The Islamic University–Gaza
Research and Postgraduate Affairs
Faculty of Engineering
Computer Engineering Department

الجامعـــــة الإســـــلاميـــــة – غـزة
عمادة البحث العلمي والدراســات العليا
كليـــــــــــة الهندســـــــــــة
قســـم هندســـــة الحاســـــــوب

Clustering Big Data Based On IWC-PSO and MapReduce

عنقدة البيانات الكبيرة بالاعتماد على خوارزميات تحسين الأداء والحوسبة الموزّعة

By

Ahmed Z. Skaik

Supervised by

Associate Prof. Wesam M. Ashour

Thesis Submitted in Partial Fulfillment of the Requirements

For the Degree of Master in Computer Engineering

(1439 هـ – 2018م)

إقـــــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

Clustering Big Data Based On IWC-PSO and MapReduce

عنقدة البيانات الكبيرة بالاعتماد على خوارزميات تحسين الأداء والحوسبة الموزّعة

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وأن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل الاخرين لنيل درجة أو لقب علمي أو بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.

Declaration

I understand the nature of plagiarism, and I am aware of the University's policy on this.

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted by others elsewhere for any other degree or qualification.

Student's name:	أحمد زكريا سكيك	اسم الطالب:
Signature:		التوقيع:
Date:	13/10/2018	التاريخ:

بسم الله الرحمن الرحيم

الجامعة الإسلامية بغزة
The Islamic University of Gaza
عمادة البحث العلمي والدراسات العليا
هاتف داخلي: 1150

الرقم Ref ...ج.س.غ/35/...
التاريخ Date ...2018/11/24م...

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة عمادة البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ **احمد زكريا ابراهيم سكيك** لنيل درجة الماجستير في كلية **الهندسة**/ برنامج **هندسة الحاسوب** وموضوعها:

عنقدة البيانات الكبيرة بالاعتماد على خوارزميات تحسين الأداء والحوسبة الموزّعة

Clustering Big Data Based On IWC-PSO and MapReduce

وبعد المناقشة التي تمت اليوم **الثلاثاء 26 صفر 1440**هـ الموافق **2018/11/06**م الساعة التاسعة صباحاً، في قاعة اجتماعات الكلية اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. وسام محمود عاشور	**مشرفاً ورئيساً**	.....
د. محمد أحمد الحنجوري	**مناقشاً داخلياً**	.....
د. إيهاب صلاح زقوت	**مناقشاً خارجياً**	.....

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في **كلية الهندسة/برنامج هندسة الحاسوب**.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله تعالى ولزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق،،،

عميد البحث العلمي والدراسات العليا

أ.د. مازن إسماعيل هنية

P.O. Box 108, Rimal, Gaza, Palestine Tel: + 970 (8) 264 4400 هاتف: Fax: + 970 (8) 264 4800 فاكس: ص.ب. 108 الرمال. غزة. فلسطين
public@iugaza.edu.ps www.iugaza.edu.ps

الرقم العام للنسخة	3105925	اللغة	ع

التاريخ: ٢٦/ ١١ /١٨ 20

الموضوع/ استلام النسخة الإلكترونية لرسالة علمية

قامت إدارة المكتبات بالجامعة الإسلامية باستلام **النسخة الإلكترونية** من رسالة

الطالب/ أحمد زكريا إبراهيم سكيك

رقم جامعي: 120162595 قسم: الحاسوب كلية: الهندسة

وتم الاطلاع عليها، ومطابقتها بالنسخة الورقية للرسالة نفسها، ضمن المحددات المبينة أدناه:

- تم إجراء جميع التعديلات التي طلبتها لجنة المناقشة.
- تم توقيع المشرف/المشرفين على النسخة الورقية لاعتمادها كنسخة معدلة ونهائية.
- تم وضع ختم "عمادة الدراسات العليا" على النسخة الورقية لاعتماد توقيع المشرف/المشرفين.
- وجود جميع فصول الرسالة مجمَّعة في ملف (WORD) وآخر (PDF).
- وجود فهرس الرسالة، والملخصين باللغتين العربية والإنجليزية بملفات منفصلة (PDF +WORD)
- تطابق النص في كل صفحة ورقية مع النص في كل صفحة تقابلها في الصفحات الإلكترونية.
- تطابقُ التنسيق في جميع الصفحات (نوع وحجم الخط) بين النسخة الورقية والإلكترونية.

381

ملاحظة: ستقوم إدارة المكتبات بنشر هذه الرسالة كاملة بصيغة (PDF) على موقع المكتبة الإلكتروني.

والله ولي التوفيق،

إدارة المكتبة المركزية

توقيع الطالب

Abstract

There has been a massive growth in the data volume generated of over the recent few years. Manipulate of this huge amounts of data, commonly defined as big data processing, requires considerably extensible data analysis strategies. Data gathering and eliciting aka Clustering is an exploratory data analysis approach used to reveal the implicit groups in the data. Further, clustering is a widely used technique of finding interesting patterns residing in the dataset that are not obviously known.

In medicine for example, all the movements, characteristics and genetic information are collected in order to identify the infected and healthy cells, and attempt to predict and diagnose many diseases in preliminary stages.

Conventional clustering methods were no longer appropriate for use in data mining applications that make use of big data. There have been a plenty of big data clustering algorithms developed in recent years. However the majority of them do not attain clustering with high quality.

