MIDDLE EAST

TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

DEPARTMENT

Requirement Analysis Report

This document is not intended as a de facto standard for the future of the project and the items presented here are subject to change by SciComp team, after being approved by TA and ETC.

- Myrzabek MURATALIEV – *1408665*
- Serdar DALGIÇ – *1448570*
- Haşim TİMURTAŞ – *1449149*
- Barış YÜKSEL – *1449826*

Table Of Contents

1. INTRODUCTION

This analysis report is prepared by the SciComp Team to show the progress they have shown during their senior design project. The report includes detailed definition of their problem domain and further analysis on functional and technical requirements. System models and management plans are also included.

1.1. Background of the Project and the Team

There are many challenging problems that require a heavy loaded computation, although their algorithmic complexities are as low as they can be. The need of higher computation abilities and the restriction of power consumptions pushed the engineers to develop a new technology called Parallel Computing.

A good example of using Parallel Computing is experimenting with data mining problems. In recent years, data mining has been widely used in area of science and engineering, such as bioinformatics, genetics, medicine, education and electric power engineering. Cluster analysis is one of these implementations, the one which we are interested in. Clustering is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure. K-means algorithm is a part of partitional clustering algorithms, of which are typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering. X-means algorithm is somewhat derivative of this algorithm.

Weka (Waikato Environment for Knowledge Analysis) is a free software (licensed by GNU General Public License) and a popular suite of machine learning software written in Java, developed at the University of Waikato. Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. The Cluster panel gives access to the clustering techniques in Weka, e.g., the simple k-means algorithm.

Within these aspects, we decided to drive onto the subject of parallelizing the Weka tool. It's an advantage for us to see people around us such as assistants and professors who are familiar with Weka tool.

Members of the SciComp Team, all consists of METU CEng senior students. Members of the SciComp Team have taken (and are still taking) several courses related to data mining and clusteralization. All members of the team are involved in Java programming and familiar with Java programming tools and concept. These consequences are the reasons why this project is the very thing for the team.

1.2. Problem Definition

The project consists of the task to make the current version of Weka evolve into a parallelized variant which is capable of data or task parallelism. We plan to choose the type of parallelism in latter parts of the project.

This can be done by parallelizing the *Instance class* of the API. While coping with this issue, we plan to be guided by X-Means clustering algorithm's execution in Weka.

1.3. Project Goals and Scope

The main goals of the project is below:

1. There are several approaches that are intended to parallelize Weka; however none of these can be considered as a solid solution for the parallelization issue. A fork of Weka called Weka-Parallel[1] exists; which is created with the intention of being able to run the cross-validation portion of any given classifier very quickly. Instead of dealing with such kind of data parallelism; we want to focus on both task parallelism that would enable doing concurrent tasks and data parallelism which will enhance Weka's capabilities on data mining.

2. Our Weka implementation is planned to be capable of parallelizing K – means / X – means algorithm with an efficient result.
3. Our Weka implementation is also needed to be accomplishing not only X – means algorithm but also any other parallelization tasks for given algorithms.
4. An intuitive and user friendly GUI that makes it easy to use the parallelized version of Weka will be provided.

The scope of this project involves these parts:

1. Analysis and documentation of various parallelization examples in data mining and clusteralization world.
2. Development of a detailed parallel API specificiations.
3. Design of an appropriate tool to reflect our inquiries.
4. Implementation of this tool.
5. Documentation of the software product.

2. SOFTWARE DEVELOPMENT PROCESS

2.1. Software Development Process Model

In order to choose the optimal software development process model for this project, one has to consider the following fact: Although the modules of the Weka software depend on each other, *Instances class* is one of the classes that has purity on dependency and forming the start point of the journey. As the methods of the class varies and parallelization issues related to *Instances class* can be designed in parallel, the development process can be coherently analogous design.

Another point of *Instances class*, which may be considered as a pro or con, or both; this class holds control of many actions and any improvement or change in this class will affect more than one positions. A point to keep an eye on is to take action step by step.

Thus, we considered an **agile development method** with **spiral development model** is what we are looking for. When the inner nature of SciComp Team and the dynamics between team members is considered, agile development model that consists of small increments with minimal planning, instead of long-term planning and iterations as short time frames is the optimum solution. In addition to this, we also prefer to work in a Parallel Software Design Model (PSDM).

