL*** coverage is for the *Ask.com* engine – 30%. Its 60% set coverage is third overall. For the *Game theory* method every engine set-coverage is at least 60% and it is overall higher than in case of the *Auction* method. However, position-wise comparison presents very low values. Even the *Yahoo!* engine with its 80% set-coverage has only one URL at the same place in its result set as has the result set returned by the *Game theory* method. For the *Ask*.com ***URL to URL*** coverage is 30% and is the highest one. *Google* and *Interia* are at the bottom, with ***URL to URL*** coverage equal to 0%.

Game theory method as the top most URL returned the URL which was contained only in 3 out of 5 search engines. This happened because during the game, other higher ranked URLs were discarded as keep payoffs for those were smaller than the change ones. This resulted in elimination of those URLs from the consideration leaving the one which was selected. However the 2nd URL which is returned is an URL which had the worst position in the result sets of the search engines equal to 4. This means that the keep payoff for this URL was high and it was not eliminated during the game. Some overall higher ranked links appear on the lower places. It means that in the end those were taken as a part of the result. This happened because of the removal of the already selected URLs from further consideration, thus allowing the keep payoffs to be high enough for those URLs to appear.

Game theory method returns the overall top most URLs from all search engines. Even though, those URLs are not necessarily at the top places in the final result. In this case it means that if an URL is overall ranked high enough, it will be taken into consideration even in the latter part of the process of preparing the final answer.

The following section presents results of the *Consensus* method and 10 top most URLs from each search engine (again the individual results are kept of simplicity of the comparison).

Consensus method vs. Search engines

#	**Consensus (not consistent)**	**Google**
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus
3	http://www.npd-solutions.com/consensus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://www.casagordita.com/consensus.htm	http://www.npd-solutions.com/consensus.html
5	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.seedsforchange.org.uk/free/consflow.pdf
6	http://en.wikipedia.org/wiki/Consensus	http://www.seedsforchange.org.uk/free/consens
7	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.casagordita.com/consensus.htm
8	http://www.zmag.org/forums/consenthread.htm	http://www.ic.org/pnp/ocac/
9	http://www.seedsforchange.org.uk/free/consens	http://globenet.org/horizon-local/perso/consent.html
10	http://www.seedsforchange.org.uk/free/consflow.pdf	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Ask.com**	**Live**
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.casagordita.com/consensus.htm	http://en.wikipedia.org/wiki/Wikipedia:CON
3	http://www.npd-solutions.com/consensus.html	http://www.npd-solutions.com/consensus.html
4	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
5	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.consensus.net/
6	http://www.zmag.org/forums/consenthread.htm	http://www.casagordita.com/consensus.htm
7	http://en.wikipedia.org/wiki/Consensus_decision-making	http://www.reclaiming.org/resources/consensus/invert.html
8	http://www.spokane-county.wsu.edu/family/consen.htm	http://vagreenparty.org/consensus.html
9	http://www.msu.edu/~corcora5/org/consensus.html	http://www.nato.int/issues/consensus/index.html
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
#	**Yahoo**	**Interia**
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://en.wikipedia.org/wiki/Consensus
3	http://www.zmag.org/forums/consenthread.htm	http://www.actupny.org/documents/CDdocuments/Consensus.html
4	http://en.wikipedia.org/wiki/Consensus	http://www.npd-solutions.com/consensus.html
5	http://www.casagordita.com/consensus.htm	http://www.zmag.org/forums/consenthread.htm
6	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.seedsforchange.org.uk/free/consens
7	http://lefh.net/pcpo/CONSENSUSSteps.pdf	http://www.seedsforchange.org.uk/free/consflow.pdf
8	http://www.npd-solutions.com/consensus.html	http://www.ic.org/pnp/ocac/
9	http://www.reclaiming.org/resources/consensus/invert.html	http://www.casagordita.com/consensus.htm
10	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://globenet.org/horizon-local/perso/consent.html

Table 4.1.5 Results of *Consensus* method and search engines for simple query

Consensus	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	70%	50%	80%	70%	80%
URL to URL	20%	20%	10%	20%	10%

Table 4.1.6 Coverage of results of *Consensus* method and search engines for simple query

Table 4.1.5 presents the comparison of result sets between C*onsensus* method and each of the result sets returned by the search engines. Table 4.1.6 presents the sets coverage. Sets ***set-coverage*** is at least 50% but ***URL to URL*** coverage is very low and is at most 20%. Also the result set which was provided by *Consensus* method was not consistent. It means that average of distances (Levenshtein) between consensus answer and the result set each of the search engines was higher than average of distance between result sets between search engines. This shows that the result sets are dispersed in the sense of the Levenshtein distance.

Consensus method's answer processing algorithm is based on average ranks of the URLs from the result sets of the engines - in the final answer there will be overall highest ranked URLs. If an URL was at the top places throughout the result sets of the engines, it will be at one of the topmost places in the consensus answer. If its overall ranking was low it will be on low place or not at all in the final answer.

When measuring distances between result set using Levenshtein distance, this method is highly dependent on URL positions – where a particular URL is placed in first set and where it is placed in the second set, highly contributes to the final distance value. It can be observed that URL coverage of the consensus answer is very low for each result set. This resulted in a consensus answer which is said to be inconsistent. Nevertheless, it is a subjective result, the inconsistency. If to use some other metric of measuring distance, it could happen that the result would be marked as being consistent.

This part will present comparison of the results yielded by three methods. First methods will be compared with each other and then those will be compared to Menczer's *MySpiders* [11] information retrieval system. *MySpiders* system is a multi-agent system, where answers are processed using feed-forward neural network.

Methods vs. *MySpiders*

#	Auction	Game theory	Consensus (not consistent)	MySpiders
1	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus	http://en.wikipedia.org/wiki/Consensus_decision-making	http://en.wikipedia.org/wiki/Consensus_decision-making
2	http://en.wikipedia.org/wiki/Consensus	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.welcomehome.org/rainbow/focalizers/consenseus.html
3	http://www.zmag.org/forums/consenthread.htm	http://www.npd-solutions.com/consensus.html	http://www.npd-solutions.com/consensus.html	http://www.seedsforchange.org.uk/free/consens
4	http://www.casagordita.com/consensus.htm	http://www.casagordita.com/consensus.htm	http://www.casagordita.com/consensus.htm	http://www.actupny.org/documents/CDdocuments/Consensus.html
5	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.ballfoundation.org/ei/tools/consensus.html	http://www.casagordita.com/consensus.htm
6	http://www.ic.org/pnp/ocac/	http://www.consensus.net/	http://en.wikipedia.org/wiki/Consensus	http://en.wikipedia.org/wiki/Consensus
7	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://en.wikipedia.org/wiki/Consensus_decision-making	http://www.welcomehome.org/rainbow/focalizers/consenseus.html	http://www.npd-solutions.com/consensus.html
8	http://www.npd-solutions.com/consensus.html	http://www.zmag.org/forums/consenthread.htm	http://www.zmag.org/forums/consenthread.htm	http://www.seedsforchange.org.uk/free/resources
9	http://www.seedsforchange.org.uk/free/consens	http://www.seedsforchange.org.uk/free/consens	http://www.seedsforchange.org.uk/free/consens	http://www.npd-solutions.com/teambldgws.html
10	http://web.mit.edu/hr/oed/learn/teams/art_decisions.html	http://www.reclaiming.org/resources/consensus/invert.html	http://www.seedsforchange.org.uk/free/consflow.pdf	http://www.actupny.org/documents/CDdocuments/Jailsolid.html

Table 4.1.7 Results of methods and *MySpiders* system for simple query

The next section will compare the results provided by *Auction*, *Game theory* and *Consensus* methods. Results are presented in the Table 4.1.7. Table 4.1.8 presents methods' result sets coverage.