Although the K-Means and Inverse Weighted Clustering are accurate and effective in simple and traditional data clustering, they are not operative for large-scaled data.

In this thesis, we will introduce a new approach that conquers the drawbacks of both algorithms, enhance big data clustering and avoids being trapped in a local optimal solution leveraging a powerful optimization algorithm (Particle Swarm Optimizing) so-called PSO and take care of decreasing the time and resources consumption by utilizing a powerful distribution framework Apache/Spark.

The proposed algorithm can be applied in a numerous of real-world applications. Where we prove the leverage of the hybrid algorithm using more than 80 experiments and interestingly the results show that the algorithm can considerably reduce the clustering cost and produce superior clustering outputs in a way that is accurate and fruitful than standalone K-Means, IWC and PSO algorithms.

الملخّص

يعتبر علم البيانات من حيث دراستها وتحليلها والتنقيب فيها واستنباط المعلومات المخفية بداخلها من أشهر العلوم في الوقت الراهن، خصوصاً بعد ظهور وانتشار البيانات الضخمة في مختلف مناحي الحياة وتنوّع أساليب تجميعها ومعالجتها، حيث تقوم الخوارزميات الحديثة وباستخدام تقنيات تعليم الآلية والذكاء الاصطناعي بجمع وتصنيف كل صغيرة وكبيرة في معظم المجالات الحيوية.

في مجال الطب مثلاً، يتم دراسة وتجميع كل حركات وصفات وبيانات الخلايا الحيوية وتمييز الخلايا المصابة والسليمة ومحاولة التنبؤ والتشخيص المبكّر للكثير من الأمراض بناءً على المعلومات التي يتم استنباطها واستقراءها من النتائج. بالإضافة لمجال الطلب يدخل علم ودراسة البيانات الضخمة في كثير من تفاصيل ومجالات العلوم المختلفة. أيضاً فإن دراسة وتحليل ما يتم نشره عبر وسائل التواصل الاجتماعي المختلفة من بيانات وصور وأحداث ومنشورات ومحادثات قد ساهم بشكل فعليّ وكبير في الكثير من القرارات الاقتصادية والسياسية للدول الكبيرة وقد أثّر بشكل كبير على نتائج الانتخابات مثلا في بعض الدول أيضاً.

ولإدارة عملية التنقيب ومعالجة البيانات فإنه يتم استخدام العديد من الخوارزميات والأنظمة التي من شأنها دراسة وتحليل وتصنيف وعنقدة هذه البيانات وفق معايير يتم ضبطها من قبل المهندسين والخبراء للوصول إلى نتائج مرضية وبجودة عالية.

من أشهر هذه الخوارزميات هي خوارزميات الشبكات العصبونية وشجرة القرارات والعنقدة المبنية على أنماط وصفات معينة للبيانات مثل خوارزمية K-Means والنسخة المعدلة منها Inverse Weighted Clustering.

تقترح هذه الرسالة، خوارزمية هجينة جديدة، تعمل على معالجة وعنقدة البيانات الكبيرة وتقترح حلولاً لبعض المشاكل الموجودة في الخوارزميات السابقة. حيث تعمل الخوارزمية المقترحة على استغلال نقاط القوة في خوارزمية Inverse Weighted Clustering من حيث إيجاد مراكز العناقيد والمجموعات المطلوبة بسرعة وكفاءة عالية ودمجها مع خوارزمية تحسين الأداء المشهورة Particle Swarm Optimization في تسريع وتحسين النتائج من خلال تقليل مساحات البحث وحجم البيانات المطلوب معالجتها للعثور على النتائج المرجوّة.

كذلك تضمّن هذا البحث استخدام هذه الخوارزمية في نطاق بيئة عمل خاصة بالأنظمة الموزّعة والمتوازية، حيث تم استخدام معايير Map/Reduce وباستخدام محرّك Apache Spark للاستفادة القصوى من عملية تخفيض الأحمال وموارد الأجهزة الخاصة بعملية معالجة البيانات الضخمة، لما يتطلبه الحجم الهائل من هذه البيانات – التي قد تصل في كثير من الأحيان إلى أكثر من بليون سجل – من مواصفات وأجهزة حاسوبية عالية الموارد ومعالجات فائقة السرعة.

النتائج العملية الموضحة والمرفقة في هذا البحث والتي تم تشمل نتائج أكثر من 80 تجربة عملية، تؤكد على أنه يمكن الاعتماد على هذه الخوارزمية بشكل كامل في معالجة وعنقدة البيانات الكبيرة بكفاءة وسرعة وجودة عالية جداً مقارنةً مع النتائج التي ظهرت مع تجريب النماذج والأنظمة السابقة والتقليدية.

Dedication

To my family who made this accomplishment possible

Acknowledgments

My deep thanks and gratitude are due to Allah, the Almighty, who granted me knowledge. Without his support and guidance, this work would not have been possible, so all praise is to ALLAH.

Then, My thanks to all those who generously contributed their favorite recipes. Without their help, this work would have never been possible.

Also, I would like to express my sense of gratitude and thanks to my supervisor Associate Professor Wesam Ashour for his support, advice, and encouragement throughout this study. I wish also to thank the discussion committee for their efforts.