2.2. Team Organization

The specifications of the problem is clear. However, in order to work in parallel, modulation of the problem is a significantly critical point. The code we are going to deal with is specific but partitioning the problem into pieces is still a big deal.

Segmenting the problem into parts seems to be a decision that all members of the team should agree on; so this concept brings us to chose the **"Democratic Decentralized" (DD)** model as this organizational schema is the most suitable one for us.

This model gives us the ability to split the problem into pieces easily and organize/reorganize subgroups given to a spesific topic.

2.3. Major Constraints

We determined five types of constraints:

2.3.1. Project Deadline

The project is intended to finish by the end of May 2009. Thus the approximate time calculated for the project is 7 months. A detailed time schedule can be seen in the appendix part.

2.3.2. Programming Language Constraints

As Weka tool is written in Java programming language, We plan to develop our code in Java. Eclipse usage will improve the speed of coding and design processes. Openmpi and erlang codes used in parallel computing is present also. In the design process and for various inspirations neede in latter stages, these languages and sample codes seems to be vital. Openmpi to Java conversion kind of codes are expected to be searched too..

2.3.3. Data Constraints

Sample data of data mining sets is needed in the testing phases of the project. Various samples are present when searched via Google.

2.3.4. Execution Speed Constraints

This is the vital part of our project. The reason why parallelization is needed. Our design needs to be sufficient enough to improve the efficiency of the work done by parallel computers when compared to single computers. The speed that we plan to gain should be a satisfying result that worths parallelization.

2.3.5. User Interface Constraints

An intended GUI with parallelization options and functions is planned to be accomplished. The latter stages and the result of the parallelization issue is going to determine this part. We plan to implement a user interface that would give the chance to control the type of the parallelization and memory, cpu needs.

3. LITERATURE SURVEY AND RELEVANT SEARCH

First, when searching about parallel computing; we saw that there are several different forms of parallel computing: bit – level, instruction – level, data and task parallelism.

Bit-level parallelism is a form of parallel computing based on increasing processor word size, depending on very-large-scale integration (VLSI) technology. Enhancements in computers designs were done by increasing bit-level parallelism.[2]

Instruction-level parallelism (ILP) is a measure of how many of the operations in a computer program can be performed simultaneously.[3]

Data parallelism (also known as **loop-level parallelism**) is a form of parallelization of computing across multiple processors in parallel computing environments. Data parallelism focuses on distributing the data across different parallel computing nodes.[4]

Task parallelism (also known as **function parallelism** and **control parallelism**) is a form of parallelization of computer code across multiple processors in parallel computing environments. Task parallelism focusses on distributing execution processes (threads) across different parallel computing nodes. It contrasts to data parallelism as another form of parallelism. Task parallelism emphasizes the distributed (parallelized) nature of the processing (i.e. threads), as opposed to the data (data parallelism). Most real programs fall somewhere on a continuum between Task parallelism and Data parallelism.[5]

After we have learned the types of parallel computing; we needed to learn how to do the parallel computing and on what domain shall we study. So we have deepened our research on that perspective. After our survey started to give first outcomes, we have seen that the following issues are closely related to our project and we can get benefit from them:

1. Data Mining
2. Weka Tool
3. X – means and K – means algorithms

3.1. Data Mining

According to Kurt Thearling, Ph.D; data mining is the automated extraction of information from (large) databases. There exists three keywords here:

1. Automated
2. Hidden
3. Predictive

That means we will make guesses from the information we already had. Data mining is usually used by business intelligence organizations, and financial analysts; but it is increasingly being used in the sciences to extract information from the enormous data sets generated by modern experimental and observational methods. When dealing data mining, we generally does not mean easy inductions, we mean that data mining provides information that may be difficult to obtain otherwise. To illustrate, when the data collected involves individual people, there are many questions concerning privacy, legality, and ethics.

The process of data mining consists of three stages:

1. The initial exploration,
2. Model building or pattern identification with validation/verification, and
3. Deployment (i.e., the application of the model to new data in order to generate predictions).[6]

3.1.1. Parallel Data Mining

A common feature of most data mining tasks is that they are resource intensive and operate on large sets of data. Data sources measuring in gigabytes or terabytes are now quite common in data mining. For example, WalMart records 20 million transactions and AT&T produces 275 million call records every day. This calls for fast data mining algorithms that can mine huge databases in a reasonable amount of time.