	Auction	Consensus	Game theory
Auction	–	70%	60%
Consensus	30%	–	80%
Game theory	20%	60%	–

Table 4.1.8 Coverage of methods' results

Table above presents the ***set–coverage*** and ***URL to URL*** coverage, between result sets returned by those methods. ***Set coverage*** values are placed in the upper-right corner while ***URL to URL*** values are in the lower-left corner.

It can be observed that result sets set-coverage is at least 60% for each pair. As for URL to URL coverage *Consensus* and *Game theory* are covered in 60% while *Auction* has 3 URLs on the

same place as the *Consensus* and 2 at the same place as the *Game theory* method.

To compare quality of those results the 3 top most URLs from each of the result sets were investigated. *Auction* method provided us with *Wikipedia* definitions of the word *Consensus* and *Consensus decision making* process. As the third URL *Auction* method provided an URL to resource which is an interesting dispute about the real-life application of consensus decision making. It presented some good and some ridiculous aspects of this decision process when dealing with particular real-life situation *Game theory* method as the first URL provided the Wikipedia definition of the word consensus. Second and third URL pointed to resources which were also disputes about real life application of consensus decision making. In those resources one could find essential information about how the consensus decision making process should be performed. The three URLs returned by the *Consensus* method were the most promising however. Only one of those contained raw definition but still more precise than the resource with definition from the result set of *Game theory*. Two latter URLs did not contain the raw definitions of this process but rather examples and requirements for this process to be applied successfully. Two of those were in the three topmost URLs of the *Game theory* method as well. However *Consensus* answer was inconsistent, so those results according to the *Consensus* theory are not a successful consensus-made decision since many of the search engines' result sets, which this answer was comprised of, were highly dispersed. Nevertheless if to compare the result sets regardless if the consensus answer was consistent or not, they results should be classified as follows:

1. *Consensus* method – provided most promising resources, not just raw definition of consensus but rather definition of the process consensus decision making.
2. *Game theory* method – provided two interesting resources in the three top most URLs and one raw definition of consensus (not consensus decision making)
3. *Auction* method – only one resource was something more than just raw definition of the terms contained in the query.

The following part will present comparison of result sets returned by methods, with the result set returned by *MySpiders* system.

MySpiders	Auction	Game theory	Consensus
Set Coverage	60%	60%	70%
URL to URL	10%	0%	20%

Table 4.1.9 Coverage of methods' results and results of *MySpiders* system for simple query

The table 4.1.9 presents coverage of *MySpiders* system vs. the result sets returned by the algorithms. It can be observed that *MySpiders* system as the 10 top URLs returns 6 URLs which are in the *Auction* and *Game theory* method result sets. *Consensus* method is covered by 7 URLs. URL to URL coverage is very low and only result set yield by *Auction* and *Consensus* method have at least one URL (1 and 2 respectively) on the same position as the *MySpiders* system. Nevertheless, ***set-coverage*** greater or equal 50% means that answer sets are very similar.

As the first URL *MySpiders* returns the *Wikipedia* definition of consensus decision making. This URL is contained in all result sets of the algorithms tested. Second URL points to the resource which is a short description on how the consensus decision making process should look like. This URL is present in *Consensus* method result set at the 7th position. Third URL is another resource about the consensus decision making where consensus decision making is widely described. The URL pointing to this resource is also present in result set of every method and in each it is placed at 9th position. Also in this resource some questions concerning consensus decision making are being answered. In general only one URL (10th) was not contained any of the tested methods result sets. However, the webpage which contains the resource this URL pointed to is pointed to by other URLs which are contained in the result sets of the every tested method.

Summarizing, the *Consensus* method provided the closest view to the *MySpiders* system. While *Consensus* method provided the closest view the other methods are not far behind and differ only by one URL from the result set returned by *MySpiders. MySpiders* is also a content based method, but it does not produce necessarily better results. Almost all URLs were found in the result sets of tested methods (7 of those were in *Consensus* method result set), so the content of the resources was important in this case. Content of the resources is already processed by the search engines, so it was enough to select the best URLs out of the top results presented by the search engines. In fact by processing content once more, one can filter the answers too extensively. However, this was not the case here, as the URLs provided by *MySpiders* proven to be of value.

4.2 Tests with more complex query

This part of the chapter presents results for query: `consensus decision making for conflict solving`.

The following section presents results of the *Auction* method when compared to the search engines.

Auction method vs. Search Engines

#	**Auction**	**Google**
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.npd-solutions.com/consensus.html
2	http://www.exedes.com/	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://www.crcvt.org/mission.html	http://www.managingwholes.com/--consensus.htm
4	http://www.managingwholes.com/--consensus.htm	http://www.wiley.com/WileyCDA/WileyTitle/productCd-0893842567.html
5	http://www.ic.org/pnp/ocac/	http://www.exedes.com/
6	http://www.teach-nology.com/teachers/lesson_plans/health/conflict/	http://www.ic.org/pnp/ocac/
7	http://www.education-world.com/a_curr/curr171.shtml	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://www.vernalproject.org/papers/process/ConsensNotes.pdf	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.peacemakers.ca/bibliography/bib50resolution.html	http://docs.indymedia.org/view/Global/ConflictResolution

10	http://www.treegroup.info/topics/	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz
#	**Ask.com**	**Live**
1	http://www.exedes.com/	http://www.exedes.com/main.htm
2	http://www.exedes.com/main.htm	http://www.exedes.com/
3	http://allentech.net/techstore/related_1560521996.html	http://www.hrdq.com/products/40decisionativitiesSB.htm
4	http://www.urbanministry.org/esa/maintaining-unity-decision-making-problem-solving	http://www.hrdq.com/products/25problemsolving.htm
5	http://www.sasked.gov.sk.ca/docs/native30/nt30app.html	http://www.teleometrics.com/programs/decision_making_and_consensus_building.html
6	http://www.essentialschools.org/cs/resources/view/ces_res/90	http://www.nsdc.org/library/publications/tools/tools9-97rich.cfm
7	http://www.ncjrs.gov/txtfiles/160935.txt	http://store.teambuildinginc.com/items/books/25-problem-solving-decision-making-activities-1018e1ab-detail.htm?1=1
8	http://www.policy.rutgers.edu/CNCR/pdmcm.html	http://en.wikipedia.org/wiki/Decision_making
9	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.umsl.edu/divisions/conted/cpp/toolkit/pdf/PlanningVisioning-ConsensusDecisionMaking.pdf
10	http://www.annfammed.org/cgi/content/full/3/4/307	http://www.mindtools.com/pages/article/newTMC_95.htm
#	**Yahoo**	**Interia**
1	http://policy.rutgers.edu/CNCR/pdmcm.html	http://www.npd-solutions.com/consensus.html
2	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://www.exedes.com/main.htm	http://www.managingwholes.com/--consensus.htm
4	http://www.exedes.com/	http://www.crcvt.org/mission.html
5	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://www.exedes.com/
6	http://www.healthteacher.com/teachersupports/skills6.asp	http://www.ic.org/pnp/ocac/
7	http://www.communicationism.org/docs/Consensus_Decision-Making_Booklet_0-02-14.pdf	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://arscna.org/pdf/refs/Consensus.pdf	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.sasked.gov.sk.ca/docs/elemsoc/g3u41ess.html	http://docs.indymedia.org/view/Global/ConflictResolution
10	http://www.madison.k12.ct.us/publications/shareddesic.htm	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz

Table 4.2.1 Results of *Auction* method and search engines for more complex query

Auction	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	10%	10%	50%	10%	40%
URL to URL	0%	10%	0%	0%	0%

Table 4.2.2 Coverage of results of *Auction* method and search engines for more complex query

Table above presents the coverage of each search engine vs. the *Auction* method. It can be observed that for this query coverage is low no matter the search engine. The engine with best coverage, 50%, is *Interia*. Second in line is *Google* with 40% ***set-coverage***, while other engines are covered in 10%. However both *Interia* and *Google* have 0% of ***URL to URL*** coverage and there is only one engine with only one URL with the same position – *Live*.