Table Of Contents

List of Tables

List of Figures

List of Abbreviations

AAPC	Adaptive Affinity Propagation Clustering
API	Application Programming Interface
AP	Affinity Propagation
APP	Adaptive Affinity Propagation
AC	Accuracy
CCIA	Cluster Center Initialization Algorithm
d	Dimensionality of the input space
DAG	Directed Acyclic Graph
FMOPSO	Fuzzy Multi-Objective Particle Swarm Optimization framework
gBest	Global Best Experience
GA	Genetic Algorithm
GPU	Graphics Processing Units
HDFS	Hadoop Distribution File System
I/O	Input/output
IWC	Inverse Weighted Clustering Algorithm
JSC	Java Spark Context
JVM	Java Virtual Machine
KDD	Knowledge Discovery and Data Mining
KM	K-Means Clustering
MPI	Message Passing Interface
ML	Machine Learning
NOWs	Network Of Workstations
PSO	Particle Swarm Optimization
pBest	Particle Best Location
POP	Population Of Particles
RDD	Resilient Distributed Dataset
SOM	Self-Organizing Map
SPMD	Single Program Multiple Data

SSE	Sum of Squared Errors
WEKA	Waikato Environment for Knowledge Analysis
W	Inertia weight
WMS	Warehouse Management Systems

Chapter 1
Introduction

Chapter 1
Introduction

1.1 Background

With the vast growth of search engines capabilities (such as Google, Bing and Yahoo), social media networks (such as Facebook and Twitter), genetic analysis, and a spectacular raise in the devices located everywhere including embedded sensors, smart phones and tablet computers, data volumes generated and processed cross the peta-bytes scale threshold. The question enveloped with these large datasets or big data are about the storage and management, fetching and processing.

One way to gain this problem is to have big data clustered in a consolidated form that will keep it informative version of the whole data. To solve those huge computational demands, we need efficient, scalable, and parallel algorithms either for classifying the huge amount of data into correlative subsets or for increasing the efficiency and effectiveness of its massive computational requirements.

There are many difficulties in dealing with such big amount of data. Thereby data mining is a technique in which valuable information and invisible relationships between data elements are explored. The traditional data mining algorithms is not directly applied on big data as it encounters hurdles to analyse big data. Clustering is one of the leading aspects used for data mining where mining is implement by finding out clusters having homogeneous group of data and used for exploratory data analysis and segmenting. It supports to analyse huge volumes of data visually and statistically therefore helps in picking a right and quick decisions.

Clustering is an unsupervised technique used to process large datasets into correlative groups. No predefined class label are required for the data points. Clustering group datasets into subsets in such a manner that similar instances are grouped together, whereas distinct points belong to distinct groups and these groups so called clusters.

Moreover, one of the important frameworks to deal with the huge computational requirement is the MapReduce approach. MapReduce was first introduced by Google

though their published article about "MapReduce: Simplified Data Processing on Large Clusters." Google‘s MapReduce [1] implementation is a proprietary solution and has not been released to the public yet.

Applications for Data mining suggest the following requirements on clustering techniques in order to be able to deal with big data effectively:

- **Scalability**: Clustering solutions mostly comprise large datasets that include a tremendous data stream. Therefore, highly extensible clustering techniques are required in order to efficiently formulate the clusters.
- **Flexibility to handle heterogeneous attributes**: Data items might have several data types such as categorical, nominal, ordinal, numerical and binary. Several applications might demand clustering and manipulating the data of mixture data types.
- **Discovery of arbitrary forms**: Number of clustering techniques identify clusters in regard to distance measurement such as Euclidean and Manhattan. Those techniques compose globular clusters shape. While some other clustering techniques intended to determine clusters with ambiguous forms such as density-based algorithms.
- **Noisy data insensitiveness**: Commonly, the clustering methods should be insensible to irregular distribution of the data such as outlier and noise data to avoid unsatisfactory and poor clustering outputs.
- **High dimensionality**: Plenty of clustering techniques are identify clusters of low dimensional attributes. Nevertheless, clustering datasets with high dimensions is a demanding action where relations among data points come to be too complex and the density of the data points is also completely low.

1.2 Problem statement

Regardless it is utilized in a variety of applications and presences, the K-Means and other distance-based algorithms are not free of drawbacks, primarily: [2, 3]

- As many clustering methods, the K-Means algorithm assumes that the number of clusters k in the dataset is known in advance which certainly, is not fundamentally true in real-world applications.

- Such as an iterative methods, the K-Means algorithm is particularly sensitive to the primary centroids selection.
- The K-Means and Inverse Weighted K-Means (IWC) algorithms are subjected to converge to local minima.
- Using K-Means with large-scaled data is not a good practice.

It is not only that K-Means that has been affected by the primary starting conditions, many clustering algorithms are affected too. The following Figures 1.1&1.2 describe the problem. Each figure of them is divided as demonstrated bellow into three depicts for more clarification. The figures show a group of data-points consists of three clusters green, brown and blue. In Figure 1.1 iteration 1 show that the three centres are randomly selected.