However, despite the many algorithmic improvements proposed in many serial algorithms, the large size and dimensionality of many databases makes data mining tasks too slow and too big to be run on a single processor machine. It is therefore a growing need to develop efficient parallel data mining algorithms that can run on a distributed system. [7]

3.2. Weka Tool

After we have learned a bit about data mining, its power and power consumption; we found out that we need some algorithms to work on large sets of data. After some survey (we will mention later), we reached Weka.

Weka (**W**aikato **E**nvironment for **K**nowledge **A**nalysis) is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.[8]

The overall goal of the weka project is to build a state-of-the-art facility for developing machine learning (ML) techniques and to apply them to real-world data mining problems.[8]

Data Mining studies algorithms and computational paradigms that allow computers to discover structure in databases, perform prediction and forecasting, and generally improve their performance through interaction with data. Machine learning is concerned with building computer systems that have the ability to improve their performance in a given domain through experience.

Machine learning and data mining are becoming increasingly important areas of engineering and computer science and have been successfully applied to a wide range of problems in science and engineering.

3.3. K – means and X – means Clustering (grouping) Algorithms

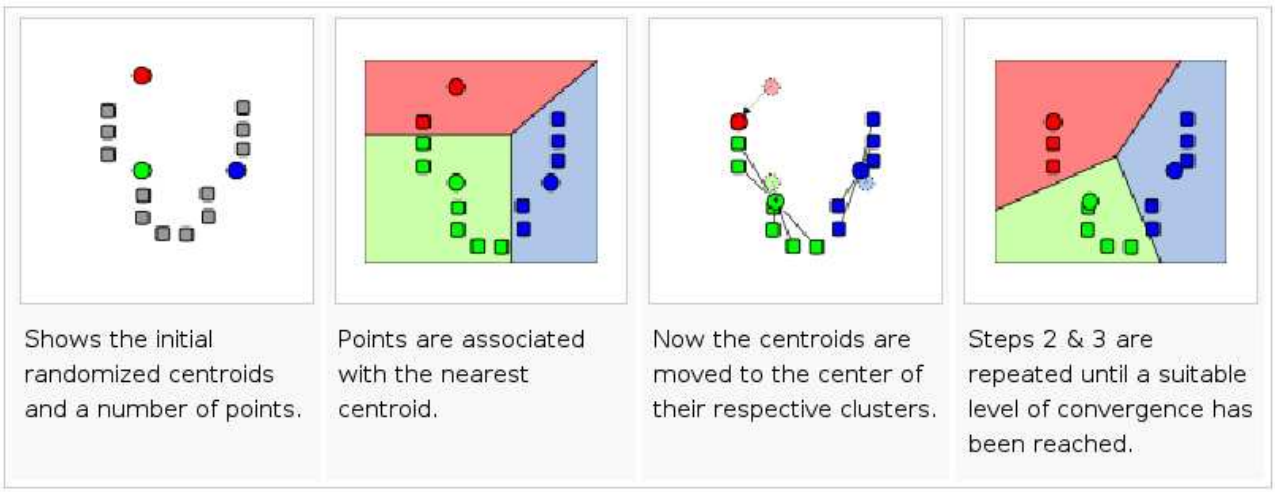
The ***k*-means algorithm** is an algorithm to cluster *n* objects based on attributes into *k* partitions, $k < n$.[9]

The objective it tries to achieve is to minimize total intra – cluster variance, or the squared error function

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

where there are *k* clusters S_i, $i = 1, 2, ..., k$, and μ_i is the centroid or mean point of all the points $x_j \in S_i$.[9]

To demonstrate the K – means clustering algorithm here is a picture which its initial centroid places are choosen randomly.



Here, we talk about K – means so much because we want to use it as a proof of concept for our parallelization project. Actually, our product will work for different algorithms such as X-means clustering algorithm.

3.4. Interview and Brainstorming with Asst. Çağatay Çallı

Up to here; we have mentioned about our literature survey. However; we haven't still mentioned yet about our “angel” for the project. Stumbling around and trying to find a spesific topic for our project, after a conversation with Asst. Prof. Dr. Pınar Şenkul Karagöz, She offered us to talk with Asst. Çağatay Çallı about parallelization need of Weka project. Çağatay welcomed us well, and behaved in an helpful manner. He informed us about Weka project and what was the missing thing in terms of parallelization. It was a good introduction for us which cleared our mind and increased our motivation about the topic.