For this query *Auction* method as the topmost URL returned a one which is only in 2 result sets of the search engines. As a second however, an URL which is contained by every search engine

was yield. However most of the returned URLs was also returned by the *Interia* search engine what shows that this algorithm most of the time returns URLs which were not eliminated at the end of processing, rather than keeping an URL from the beginning of the process. For this query *Auction* method behaves like in the previous case. The majority of returned URLs are not necessary in all result sets of search engines. Many of the URLs which were overall ranked higher are discarded because of the high cost of such. Instead those which were not eliminated because of having their costs kept low are retained and then presented as the final ones.

The conclusion is similar as for previous query: *Auction* method bases on results of each search engine separately and the fact that some particular URL is in many search engine result sets, does not imply that this URL will be found in the final result set.

The following part presents the comparison of result set returned by *Game theory* method and result sets of search engines.

Game theory method vs. Search Engines

#	**Game theory**	**Google**
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.npd-solutions.com/consensus.html
2	http://www.npd-solutions.com/consensus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://allentech.net/techstore/related_1560521996.html	http://www.managingwholes.com/--consensus.htm
4	http://www.exedes.com/main.htm	http://www.wiley.com/WileyCDA/WileyTitle/productCd-0893842567.html
5	http://www.exedes.com/	http://www.exedes.com/
6	http://www.urbanministry.org/esa/maintaining-unity-decision-making-problem-solving	http://www.ic.org/pnp/ocac/
7	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://www.hrdq.com/products/25problemsolving.htm	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.sasked.gov.sk.ca/docs/native30/nt30app.html	http://docs.indymedia.org/view/Global/ConflictResolution
10	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz
#	**Ask.com**	**Live**
1	http://www.exedes.com/	http://www.exedes.com/main.htm
2	http://www.exedes.com/main.htm	http://www.exedes.com/
3	http://allentech.net/techstore/related_1560521996.html	http://www.hrdq.com/products/40decisionactivitiesSB.htm
4	http://www.urbanministry.org/esa/maintaining-unity-decision-making-problem-solving	http://www.hrdq.com/products/25problemsolving.htm
5	http://www.sasked.gov.sk.ca/docs/native30/nt30app.html	http://www.teleometrics.com/programs/decision_making_and_consensus_building.html
6	http://www.essentialschools.org/cs/resources/view/ces_res/90	http://www.nsdc.org/library/publications/tools/tools9-97rich.cfm
7	http://www.ncjrs.gov/txtfiles/160935.txt	http://store.teambuildinginc.com/items/books/25-problem-solving-decision-making-activities-1018e1ab-detail.htm?1=1
8	http://www.policy.rutgers.edu/CNCR/pdmcm.html	http://en.wikipedia.org/wiki/Decision_making
9	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.umsl.edu/divisions/conted/cpp/toolkit/pdf/PlanningVisioning-ConsensusDecisionMaking.pdf
10	http://www.annfammed.org/cgi/content/full/3/4/307	http://www.mindtools.com/pages/article/newTMC_95.htm

#	Yahoo	Interia
1	http://policy.rutgers.edu/CNCR/pdmcm.html	http://www.npd-solutions.com/consensus.html
2	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://www.exedes.com/main.htm	http://www.managingwholes.com/--consensus.htm
4	http://www.exedes.com/	http://www.crcvt.org/mission.html
5	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://www.exedes.com/
6	http://www.healthteacher.com/teachersupports/skills6.asp	http://www.ic.org/pnp/ocac/
7	http://www.communicationism.org/docs/Consensus_Decision-Making_Booklet_0-02-14.pdf	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://arscna.org/pdf/refs/Consensus.pdf	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.sasked.gov.sk.ca/docs/elemsoc/g3u41ess.html	http://docs.indymedia.org/view/Global/ConflictResolution
10	http://www.madison.k12.ct.us/publications/shareddesic.htm	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz

Table 4.2.3 Results of *Game theory* method and search engines for more complex query

Game theory	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	60%	40%	30%	40%	30%
URL to URL	40%	0%	10%	0%	10%

Table 4.2.4 Coverage of *Game theory* method and search engines for more complex query

Table above presents how the result set returned by the *Game theory* algorithm covers each of the result sets of the search engines. It can be observed that in this case ***set-coverage*** varies from the 30% (*Interia*, *Google)* to 60 % (*Ask.com)*. ***URL to URL*** coverage is low like in the previous cases, only *Ask.com* engine has its URLs covered by 40%. Other non-zero values are assigned to *Google* and *Interia* search engines. Also, like for the previously disputed query, average ***set-coverage*** is higher than in case of the Auction method.

Game theory method returns an URL which is found in 2 out of 5 search engines, as the first returned result. Like in the previous tested query, the absence URLs at high places which are on high places in result sets in search engines could be explained by the nature of this algorithm. Those answers were discarded at the beginning due to the keep payoff not high enough in the first rounds. Then after the topmost URLs were already selected, situation has changed – the keep payoffs were high enough - and the overall highly ranked URLs were added to the final answer, however on lower places.

Like for the previous query, in this case result of the *Game theory* algorithm backs the thesis which was stated earlier – if there is an URL which is highly ranked in all input result sets it will be contained in the result set returned by this method; however not necessarily on some high position.

The following part presents comparison of result set returned by *Consensus* method and result sets returned by search engines.