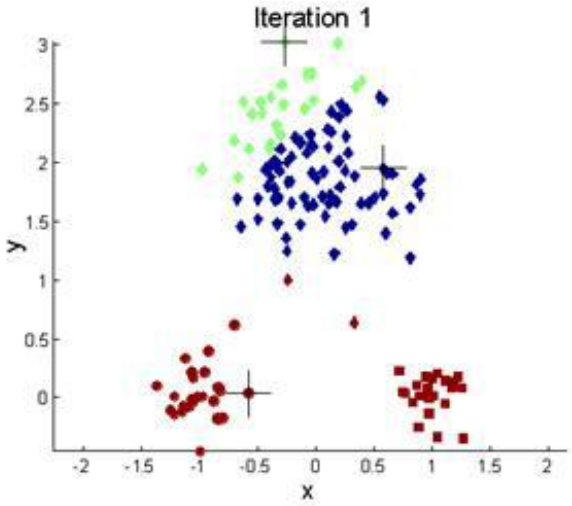


Figure 1.1 K-Means Clustering problem example (a)

While in In Figure 1.2 iterations 2 to 4 are explained. At this stage, the calculations of distances between data points and the selected center is took place and each data point is associated with the closest centroid. And as any iterative algorithm, the distance between all data points and the closest centroid is calculated continuously in each iteration and the centroids are adjusted according to the midpoint calculations. The new center is calculated by taking the average of obtained distances of all data points with their centers. Repeat this calculation and allocation in each iteration of the entire flow.

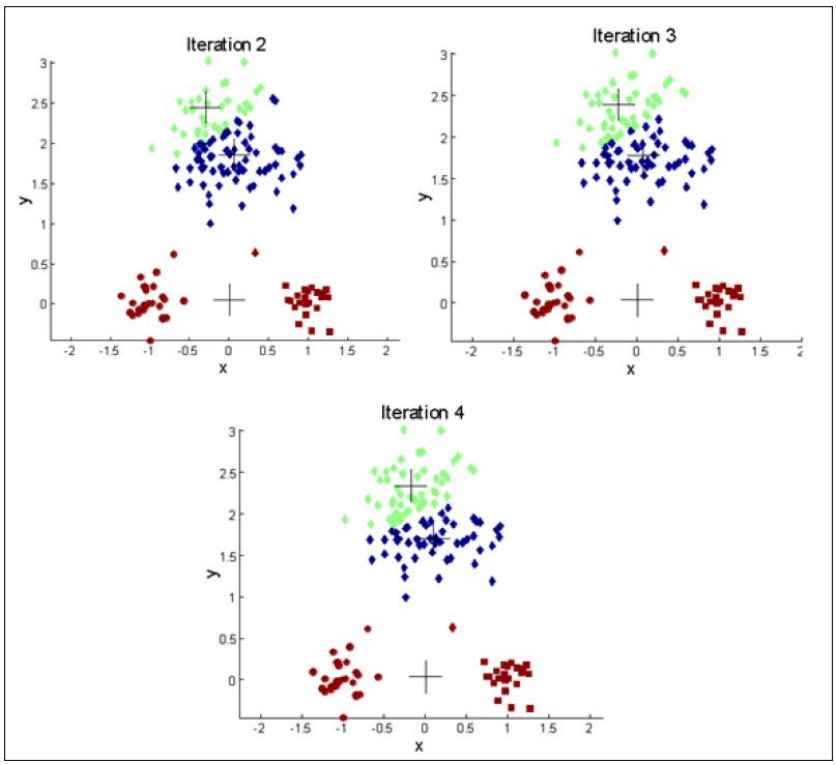


Figure 1.2 K-Means Clustering problem example (b)

K-Means also have some disadvantages. Those are:

- Difficulties in comparing the quality of the produced clusters (for example, it affects the results for different starting partitions or K values).
- Fixed number of clusters can make it difficult to predict what K should be.
- The learning algorithm demands premise determination of number of cluster centers.
- The learning technique is not consistent (i.e. with different representation of data we get different outputs.
- The use of restrictive association – If there is X highly overlapping data points then K-Means is usually not able to determine that there is X clusters.
- Arbitrarily selecting of the cluster centroids lead us to an inconsistence result.

Furthermore, we can acquire superior initial clusters centers using some other techniques and thereby that what leads us to employ the IWC algorithm later on in this research to operate better in the manner of determination the best likely clustering centers by the way of enhancing the clusters centroids [4].

However, The Particle Swarm Optimization (PSO) algorithm is a heuristic algorithm that works on optimizing a problem by iteratively trying to improve a prospect solution with regard to a given measure of quality [5-7]. The better primary clusters centroids can be gained using PSO.

We have developed a new approach that utilize PSO incorporated with IWC algorithm that generates a quick and superior clusters and also get rid of getting stuck in local optima.

The empirical results show that the novel proposed IWC-PSO algorithm can produce the best outcome in minimum number of iterations when compared to standard K-Means and PSO algorithms individually.

1.3 Methodology

This research aiming to enhance the performance of clustering big data employing Apache Spark framework as a computational engine for all parts mentioned in the earlier sections.

Apache Spark supports in-memory processing and operates better on iterative algorithms, whereas the same code is performed multiple times and the output of one iteration is the input of the next one.

Particularly, we focused on two well-studied problems, the first is how to reformulate the K-Means algorithm for solving the early mentioned problems by using IWC-PSO algorithm, and the second is how to use the distributed version of the IWC-PSO algorithm using the MapReduce framework.

As there are currently no implementations of IWC and PSO on MapReduce, the machine learning library MLib used, that executes on Spark. Chapter 3 will introduce how we are going to train and test the proposed algorithm on Spark clustering processor.

1.4 Research objectives

To design a novel hybrid algorithm, which provide a high efficient big data clustering with high accuracy and speed, imposes the robustness of clustering, optimization and time reservation algorithms [8, 9].