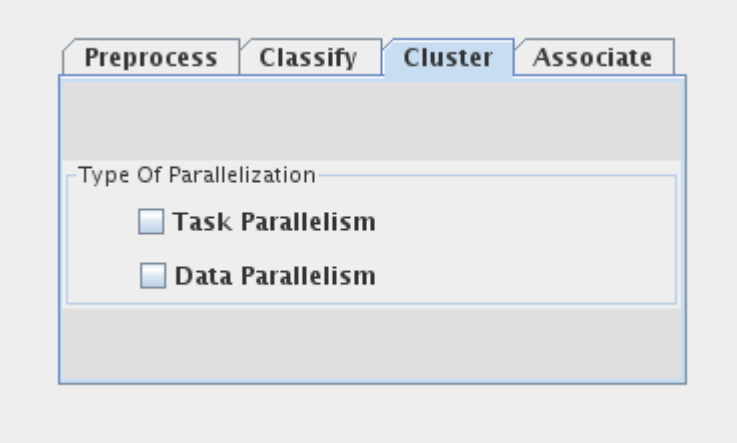
3.5. Getting Contact With Weka Developers and Dan Pelleg

After we gained some approach on the topic, we started communicating with the key persons who would lighten our roadmap. We are tracking weka mailing list which receives 15 – 20 mails per day. We learned useful tips from this mailing list. We also contacted with Dan Pelleg [10] ; eager to hear from him, we are waiting for his reply.

4. PROJECT REQUIREMENTS

4.1. UI REQUIREMENTS

Except the fact that we focus on the stability and the speed of the program, user interface requirements are still a significant point for us to step on. At the end, users will be able to select what kind of parallelism they prefer, which properties of the computers they will persist (more ram or more cpu).



An Example of Choosing Parallelization Type

Most of our work is general based on inner parts of Weka, implementations including core libraries, algorithm – based improving and parallelizations. UI requirements are not a big concern for us and as a user requirement.

4.2. INPUT – OUTPUT REQUIREMENTS

Attribute – Relation File Format (ARFF) type files are present when googled with keywords “weka, input” etc. An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. These file samples also available in Weka environment.

4.3. HARDWARE & SOFTWARE REQUIREMENTS

As Weka is written on Java, it can run on any environment containing a JRE. We plan to do our development on linux environment importing Weka source code into Eclipse and modifying the code in Eclipse.

But for testing phases, we need to use a HPC environment, that's NAR, which will be supplied to our project under certain circumstances. The code we created will need a parallel environment, so in latter phases of the project, we will desperately need to run our test cases on HPC environment.