Consensus method vs. Search Engines

#	**Consensus (inconsistent)**	**Google**
1	http://www.exedes.com/main.htm	http://www.npd-solutions.com/consensus.html
2	http://www.exedes.com/	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://www.colorado.edu/conflict/peace/glossary.htm	http://www.managingwholes.com/--consensus.htm
4	http://www.policy.rutgers.edu/CNCR/pdmcm.html	http://www.wiley.com/WileyCDA/WileyTitle/productCd-0893842567.html
5	http://www.npd-solutions.com/consensus.html	http://www.exedes.com/
6	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.ic.org/pnp/ocac/
7	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://www.managingwholes.com/--consensus.htm	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://docs.indymedia.org/view/Global/ConflictResolution
10	http://www.ic.org/pnp/ocac/	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz
#	**Ask.com**	**Live**
1	http://www.exedes.com/	http://www.exedes.com/main.htm
2	http://www.exedes.com/main.htm	http://www.exedes.com/
3	http://allentech.net/techstore/related_1560521996.html	http://www.hrdq.com/products/40decisionactivitiesSB.htm
4	http://www.urbanministry.org/esa/maintaining-unity-decision-making-problem-solving	http://www.hrdq.com/products/25problemsolving.htm
5	http://www.sasked.gov.sk.ca/docs/native30/nt30app.html	http://www.teleometrics.com/programs/decision_making_and_consensus_building.html
6	http://www.essentialschools.org/cs/resources/view/ces_res/90	http://www.nsdc.org/library/publications/tools/tools9-97rich.cfm
7	http://www.ncjrs.gov/txtfiles/160935.txt	http://store.teambuildinginc.com/items/books/25-problem-solving-decision-making-activities-1018e1ab-detail.htm?1=1
8	http://www.policy.rutgers.edu/CNCR/pdmcm.html	http://en.wikipedia.org/wiki/Decision_making
9	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.umsl.edu/divisions/conted/cpp/toolkit/pdf/PlanningVisioning-ConsensusDecisionMaking.pdf
10	http://www.annfammed.org/cgi/content/full/3/4/307	http://www.mindtools.com/pages/article/newTMC_95.htm
#	**Yahoo**	**Interia**
1	http://policy.rutgers.edu/CNCR/pdmcm.html	http://www.npd-solutions.com/consensus.html
2	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.actupny.org/documents/CDdocuments/Consensus.html
3	http://www.exedes.com/main.htm	http://www.managingwholes.com/--consensus.htm
4	http://www.exedes.com/	http://www.crcvt.org/mission.html
5	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://www.exedes.com/
6	http://www.healthteacher.com/teachersupports/skills6.asp	http://www.ic.org/pnp/ocac/
7	http://www.communicationism.org/docs/Consensus_Decision-Making_Booklet_0-02-14.pdf	http://www.colorado.edu/conflict/peace/glossary.htm
8	http://arscna.org/pdf/refs/Consensus.pdf	http://www.marxists.org/glossary/terms/c/o.htm
9	http://www.sasked.gov.sk.ca/docs/elemsoc/g3u41ess.html	http://docs.indymedia.org/view/Global/ConflictResolution

10	http://www.madison.k12.ct.us/publications/shareddesic.htm	http://ieeexplore.ieee.org/iel5/4106395/4106396/04106417.pdf?isnumber=4106396&prod=CNF&arnumber=4106417&arSt=96&ared=101&arAuthor=Muhammad Nawaz

Table 4.2.5 Results of *Consensus* method and search engines for more complex query

Consensus	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	50%	30%	70%	40%	60%
URL to URL	0%	20%	0%	0%	0%

Table 4.2.6 Coverage of *Consensus* method and search engines for more complex query

It can be observed that the answer of *Consensus* method covers in some extent every engine. The most covered engine is *Interia* (70% ***set-coverage***) while the *Live* engine is the least ***set-covered*** one (30%). ***URL to URL*** coverage is very low – that is why the final result was considered as inconsistent. As stated before, Levenshtein distance is highly dependent on URL positions, thus leading to the large distances between each of the engines' result sets and the consensus answer.

Consensus method is highly rank based one. Like in the previous example, the URLs which the final result set is comprised of, are highly ranked URLs in general. So if the URL was on the top places throughout the engines' result sets it will be contained in the final answer of the *Consensus* method. If the ranking was low, it will not be contained as its average rank will be very low.

The problem with consistence of the answer is like with the previous case. Low ***URL to URL*** coverage, results in Levenshtein distance to grow, thus leading the average of distances to grow. The ***URL to URL*** coverage also means that the result sets were highly dispersed when measuring distances using Levenshtein distance. If the ***URL to URL*** coverage was about 60-80% for each search engine, probably the answer would be marked as consistent.

The following part presents comparison of the result sets provided by the each of the algorithms. Afterwards, comparison between result sets returned by methods and result set returned by *MySpiders* system will be described.

#	Auction	Game theory	Consensus (inconsistent)	MySpiders
1	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.actupny.org/documents/CDdocuments/Consensus.html	http://www.exedes.com/main.htm	http://www.exedes.com/main.htm
2	http://www.exedes.com/	http://www.npd-solutions.com/consensus.html	http://www.exedes.com/	http://www.peacemakers.ca/bibliography/bib50resolution.html
3	http://www.crcvt.org/mission.html	http://allentech.net/techstore/related_1560521996.html	http://www.colorado.edu/conflict/peace/glossary.htm	http://www.managingwholes.com/--consensus.htm
4	http://www.managingwholes.com/--consensus.htm	http://www.exedes.com/main.htm	http://www.policy.rutgers.edu/CNCR/pdmcm.html	http://www.managingwholes.com/glossary-p/c.htm
5	http://www.ic.org/pnp/ocac/	http://www.exedes.com/	http://www.npd-solutions.com/consensus.html	http://www.actupny.org/documents/CDdocuments/HistoryNV.html
6	http://www.teach-nology.com/teachers/lesson_plans/health/conflict/	http://www.urbanministry.org/esa/maintaining-unity-decision-making-problem-solving	http://www.actupny.org/documents/CDdocuments/Consensus.html	
7	http://www.education-world.com/a_curr/curr171.shtml	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/pt3ch11.html	
8	http://www.vernalproject.org/papers/process/ConsensNotes.pdf	http://www.hrdq.com/products/25problemsolving.htm	http://www.managingwholes.com/--consensus.htm	
9	http://www.peacemakers.ca/bibliography/bib50resolution.html	http://www.sasked.gov.sk.ca/docs/native30/nt30app.html	http://www.hrdq.com/products/40decisionactivitiesSB.htm	
10	http://www.treegroup.info/topics/	http://www.hrdq.com/products/40decisionactivitiesSB.htm	http://www.ic.org/pnp/ocac/	

Table 4.2.7 Results of methods and *MySpiders* system for more complex query

	Auction	Consensus	Game theory
Auction	–	40%	20%
Consensus	10%	–	60%
Game theory	10%	10%	–

Table 4.2.8 Coverage of methods for more complex query

Table above presents the set–coverage and ***URL to URL*** coverage, between result sets returned by those methods. ***Set coverage*** values are placed in the upper-right corner while ***URL to URL*** values are in the lower-left corner.

It can be observed that the highest ***set-coverage*** of result sets is 60%. This coverage is observed between *Consensus* and *Game theory*. Lowest ***set-coverage*** is observed between *Auction* and *Game theory* (20%). *Auction* and *Consensus* method are covered in 40%. As for ***URL to URL*** coverage all sets are covered with each other in 10%.

Like for the previous query, to compare quality of those results the 3 top most URLs from

each of the result sets were investigated. *Auction* method as the first URL provided the resource treating about the civil disobedience training. This resource was pointed to by *Game theory* in the previous example. It, for instance compares consensus process to voting process. As the second URL *Auction* method provided an URL to resource which is an "*Executive Decision Services"* company webpage. This company employs experts whom are consultants and coordinators which are supposed to solve any business conflicts using consensus decision making. Third URL returned is a webpage of non-profit pacifist organization which seeks to "promote non-violent conflict resolution skills and processes". *Game theory* method returns the same first URL which was returned by the *Auction*. It was returned before, for the previous query. The second URL was also returned before, when the first query was issued. Third URL points to the webpage which is an online-store. The resource itself is a list of books about team work and conflict resolving at work. *Consensus* method as the two top most URLs returned "*Executive Decision Services"* company webpage.