And based on problem stated above, the principal goals of the research are:

- Provide a high accuracy and speed algorithm for big data clustering.
- Deal with distributed and shared data warehouses.
- Reduce computational requirements time and complexity.
- Handle big data management, retrieval and analysis in a tolerable time.
- Design solutions to understand the big amount of data.
- Determine the relevance within large datasets and how to discover useful information from the relevant data.

1.5 Scope of work

In this research, we introduce a clustering algorithm which to executed on MapReduce framework, the most prevailing programming engine for processing large amount of data. The proposed algorithm uses (IWC) incorporated with Particle Swarm Optimization Algorithm (PSO) so-called (IWC-PSO) [10] to decrease the data size and search area and then implement it on the MapReduce framework to find the final clusters.

We then execute it on Apache Spark clustering environment, to measure the performance awarded by parallelizing the algorithm that analyses data distributed on multiple machines.

1.6 Main contributions

In order to have the IWC effective and efficient in data analysis, this research adopts a novel technique that is called IWC-PSO that incorporates IWC with PSO on the basis of the MapReduce paradigm. Also, it aiming at enhancing the overall search capability of the IWC by a way of including the PSO algorithm to refine its primary centres, and enhancing the performance acquired when dealing with large-scale of data by adjusting IWC to be execute from serial processing to parallel processing with MapReduce paradigm.

By applying the proposed hybrid algorithm we can achieve a number of features and advantages as follow:

- Ability to solve the massive computational requirements, and provide efficient, scalable, and parallel solution for classifying the huge amount of data into correlative subsets namely informative clusters.
- Leveraging an optimization algorithm (Particle Swarm Optimization Technique) in order to decrease the time consumption and obtain an optimal accuracy and performance and overcome the local minimum convergence problem.
- Utilize an efficient distribution and parallelization framework Apache/Spark to manipulate the big data imposing the MapReduce concept.
- Reduce computational requirements time and complexity.
- Handle big data management, retrieval and analysis in a tolerable time.
- Takes benefit of the outstanding global search ability of PSO and elegant search ability of IWC to improve the performance of the clustering process.
- Get rid of different allocations of centroids for multiple runs.
- Expedite centroids exploration and determination by reducing the overall searching area.
- Provide a high accuracy and speed algorithm for big data clustering.

1.7 Thesis structure

The rest of the thesis is structured as follows:

Chapter 2, shows the related works in this research field.

Chapter 3, provides the background information and state of arts required for the understanding of the research.

Chapter 4, explains the implementation details behind the novel hybrid algorithm proposed in this research. It describes how the algorithm works and how were we combined the two algorithms IWC and PSO to improve K-Means performance for big data processing.

Chapter 5, shows and discusses the experimental results.

Chapter 6, provides conclusion and interesting future work that could be done in order to further improve the algorithm.

Chapter 2
Related Works

Chapter 2
Related Works

Several researches on data clustering have been presented in the literature. Some researchers have presented new clustering techniques. While others have enhanced the existing algorithms by overcoming the drawbacks and disadvantages, whereas manage of them have held a comparative study between various clustering algorithms and attempting recommend new combinations.

However, the basic K-Means technique is amended and scaled up in various manners. Many of the amendments tackle extra exploration comprising the minimum cluster size and combining and dividing clusters.

With having background information that IWC is ineffective in processing large-scale datasets, the concept of parallelizing the processing is proposed into the algorithm as a modification. J. Zhang and G. Q. Wu Suggests in [11] a new K-Means based on parallel paradigm and attributed to MPI (Message Passing Interface) known as MKeans which enhances the overall cost and performing well on massive datasets. However, R. Farivar, D. Rebolledo et al. In [12] introduced a competent design of parallel K-Means algorithm using GPU (Graphics Processing Units) has been proposed and it revealed that this technique is reasonable and with low cost. On the other hand, S. Kantabutra and A. L. Couch proposes in [13] a parallel K-Means with no-explorations approach relying on NOWs (network of workstations) which utilizes a master-slave SPMD (single program multiple data) concept. In this way, the time required has been decreased dramatically, also the memory requirement limitations on a singular device is then eliminated. In addition to that, [14] introduced a Master/Slave programming approach and data parallelization scheme. Dynamic load balancing has been used in order to enhance the parallelization of K-Means algorithm. Also, W. Z. Zhao, H. F. Ma, and Q. He et al. in [44] uses MapReduce [15, 16] paradigm to elaborate K-Means to manipulate big data regardless the machines specifications. Interestingly, it shows that the algorithm is efficient and scalable.

Furthermore, Chunne employed a PSO algorithm to develop twitter data cluster algorithm based on Hadoop platform [17]. The twitter data has been manipulated

through tokenizing, originating and eliminating stop-words from all tweets. The database is figured out as a group of separated vectors of tweets, in which a single vector is a set of the weights of all included words, whereas the weight is then calculated based on the count of occurrences of each single word. Further, they maintain PSO by considering each particle in the swarm as a group of cluster locations looking for optimal solution (positions) using a parallel PSO method running in several machines, depending the tweets in the word discourse. It has observed that when using K-Means to identify the preliminary clusters, the positions converged faster.

Mariam El-Tarabily et al. [18] proposed a hybrid Particle Swarm Optimization which known as Subtractive + (PSO) clustering algorithm, that performs faster clustering. They have examined the combination of Subtractive + (PSO) clustering algorithm as well as the standalone Subtractive and PSO clustering algorithms on three various datasets for comparison purpose. From the results, it is demonstrated that the proposed clustering algorithm offers better clustering outputs than the other algorithms.