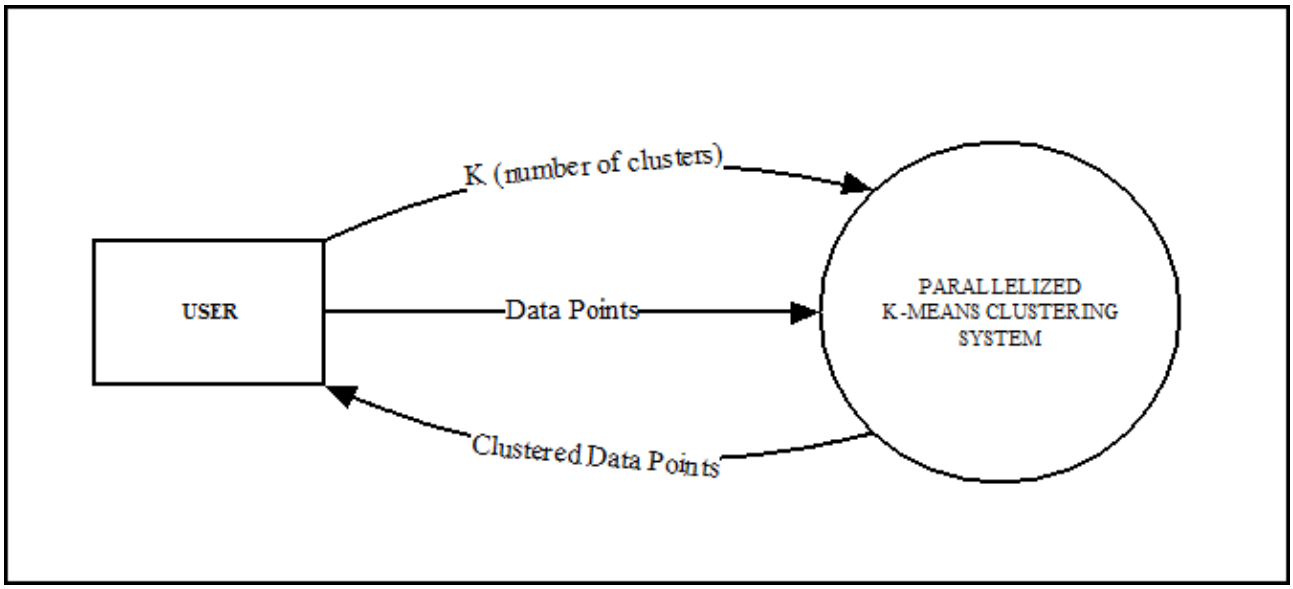
5. SYSTEM ANALYSIS AND MODELLING

5.1. Structured Analysis – Function Model

In this section, data flow diagram of parallelized K-means algorithm will be introduced in two levels: level 0 and level 1. Level 0 diagram will represent major input – output data of the system. Level 1 diagram will show detailed description of modules with input outputs.

5.1.1 Level 0 of DFD

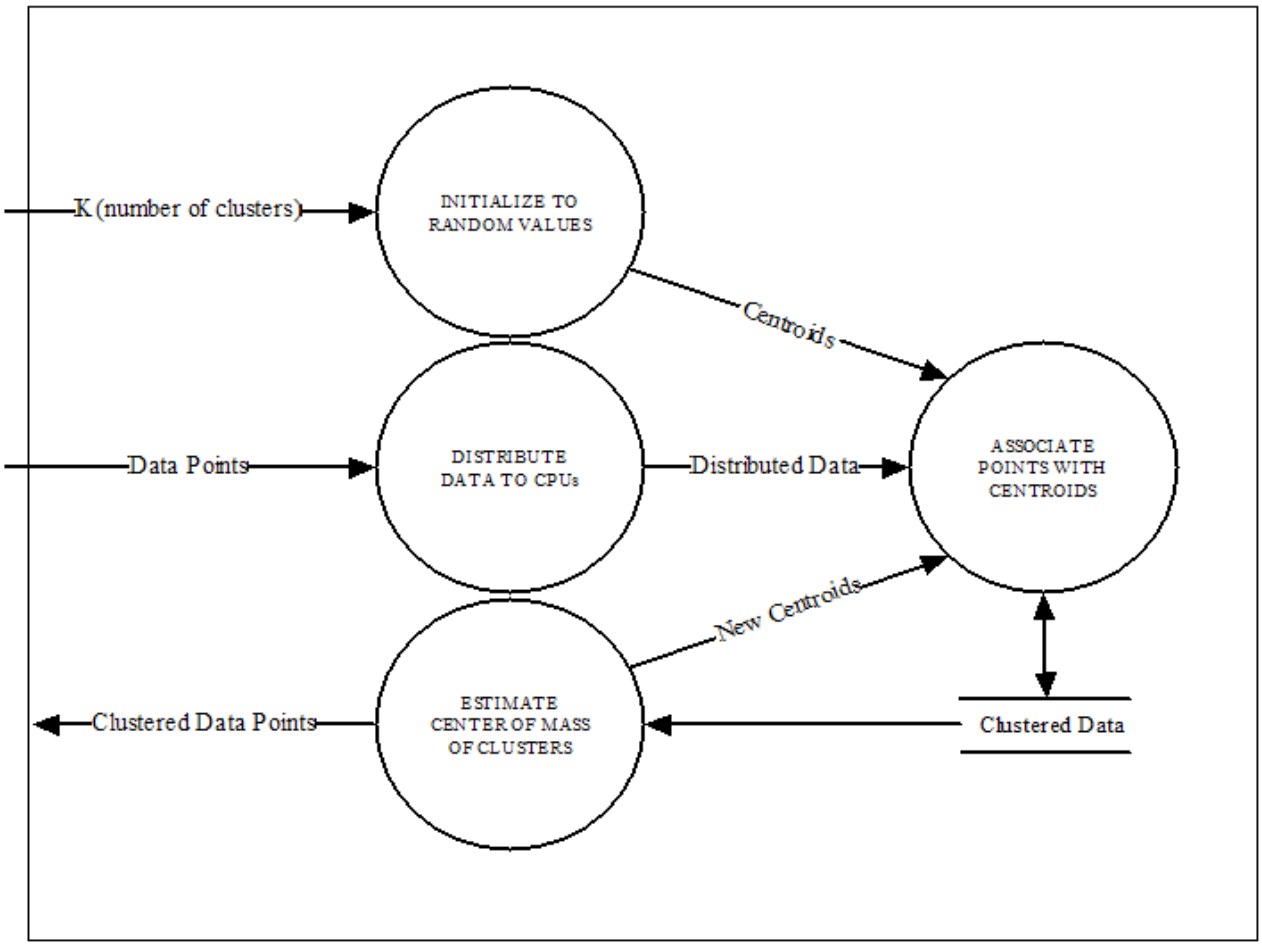
Level 0 of DFD is shown in Figure 1; it is general overview of the system.