Both of the URLs point to, de facto, the same resource but those are still distinguished by the search engines and thus treated as a different resource. Third page is the glossary of terms related to conflict. In it we can find short definitions of terms like: "Adversary", "Consensus", "Diplomacy" and etc.

Summarizing if one was to choose the best method in this case it would be hard to select. But a subjective rank looks as follows:

1. *Auction* method – provided three URLs of different purpose: company webpage, page dealing with civil disobedience training (some anarchist/pacifist organization) stating about good sides of the consensus and a pacifist organization which promote the peace idea through citizen education.
2. *Game theory* method – as the first URL provided the same page about civil disobedience training as did the *Auction* method, second URL points to resource which shows the steps of obtaining consensus in the real-life, while third points to the webpage of online-store.
3. *Consensus* method – as two top most URLs it provided in fact the same resource. It means that, in general, search engines had those links ranked as highest. Third page is the glossary of terms related to conflict.

The following part presents comparison of the algorithms and *MySpiders* system.

MySpiders	Auction	Game theory	Consensus
Set Coverage	10%	20%	20%
URL to URL	0%	10%	0%

Table 4.2.9 Coverage of methods and *MySpiders* system for more complex query

The table above presents how result sets returned by algorithms cover *MySpiders* system result. *MySpiders* returned only 5 results for this query. This is probably due to increased

complexity of the query. As the first URL *MySpiders* returned "*Executive Decision Services"* company webpage. This webpage was returned by *Consensus* method also as the first URL, *Auction* method returned URL pointing to the same webpage on the second position however, this URL was not exactly the same. *Game theory* method returns this URL at 4th position. Second webpage is a resource which lists selected biography about the "*Conflict Transformation and Peacebuilding"*. It is also returned by the *Auction* method, but on the 9th position. Third and fourth URL point to the same webpage, however provide different resources. First is lists articles about "*Conflict resolution and consensus building"*, latter is the glossary of terms (points to letter C specifically) related to the query. Third URL returned by *MySpiders* is also contained in the *Auction* and *Consensus* at the 4th and 8th place respectively. Fifth URL points to the same webpage to which URL was returned on the 1st places in *Auction* and *Game theory* methods however, this is not exactly the same resource. While the URL returned by *Auction* and *Game theory* pointed exactly to page where consensus decision making was described, the URL returned by *MySpiders* points to another page which contains information about "*History of mass nonviolent action*".

Summarizing, all of the URLs which were returned by the *MySpiders* system were present in at least one of the result sets of tested methods. This could mean that no matter which of those approaches (*Auction*, *Game theory*, *Consensus* and *MySpiders*) for answer processing is taken; some of the links will be present in one of them. In other words each pair of result sets has at least one URL in common.

4.3 Tests with very complex query

This part of the chapter presents results for query: `is consensus decision making for conflict solving good enough or maybe Game theory or auction is better`. This section is organized as previous ones – first comparison of methods and search engines will be presented and then comparison of methods' result sets will be presented.

The following section presents results of the *Auction* method when compared to the search engines.

Auction vs. Search Engines

#	**Auction**	**Google**
1	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholarr	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholar
2	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholarr	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholar
3	http://plato.stanford.edu/entries/Game theory/	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Day" intitle:"EXPRESSING PREFERENCES WITH PRICE-VECTOR AGENTS IN ..." &um=1&oi=scholar
4	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Day" intitle:"EXPRESSING PREFERENCES WITH PRICE-VECTOR AGENTS IN ..." &um=1&oi=scholarr	http://plato.stanford.edu/entries/Game theory/
5	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
6	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
7	http://links.jstor.org/sici?sici=0192-5121(199704)18:2<121:CAOJIN>2.0.CO;2-K	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
8	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
9	http://www.indiana.edu/~workshop/wsl/gamethe.htm	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
10	http://links.jstor.org/sici?sici=0020-8833(199703)41:1<87:PTRCAI>2.0.CO;2-I	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
#	**Ask.com**	**Live**
1	http://www.colorado.edu/conflict/peace/glossary.htm	http://www.primisonline.com/cgi-bin/POL_program.cgi?programCode=HBSNA&context=
2	http://learn.royalroads.ca/tcmacam/navpages/glossary.htm	http://lsolum.blogspot.com/archives/2005_09_01_lsolum_archive.html
3	http://dieoff.org/page163.htm	http://lsolum.blogspot.com/archives/2005_11_01_lsolum_archive.html
4	http://www.peacemakers.ca/publications/ADRdefinitions.html	http://www.cs.iit.edu/~xli/cs595-game/auction.htm
5	http://www.virtualschool.edu/mon/Economics/KOMT.html	http://www.csc.liv.ac.uk/~mjw/pubs/imas/distrib/powerpoint-slides/lecture07.ppt
6	http://v1.magicbeandip.com/store/browse_books_2679_p28	http://www.msu.edu/course/aec/810/studynotes.htm

7	http://www.calresco.org/lucas/pmo.htm	http://www.cs.ucf.edu/~lboloni/Teaching/EEL6938_2005/slides/MultiAgent.ppt
8	http://home.ubalt.edu/ntsbarsh/Business-stat/stat-data/DsAppendix.htm	http://www.lifewithalacrity.com/social_software/index.html
9	http://www.nanyangmba.ntu.edu.sg/subjects.asp	http://www.lifewithalacrity.com/webtech/index.html
10	http://www.mises.org/story/2451	http://www.marginalrevolution.com/marginalrevolution/2004/05/
#	**Yahoo**	**Interia**
1	http://www.msu.edu/course/aec/810/studynotes.htm	http://plato.stanford.edu/entries/Game theory/
2	http://www.cit.gu.edu.au/~s2130677/teaching/Agents/Workshops/lecture07.pdf	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
3	http://home.earthlink.net/~peter.a.taylor/manifes2.htm	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
4	http://aufrecht.org/blog/swcat/39172	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
5	http://www.concurringopinions.com/archives/economic_analysis_of_law/index.html	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
6	http://dotearth.blogs.nytimes.com/2008/01/13/a-starting-point-for-productive-climate-discourse/index.html?ex=1357966800&en=2de12bb5c6f809de&ei=5088&partner=rssnyt&emc=rss	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
7	http://aws.typepad.com/aws/2005/01/	http://www.kestencgreen.com/kgthesis.pdf
8	http://www.ferc.gov/legal/maj-ord-reg/land-docs/oligoply.pdf	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
9	http://www.drownout.com/blog/archives/cat_reading_list.html	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf
10	http://osnews.com/comments/10354	http://www.indiana.edu/~workshop/wsl/gamethe.htm

Table 4.3.1 Results of *Auction* method and search engines for very complex query

Auction	Ask.com	Live	Interia	Yahoo!	Google
Set Coverage	0%	0%	30%	0%	40%
URL to URL	0%	0%	10%	0%	20%

Table 4.3.2 Coverage of *Auction* method and search engines for very complex query

Table above presents how result sets returned by each search engine are covered by the result of the *Auction* method. It can be observed that *Auction* method covers partially the result sets returned by *Google* and *Interia* search engine (40% and 30% respectively) and no other result set. ***URL to URL*** coverage is non-zero only for result sets returned by two aforementioned search engines. This means that large part of *Auction* method result set is comprised of URLs that were not in the top 10 URLs returned by many search engines. This happened due to the high dispersion of the result sets – many highly ranked URLs were eliminated during processing because of high cost of an engine which presented such URL. High cost of those URLs can be explained by the variety of results returned by search engines. There are not many URLs that are present in every set, thus resulting in lowering their chance of appearing in the final result. Final result was comprised of those URL which were not eliminated – and it appears that *Google* search engine had low cost during many rounds of the process. The *Interia* engine was a second engine in terms of URLs used in the final result. It has some different URLs than *Google* so it can be stated that the final result is mostly comprised of the results of those two search engines. Similar situation happened before, when dealing with the previous queries. *Auction* method provides its answers on result sets of separate single engines rather than taking into account URLs which are present in many engines.