Sandeep Rana et al. [19] introduced a novel Hybrid sequential clustering algorithm. They used PSO incorporated with K-Means algorithm consequently for data clustering. This method was designed to conquer the drawbacks of both algorithms as well as enhances clustering and avoids being dormant. Four different datasets have been tested to obtain comparative outputs. For comparison purpose, various algorithms such as K-Means, PSO, Hybrid K-Means PSO, and Hybrid K-Means + Genetic Algorithm were studied. The proposed algorithm generates more accurate and superior clustering outcomes.

Bara'a Attea et al. [20] demonstrated that the performance of clustering methods degrades with more intersections among clusters in a dataset. These facts have inspired to develop a fuzzy multi-objective particle swarm optimization framework (FMOPSO) in a creative style for data clustering, which is capable to present more effective results than classical clustering algorithms. To discover the superiority of the proposed algorithm, many empirical experiments have been carried out on a sort of categorical and numerical real datasets.

K. Premalatha et al. [21] proposed a hybrid algorithm of PSO with Genetic Algorithm (GA). The proposed system figured out a better result without trapping in local optimum,

and to attain faster convergence rate. That is because when the PSO stagnation occurs, GA modifies the particle locations even though the solution is worse. This makes PSO-GA more flexible and stable. Unlike classical PSO, the PSO-GA is more fruitful in giving better quality results with sensible computational cost. Empirical results are studied with different benchmark criteria and results show that the proposed algorithm surpass the standard PSO.

Following is a concise description about some of the earlier researches on clustering algorithms:

Bo Thiesson et al. [22] compared four clustering approaches EM, SOM, K-Means and hierarchical on different test datasets and adopted some conclusion on the accuracy, quality and performance of the four methods.

Osama Abu Abbas [23] classified the clustering algorithms into four types, demonstrated their advantages and disadvantages and compared them on different criteria.

Preeti Baser et al. [24] described the drawbacks of the K-Means method and the different solutions used to overcome them.

Kehar Singh et al. [25] has introduced a detailed demonstration about different clustering algorithms and also discussed the dimensionality reduction, scalability and other general algorithmic attribute.

Gaikwad et al. [26] Discovers one of the main drawbacks of the K-Means method as it is inefficient to adjust new data incoming into a data stream.

On the other hand, many researches have been produced in order to specify the best preliminary clusters to be acquired, Ahmad and Khan [27] have proposed an algorithm known as Cluster Center Initialization Algorithm (CCIA) to identify primary cluster centroids. CCIA method is built upon the empirical fact which much like data points construct the essential clusters, where their memberships stay stable. Accordingly these alike data points help in determining preliminary cluster centroids.

Additionally, CCIA relies on the inspection which a single feature may give a knowledge in determining primary cluster centroids. CCIA concludes that every single feature of the dataset is ordinarily allocated. For K clusters the curve is split into K equal-area partitions. After that, calculate the midpoint of each interval. The features corresponding to these midpoints are then calculated utilizing the standard deviation and the mean of the each feature. Therefore, the K-Means will use this midpoints to employ as a seed point regarding this feature. This technique produces a set of *m* cluster labels. Repetition this process for all the features will produce k' series that associated with k' clusters. And check if k' is equal to k, then centroids of those k' clusters supposed to be treated as the preliminary cluster centroids for the K-Means method. While if the number of intended clusters k is less than k', alike clusters are then combined to have k-clusters, where centroids of these k-clusters going to become the primary centroids for the K-Means method.

As we know, the above methods are the recent researches in the K-Means modifications proposals. Even though those methods demonstrate adequate clustering efficiency, they come down with some limitations, like:

- It requires a costly execution time to determine preliminary centroids, "the time complexity then becomes exponential since the process of selection is repeated frequently" [27].
- Number of clusters must be identified and fixed in prior. In many cases in the real life applications number of clusters is not determine or recognized and a technique is then required to identify k prior to clustering process.

J.F. Lu et. al. [28] introduced a hierarchical initialization algorithm for determining the primary centers of K-Means. This method comprise of four essential phases; they are pre-processing, bottom–up processing, top–down processing and post-processing. The objective of pre-processing is to convert the data distribution to a shape that is intended by this method. The bottom–up and top–down processes are the essential phases of the entire method which performing sampling, reduction and clustering eventually. The post-processing then invalidates the pre-processing procedure by transforming the

inverse of coordinates to attain the primary cluster centroids in the dataset. In sampling process the date generated from the pre-processing procedure is then gradually by continually performing a sampling task, and stops when the sampled data size is the minimal number that is greater than or equal to 20 * the number of clusters k. "At the phase that the sampling process ends, iterative clustering is then performed so as to get the cluster centroids. And for the option of primary centers, data is ordered by its weight values, and then the first k greatest instances are chosen as the primary cluster centroids for that iteration" [28].

Chapter 3
Background

Chapter 3
Background

3.1 Big Data

"Big data synonym stands for to datasets whose size is over than the capacity of traditional database softwares to handle, manage, manipulate and store. This description is globally subjective and involves a flexible definition of how large a data supposed to be, in order to be intended big data" [29].