Level 0 DFD

5.1.2 Level 1 of DFD

Level 1 of DFD is shown in Figure 2; it is more detailed view of Level 0 DFD. The system is divided into four processes; Initialize to random values, Distribute data to CPUs, Associate points with centroids and Estimate center of mass of clusters.



Level 1 DFD

5.1.3 Explanation of DFD

Initialize to random values*:* K number of centroids are created and initialized to random numbers.

Distribute data to CPUs*:* CPUs are assigned with equal unique ranges on array of data points. In other words, data points are distributed among CPUs to be processed parallely.

Associate points with centroids*:* All data points are associated with closest centroids and this process is achieved by multiple CPUs. Each CPU gets one cluster of data points to process and as a result, many clusters will be processed concurrently representing data parallelism. In general overview, as whole task is divided into parts

which are completed parallelly by multiple machines, concept of task parallelism will also be conserved. K number of clusters also will be constructed.

Estimate center of mass of clusters: Centroid locations are re-estimated by taking, for each centroid, the center of mass of points associated with it. Each CPU will calculate center of mass of one cluster. Many CPUs will process parallely becoming a good example of Data Parallelism.

6. PROJECT SCHEDULE

Timing is one of the most important tasks we have concerned about. We need more literature survey and API/code investigations for Weka. Thus, we needed to put the whole procedure into tasks and tried to give some deadlines which will help us manage our time efficiently. Details about schedule and a gannt chart showing estimated deadlines are available in Appendix – A.

7. RISK MANAGEMENT

7.1 Team Management

Throughout the project, each member of the team should be aware of both his own and other members' responsibilities. This is very important, as any weakness of team member may have big negative effect on the flow of project. One of the expected risks can be unbalanced distribution of work but it can easily be avoided by objectively assessing the complexities of parts of the work. Additionally, sharing of responsibility also can be one of the ways to minimize these risk impacts.

7.2 Project Management

Since firstly we have to fully visualize the main functions of Weka tool, we may confront many complexities in understanding details as it requires deep knowledge about the subject. We may need external help or advice from people who

are familiar with this issue. Another important thing is : meeting the deadlines must be followed very strictly, because any delay occurring in any level of the project will affect the succeeding levels catastrophically.

7.2 Technical Problems

Finding proper libraries for implementation and integration of these libraries can be counted as a risk to be taken care of since it would be time consuming to recreate these libraries and it is considered as a technical problem, which may create marginal impacts on the project.

8. Conclusion

Although it has not been more than one or two months, we think we gained a remarkable improvement on data mining issues. The documentation and sample available in the environment encourages us to possess the project more and more everyday. This requirement analysis would evolve into better versions as we reshape our needs in time while getting to know Weka source code and parallelization concepts.

9. References

[1] “Weka-Parallel was created with the intention of being able to run the cross-validation portion of any given classifier very quickly.” http://weka-parallel.sourceforge.net/

[2] David E. Culler, Jaswinder Pal Singh, Anoop Gupta. Parallel Computer Architecture - A Hardware/Software Approach. Morgan Kaufmann Publishers, 1999. ISBN 1558603433, pg 15