This is due to the algorithm nature which eliminates many result sets during URL extraction process, because of high cost of the search engine which presents such URL. High cost of such engine is due to the non frequent occurrences of the URL which was selected for the *Auction* process.

Summarizing, for this query, *Auction* method is behaving much like for the previous ones. Final result set does not reflect the majority of the result sets of the search engines, but rather it contains those URLs which were not eliminated.

Following part will present results of the *Game theory* method compared vs. result sets of search engines.

Game theory method vs. Search Engines

#	**Game theory**	**Google**
1	http://www.msu.edu/course/aec/810/studynotes.htm	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholar
2	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholarr	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholar
3	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Day" intitle:"EXPRESSING PREFERENCES WITH PRICE-VECTOR AGENTS IN ..." &um=1&oi=scholar
4	http://www.primisonline.com/cgi-bin/POL_program.cgi?programCode=HBSNA&context=	http://plato.stanford.edu/entries/Game theory/
5	http://lsolum.blogspot.com/archives/2005_11_01_lsolum_archive.html	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
6	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholarr	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
7	http://plato.stanford.edu/entries/Game theory/	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
8	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Johansson" intitle:"On Coordination in Multi-agent Systems" &um=1&oi=scholar	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
9	http://lsolum.blogspot.com/archives/2005_09_01_lsolum_archive.html	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
10	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
#	**Ask.com**	**Live**
1	http://www.colorado.edu/conflict/peace/glossary.htm	http://www.primisonline.com/cgi-bin/POL_program.cgi?programCode=HBSNA&context=
2	http://learn.royalroads.ca/tcmacam/navpages/glossary.htm	http://lsolum.blogspot.com/archives/2005_09_01_lsolum_archive.html
3	http://dieoff.org/page163.htm	http://lsolum.blogspot.com/archives/2005_11_01_lsolum_archive.html
4	http://www.peacemakers.ca/publications/ADRdefinitions.html	http://www.cs.iit.edu/~xli/cs595-game/auction.htm

5	http://www.virtualschool.edu/mon/Economics/KOMT.html	http://www.csc.liv.ac.uk/~mjw/pubs/imas/distrib/powerpoint-slides/lecture07.ppt
6	http://v1.magicbeandip.com/store/browse_books_2679_p28	http://www.msu.edu/course/aec/810/studynotes.htm
7	http://www.calresco.org/lucas/pmo.htm	http://www.cs.ucf.edu/~lboloni/Teaching/EEL6938_2005/slides/MultiAgent.ppt
8	http://home.ubalt.edu/ntsbarsh/Business-stat/stat-data/DsAppendix.htm	http://www.lifewithalacrity.com/social_software/index.html
9	http://www.nanyangmba.ntu.edu.sg/subjects.asp	http://www.lifewithalacrity.com/webtech/index.html
10	http://www.mises.org/story/2451	http://www.marginalrevolution.com/marginalrevolution/2004/05/
#	**Yahoo**	**Interia**
1	http://www.msu.edu/course/aec/810/studynotes.htm	http://plato.stanford.edu/entries/Game theory/
2	http://www.cit.gu.edu.au/~s2130677/teaching/Agents/Workshops/lecture07.pdf	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
3	http://home.earthlink.net/~peter.a.taylor/manifes2.htm	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
4	http://aufrecht.org/blog/swcat/39172	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
5	http://www.concurringopinions.com/archives/economic_analysis_of_law/index.html	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
6	http://dotearth.blogs.nytimes.com/2008/01/13/a-starting-point-for-productive-climate-discourse/index.html?ex=1357966800&en=2de12bb5c6f809de&ei=5088&partner=rssnyt&emc=rss	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
7	http://aws.typepad.com/aws/2005/01/	http://www.kestencgreen.com/kgthesis.pdf
8	http://www.ferc.gov/legal/maj-ord-reg/land-docs/oligoply.pdf	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
9	http://www.drownout.com/blog/archives/cat_reading_list.html	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf
10	http://osnews.com/comments/10354	http://www.indiana.edu/~workshop/wsl/gamethe.htm

Table 4.3.3 Results of *Game theory* method and search engines for very complex query

Game theory	**Ask.com**	**Live**	**Interia**	**Yahoo!**	**Google**
Set Coverage	0%	40%	30%	10%	50%
URL to URL	0%	0%	0%	10%	0%

Table 4.3.4 Coverage of *Game theory* method and search engines for very complex query

From the table above, it can be observed that *Game theory* method presents higher set-coverage than the *Auction* method. Only result set of one search engine is not covered at all, other result sets 10 top URLs contributed to the final result. *Google* search engine's result set is the most covered of all result sets but with none ***URL to URL*** coverage. *Ask.com* engine's result set did not contribute to the final result of this method.

As in the previous cases *Game theory* returns URLs which are highly ranked by more than one search engine. That is if there is an URL, which is a part of result sets of more than one engine and it is contained in the 10 top most URLs, it will be included, with high probability, in the final result of the *Game theory* method. Final result also comprises of some URLs that are present in only one result set. That means that some, more common URLs, where eliminated during the URL yielding process, because of their low keep payoff. Due to the low keep payoff, the ranks of the URLs were diminished resulting in those not being taken into account in further process and in turn in leaving a lot of URLs which were not in the majority of the result sets of the search engines.

Game theory method still does not represent the view of the majority of the search engines.

However its result set reflects the result sets of the search engines in greater extent than the *Auction* method. For this query, much like for the previous ones, the conclusion is following, if an URL is highly ranked throughout the result sets of the search engines, it will be included in the final result, however not necessarily on some high place. Other URLs which comprise the final result set are also highly ranked URLs but rather those are contained by result set of one particular engine, rather than by the majority of the result sets.

The following part presents the comparison of the result of *Consensus* method vs. the result sets of the search engines.