One of definitions of big data is declared in IDC's (International Data Corporation) - The Digital Universe's study:

"IDC describes the Big Data technologies as a new era of technologies and architectures designed to pull out value economically from massive volumes of a diversity of data by offering high-velocity capture, discovery and analysis. Therefore, there are three main characteristics of Big Data: the data itself, the data analytics, and the results of the analytics presentation" [30].

When we are unable to handle our large scale data via our traditional data handling system, we call those data to big data for particular field. Bezdek and Hathaway illustrated a categorization of data sizes which is depicted in table 3.1 [31]. Mainly, big data was identified by some characteristics. Those are known as 4 V's characteristics of Big Data. Those are volume, velocity, variety and veracity.

- **Volume**: An example is the unstructured data streaming generated from social media and it urgent question about how to ascertain the relevance within massive data volumes and how to analyse the relevant data to generate worthy information and outcomes.
- **Velocity**: Data is growing surprisingly and it has to be handled with in rational time. Responding quickly to data velocity is one of the main obstacles and challenges in big data.

- **Variety**: Manage, merge and govern data that comes from different sources with different characteristics is another challenging problem, for instance: unstructured data, social data, email, video, audio and etc.
- **Veracity**: Big Data Veracity refers to the biases, noise and abnormality in data. Is the data that is being stored, and mined meaningful to the problem being analysed.

Table 3.1. Bezdek and Hathaway categorization for big data

			Big data		
Bytes	10^6	10^8	10^{10}	10^{12}	$10^{>12}$
Size	Medium	Large	Huge	Monster	Very large

3.2 MapReduce

MapReduce is an extensively used parallel programming approach. There are no difficulties to develop parallel programs in order to deal with data-intensive applications on commodity machines clusters. The MapReduce model represents a simplified way of parallelization for programs, which are written with the MapReduce paradigm. "In MapReduce programming paradigm, the basic unit of information is a (key; value) pair where each key and each value are binary strings" [32]. The input to the MapReduce algorithm is a set of (key; value) pairs. Operations on a set of pairs occur in three stages: the map stage, the shuffle stage, and the reduce stage.

Map Stage: In the map stage, the mapper takes a single (key; value) pair as input and generates any number of new (key; value) pairs as output. It is necessary that the map process function works on one pair at a time. This enables for effortless parallelization as various inputs for the map can be executed by different devices. And it is applied to every pair in the input dataset in parallel. This generates a list of pairs for each call. Consequently, the MapReduce paradigm collects all same pairs with the similar key from all lists, and collect them together, generating one group for each key resulted [32]. Table 3.2 describes the overview of the Map function.

Table 3.2. Map Function Input Output

Map	**Input**	**Output**
	`<k1, v1>`	`List (<k2, v2>)`

Shuffle Stage: In the Shuffle stage, the primary system that performs MapReduce sends all of the values that are correlated with a unique key to the same node. This happens automatically, and is consistent to the developer.

Reduce Stage:" In the reduce stage, the reducer takes all of the values associated with a single unique key k, and yield**s** a multi-set of (key; value) pairs with the same key k" [32]. This highlights one of the sequential prospects of the MapReduce process; all of the mappers needed to be finished before the reducers begin. Whereas the reducer has the accessibility to all the values with the same key, it can carry out sequential computations on these values. In the reduce phase, the parallelism is utilized by noticing that reducers working on different keys can be executed synchronously [33, 34]. Generally, a program in the MapReduce framework usually consists of many rounds of different mappers and reducers performed one after another successively. Table 3.3 represents the concept of the reduce function.

Table 3.3 Reduce Function Input Output

Reduce	**Input**	**Output**
	<k2, list(v2)>	List (<k3, v3>)

Each Reduce process basically outputs either one value v3 or an empty return, however, one call is permitted to return more than one value. The returns of all calls are then grouped as the desired result list. Thereby, the MapReduce framework has to transform a list of (key; value) pairs into a list of values.

Programs written in this functional arrangement are automatically parallelized and executed on a large cluster of commodity machines. "The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication" [33, 341]. The map reduce strategy is show in Figure 3.1.

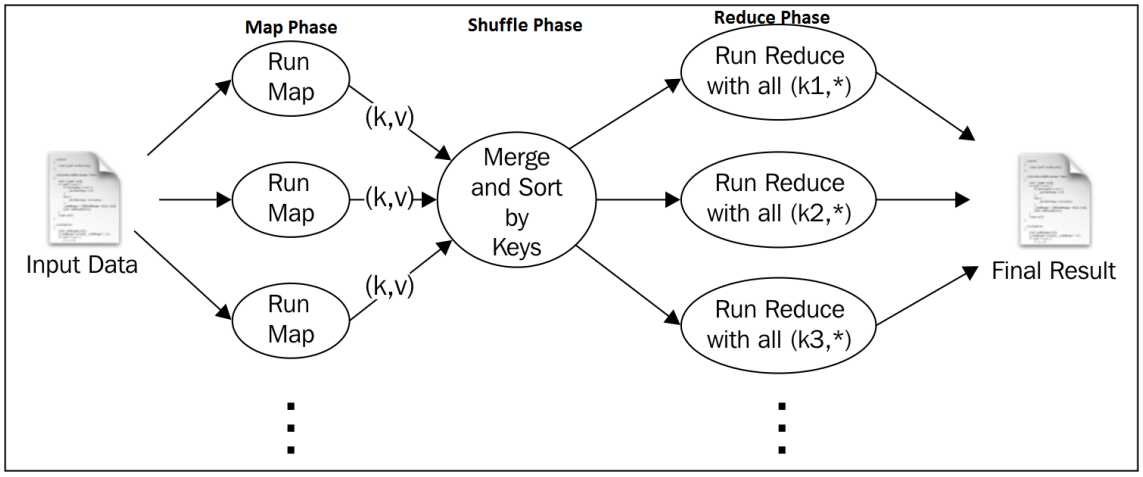


Figure 3.1 Map reduce programming model

To illustrate the concept of MapReduce with the help of an example, Figure 3.2 shows a simple word count example, given an input file, it is required to compute the frequency of each word in the file using the MapReduce strategy.