[3] http://en.wikipedia.org/wiki/Instruction_level_parallelism

[4] http://en.wikipedia.org/wiki/Data_parallelism

[5] http://en.wikipedia.org/wiki/Task_parallelism

[6] http://www.statsoft.nl/uk/textbook/stdatmin.html

[7] http://www.inf.ed.ac.uk/undergraduate/projects/mathiasengvall/

[8] http://www.cs.waikato.ac.nz/ml/weka/

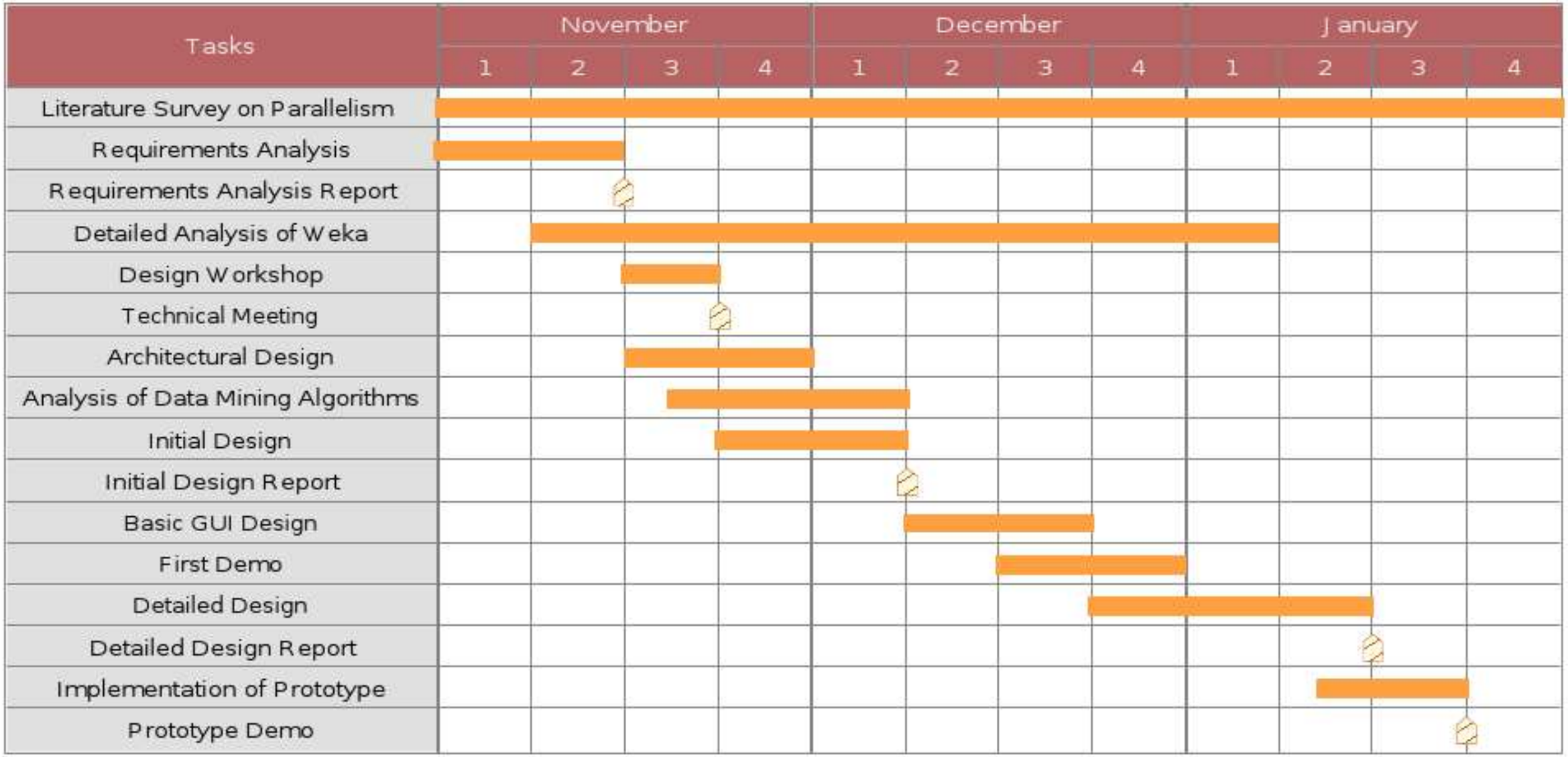
[9] http://en.wikipedia.org/wiki/K-means_algorithm

[10] Dan Pelleg's Website at Carnegie Mellon University : http://www.cs.cmu.edu/~dpelleg/

10. Appendix – A

Gannt Chart of our planned deadlines.

GANTT CHART - PROJECT



Submission

Duration