Consensus method vs. Search Engines

#	**Consensus (inconsistent)**	**Google**
1	http://plato.stanford.edu/entries/Game theory/	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholar
2	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholar
3	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Day" intitle:"EXPRESSING PREFERENCES WITH PRICE-VECTOR AGENTS IN ..." &um=1&oi=scholar
4	http://www.msu.edu/course/aec/810/studynotes.htm	http://plato.stanford.edu/entries/Game theory/
5	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
6	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
7	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
8	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
9	http://www.indiana.edu/~workshop/wsl/gamethe.htm	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
10	http://www.kestencgreen.com/kgthesis.pdf	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
#	**Ask.com**	**Live**
1	http://www.colorado.edu/conflict/peace/glossary.htm	http://www.primisonline.com/cgi-bin/POL_program.cgi?programCode=HBSNA&context=
2	http://learn.royalroads.ca/tcmacam/navpages/glossary.htm	http://lsolum.blogspot.com/archives/2005_09_01_lsolum_archive.html
3	http://dieoff.org/page163.htm	http://lsolum.blogspot.com/archives/2005_11_01_lsolum_archive.html
4	http://www.peacemakers.ca/publications/ADRdefinitions.html	http://www.cs.iit.edu/~xli/cs595-game/auction.htm
5	http://www.virtualschool.edu/mon/Economics/KOMT.html	http://www.csc.liv.ac.uk/~mjw/pubs/imas/distrib/powerpoint-slides/lecture07.ppt
6	http://v1.magicbeandip.com/store/browse_books_2679_p28	http://www.msu.edu/course/aec/810/studynotes.htm
7	http://www.calresco.org/lucas/pmo.htm	http://www.cs.ucf.edu/~lboloni/Teaching/EEL6938_2005/slides/MultiAgent.ppt
8	http://home.ubalt.edu/ntsbarsh/Business-stat/stat-data/DsAppendix.htm	http://www.lifewithalacrity.com/social_software/index.html
9	http://www.nanyangmba.ntu.edu.sg/subjects.asp	http://www.lifewithalacrity.com/webtech/index.html

10	http://www.mises.org/story/2451	http://www.marginalrevolution.com/marginalrevolution/2004/05/
#	**Yahoo**	**Interia**
1	http://www.msu.edu/course/aec/810/studynotes.htm	http://plato.stanford.edu/entries/Game theory/
2	http://www.cit.gu.edu.au/~s2130677/teaching/Agents/Workshops/lecture07.pdf	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
3	http://home.earthlink.net/~peter.a.taylor/manifes2.htm	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
4	http://aufrecht.org/blog/swcat/39172	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
5	http://www.concurringopinions.com/archives/economic_analysis_of_law/index.html	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
6	http://dotearth.blogs.nytimes.com/2008/01/13/a-starting-point-for-productive-climate-discourse/index.html?ex=1357966800&en=2de12bb5c6f809de&ei=5088&partner=rssnyt&emc=rss	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf
7	http://aws.typepad.com/aws/2005/01/	http://www.kestencgreen.com/kgthesis.pdf
8	http://www.ferc.gov/legal/maj-ord-reg/land-docs/oligoply.pdf	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
9	http://www.drownout.com/blog/archives/cat_reading_list.html	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf
10	http://osnews.com/comments/10354	http://www.indiana.edu/~workshop/wsl/gamethe.htm

Table 4.3.5 Results of *Consensus* method and search engines for very complex query

Consensus	**Ask.com**	**Live**	**Interia**	**Yahoo!**	**Google**
Set Coverage	10%	20%	90%	20%	40%
URL to URL	0%	0%	30%	0%	0%

Table 4.3.6 Coverage of *Consensus* method and search engines for very complex query

It can be observed, from the table above that *Consensus* method presents the highest set-coverage of all three methods. The result set of *Interia* engine is the most covered (90%) result set of all search engines. The lowest coverage (10%) is in the case of the *Ask.com* search engine. Also the result set of *Interia* engine is the most position-wise covered. Other result sets have 0% of URL to URL coverage.

In this case, like for the previous ones, the *Consensus* answer was said to be inconsistent. Once again, this is because of large dispersion of result sets provided by all search engines. Levenshtein distance, as being highly dependent on the URL positioning, is returning large values of distances between result sets, thus resulting in the final answer being said to be inconsistent. The URL to URL coverage exposes this fact further. Only the one engine has the non-zero URL to URL coverage, which results in high value of the average of distances from consensus answer to all result sets. Nevertheless, this method reflects the most common view of all search engines as it should, since the building of the result set is purely based on average ranks of URLs. In the final result set there are the most common URLs which are present, as top results, in most of the search engines.

The following part presents the subjective comparison the result sets returned by the three methods.

#	Auction	Game theory	Consensus (inconsistent)
1	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholarr	http://www.msu.edu/course/aec/810/studynotes.htm	http://plato.stanford.edu/entries/Game theory/
2	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholarr	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Raiffa" intitle:"Negotiation Analysis: The Science and Art of ..." &um=1&oi=scholarr	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf
3	http://plato.stanford.edu/entries/Game theory/	http://www.ejournal.unam.mx/cys/vol03-04/CYS03407.pdf	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07
4	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Day" intitle:"EXPRESSING PREFERENCES WITH PRICE-VECTOR AGENTS IN ..." &um=1&oi=scholarr	http://www.primisonline.com/cgi-bin/POL_program.cgi?programCode=HBSNA&context=	http://www.msu.edu/course/aec/810/studynotes.htm
5	http://www-static.cc.gatech.edu/~jp/Papers/Zagal et al - Collaborative Games - Lessons learned from boardgames.pdf	http://lsolum.blogspot.com/archives/2005_11_01_lsolum_archive.html	http://www.people.hbs.edu/mbazerman/curriculum_vitae.html
6	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Martimort" intitle:"Delegated Common Agency under Moral Hazard and the ..." &um=1&oi=scholarr	http://doi.ieeecomputersociety.org/10.1109/TSE.2003.1237173
7	http://links.jstor.org/sici?sici=0192-5121(199704)18:2<121:CAOJIN>2.0.CO;2-K	http://plato.stanford.edu/entries/Game theory/	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf
8	http://ieeexplore.ieee.org/iel5/32/27736/01237173.pdf	http://scholar.google.com/scholar?num=20&hl=en&ie=UTF-8&q=author:"Johansson" intitle:"On Coordination in Multi-agent Systems" &um=1&oi=scholar	http://ieeexplore.ieee.org/iel5/8856/4266804/04266807.pdf
9	http://www.indiana.edu/~workshop/wsl/gamethe.htm	http://lsolum.blogspot.com/archives/2005_09_01_lsolum_archive.html	http://www.indiana.edu/~workshop/wsl/gamethe.htm
10	http://links.jstor.org/sici?sici=0020-8833(199703)41:1<87:PTRCAI>2.0.CO;2-I	http://updatecenter.britannica.com/eb/article?articleId=109420&pid=ursd07	http://www.kestencgreen.com/kgthesis.pdf

Table 4.3.7 Results of methods for very complex query

	Auction	Consensus	Game theory
Auction	–	30%	30%
Consensus	10%	–	40%
Game theory	0%	0%	–

Table 4.3.8 Coverage of methods for very complex query

Table above illustrates coverage of result sets returned by the three methods. It can be observed that highest set-coverage (40%) is between result sets returned by *Consensus* and *Game theory* methods. In other cases the set-coverage is equal to 30%. URL to URL coverage is very low and it is non-zero (10%) only in case of *Auction* and *Consensus* method.

As in the previous cases here, to compare the results content from 3 top most URLs from each result set are investigated. *Auction* method as the 2 top most URLs returned suggestions provided by *Google* search engine, to search for the aforementioned query in its *Google Scholar* service. This however, does not point to any resource itself. As the third URL *Auction* method returned the *Stanford Encyclopedia of philosophy* webpage containing definition of Game theory. *Game theory* as the first URL returned the webpage containing information about institutional and behavioral economics. The second URL is the *Google* search engine suggestion, the same as the first URL returned by the *Auction* method. Third URL is a document about multi-agent systems and utilizing those as an approach to distributed artificial intelligence. *Consensus* as the two top most URL method returns those which were returned as 3rd ones by the *Game theory* and *Auction* method. As the third URL it presented the article from *Encyclopaedia Britannica* about the Game theory in general.

That said it seems that the search engines, when providing the URLs for this long and complex query, most of the time taken into account the term `Game theory`, rather than `consensus`, `conflict` or `auction` terms. Probably out of these topics, the Game theory is the most popular topic which could be found in Internet. This may not be the results which one could be expecting when issuing this query, but since this query is very complex, the search engines may have gotten confused. But this is a work of the algorithms presented here, to remove the confusion from the result sets, thus providing the best possible results. Nevertheless, here is the subjective comparison of the results:

1. *Consensus* – no link which is a suggestion to use some other search engine, the URLs which were described, all point to the real resource
2. *Game theory* – one link which is a suggestion to use other search, two URLs pointed to a real resources
3. *Auction* – two links were suggestion to use another search, one pointed to a real resource

There is no comparison of methods vs. *MySpiders* for this query. Query proven to complex for *MySpiders* to handle – it is a content based search and probably it did not find any resource which contained all of the terms from the issued query. In turn *MySpiders* did not return any URL for this query.

This part will present the summary of comparison tests. The conclusions which appeared after each part of comparison will be summarized here.

Auction method is a method which is highly dependent on each separate single result sets of search engines rather than the combined view of all search engines. The results presented here show, that no matter if the URL is in many result sets of the search engines, it may not still be taken as a part of the final result. Instead, a lot of URLs are returned, which appear in only one of the result sets provided by search engines. *Game theory* method also has the tendencies to not to take the whole combined view of the search engines into the final result. However, if an URL was at the top most places of more than one result set provided by search engines, it probably will be incorporated (with high probability) into the final result set provided by this method. It may not be the average place of such URL, but still it probably will appear somewhere at the bottom of the list. *Consensus* method in general returned the results which are the most common view of the search engines. However in three tested cases all result sets which were returned were inconsistent according to consensus theory. This happens probably due to the high position-wise URL dispersion throughout the result sets. There are situations where an URL is, for instance, on the 1st place in one result set on the 6th place in another result set and on the 3rd. This results in Levenshtein distance to be very large and thus resulting in result sets being dispersed. Nevertheless if one was not to take the consistency into account, the *Consensus* method provided the results which are best overall.

5. Final remarks

In this thesis application of the three approaches (*Game theory, Auction* and *Consensus based one*) for combining information was presented. These methods were implemented and tested, thus providing insight on the main aim of the thesis: to find out if those approaches could be applied to the combining information problem, to check if those methods are able to improve the quality of retrieved information by consolidating results of search engines into one result set.

The main benefit of those approaches is that they filter the result sets in some extent, providing better out of box results, than ordinary search engines. The results provided by those methods were compared to each of the search engine that contributed to the methods answer. It turned out that result sets created by through combination of URLs provided by search engines provided better insight on the query. For simple query results were extraordinary – the information provided by every algorithm was highly relevant. For complex queries, *Consensus* and *Game theory* based methods provided better results than those provided by *Auction*. In general, *Auction* method, provided worst (this is a subjective opinion) results of all methods, but still when dealing with simple queries, quality of those was comparable with other methods.

Those methods are ranking based methods – they do not take into account the content of the resources. Still those were compared with content based approach of information combining which is *MySpiders* system. It turned out, that the tested methods provided very similar results to those provided by the *MySpiders*. However, the *MySpiders* system provided results only for the two simpler queries. The last query has proven, to be too much for the *MySpiders* to handle. Nevertheless, when comparing *MySpiders* to the methods, after issuing simpler queries, results were very similar, thus proving that the content based approaches may not necessarily be better than the purely rank based ones.

As a possible future work, one could introduce rankings of the search engines, based on previous results of the methods. Rankings of those engines were implemented in very simple manner, but those were not tested (and not used during testing of the methods) as it was not the main aim of the thesis. However, introducing engines' ranks would allow for further filtering of the answers, by giving handicap to the engines which contributed in smaller extent to the final results of the methods.

Another possibility is to extend of the tool that was used to testing by adding some new methods for combining information. The tool was written in multi-agent environment, so is easily extensible and it could be used for testing another, more even more complex algorithms for combining information from multiple Internet sources.

6. References

[1] Nguyen N.T., Ganzha M., Paprzycki M., *A Consensus-based Approach for Information Retrieval in Internet.* Lecture Notes in Computer Science 3993 (2006) 208-215.

[2] Nguyen N.T., *Processing Inconsistency of Knowledge at Semantic Level.* Journal of Universal Computer Science 11, 2 (2005) 285-302.

[3] Nguyen N.T, Małowiecki M., *Consistency Measures for Conflict Profiles.* LNCS Transactions on Rough Sets 1 (2004) 169-186.

[4] Nguyen N.T., *Consensus System for Solving Conflicts in Distributed Systems.* Journal of Information Sciences 147 (2002) 91-122.

[5] Nguyen N.T., *Methods for Achieving Susceptibility to Consensus for Conflict Profiles.* Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology 17, 3 (2006) 219-229.

[6] Nguyen N.T., *An Inquiry into Distributed Consensus.*

[7] Santana L.E.A., Canuto A. M. P., Junior Xavier J.C., Campos A.M.C., *A Comparative Analysis of Data Distribution Methods in an Agent-based Neural System for Classification Tasks.* Proceedings of the Sixth International Conference on Hybrid Intelligent Systems (2006) 9.

[8] Santana L.E.A., Canuto A. M. P., Abreu M.C.C., *Analyzing the Performance of an Agent-based Neural System for Classification Tasks Using Data Distribution among the Agents.* International Joint Conference on Neural Networks (2006) 2951-2958.

[9] Canuto A. M. P., Abreu M.C.C., *Analyzing the Benefits of Using a Fuzzy-Neuro Model in the Accuracy of the NeurAge System: an Agent-Based System for Classification Tasks.* International Joint Conference on Neural Networks (2006) 2959-2966.

[10] Błażowski A., Nguyen N.T., *AGWI - Multi-agent System Aiding Information Retrieval in Internet.* In Proceedings of SOFSEM 2005. Lecture Notes in Computer Science 3381 (2005) 399-403.

[11] Menczer F., *Complementing Search Engines with Online Web Mining Agents.* Decision Support Systems 35 (2003) 195-212.

[12] JADE Homepage (http://jade.tilab.com).

[13] Vaucher J., Ncho A., *Jade Tutorial and Primer.* 2003 (http://www.iro.umontreal.ca/~vaucher/Agents/Jade/JadePrimer.html)

[14] Agent Technology Group, *JADE implementation short guide.* (http://agents.felk.cvut.cz/teaching/ui2/JADE_tutorial.htm)

[15] Kessler R.R., Griss M.L., *Making Java Agents and JBuilder Work for You.* Half-day BORCON Pre-Conference Tutorial: November 1, 2003. (http://www.soe.ucsc.edu/research/agents/borcon/)

[16] Sun Microsystems Inc. *Final Release of the Servlet 2.5 Specification.* 2006. (http://jcp.org/aboutJava/communityprocess/mrel/jsr154/index.html).

[17] Sun Microsystems Inc. *Final Release of the JavaServer Pages Specification.* 2006. (http://jcp.org/aboutJava/communityprocess/final/jsr245/index.html).

[18] Encyclopedia Wikipedia (http://en.wikipedia.org).

Table of listings

Table of figures

Table of tables

Warszawa, dnia 21.04.2008r.

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem; „Combining Information from Multiple Internet Sources”, której promotorem jest dr Marcin Paprzycki, wykonałem samodzielnie, co poświadczam własnoręcznym podpisem.

.....