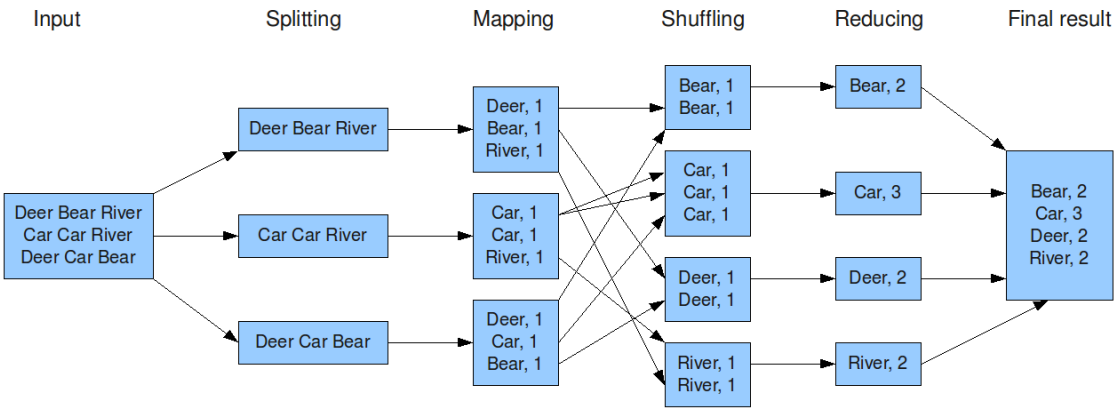


Figure 3.2. Map reduce flow of simple word count program

MapReduce Advantages:

Even though, not all algorithms can be effectively formulated in terms of map and reduce functions, MapReduce provides many advantages over other parallel processing models.

In such models, a program comprises of only a one map function and a one reduce function. Everything else is typical to all other programs [35]. The infrastructure granted by a MapReduce implementation handles all of the communication details, fault tolerance, load balancing, job startup, file distribution and resource allocation.

This run-time paradigm is implemented and asserted by experts of parallel programming model and they guarantee that this approach is powerful, while those who impalement mappers and reducers programs are concentrate on the problem without distract themselves from implementation details.

A MapReduce method identifies task segregation at run-time and associates tasks to calculate nodes as long as processors come to be obtainable. If some nodes are faster than others, they will be disposed more tasks, and if a node fails or aborts, the system reassigns the interrupted task automatically.

3.3 K-Means Clustering

K-Means is a method of analysing data, which considered to determine k groups from n data points taken as an input. The division occurs based on the data point engaging itself into the closest mean point.

The basic workflow of K-Means method is described as follows:

1. Select K data points randomly to represent the desired centroids.
2. Originate a new group by assigning each data point to its nearest centroid (assignment happens based on the closest mean).
3. Calculate new cluster centers (using the mean formulae for multidimensional data-points) Figure 3.3 shows the results of K clusters data points.
4. Repeat steps 2 and 3 till cluster membership stabilizes, by either number of iterations specified by the user exceeded, or the dimensions of centroid does not change.

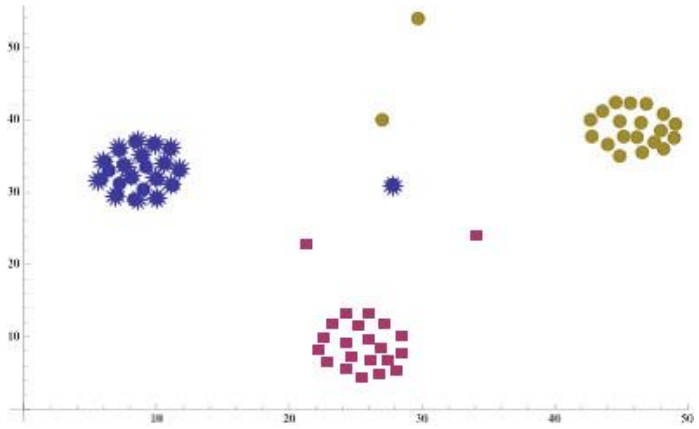


Figure 3.3 K-Means representation applied on a data set of 65 data points (n=65) and number of clusters to be 3 (k=3)

K-Means method detects a cluster where the sum of squared error (distances) between the calculated mean of cluster and the data points in the cluster is minimized.

Let μ_k be the mean of cluster c_k. The sum of the squared error between μ_k and the points in cluster c_k is defined as:

$$J(c_k) = \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \qquad (1)$$

K-Means aims at minimizing the overall squared error for all K clusters,

$$J(c) = \sum_{k=1}^{K} \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \qquad (2)$$

As we can observe, we would require getting all the data-points in memory to start their comparison with the centroids of the cluster. This is plausible only in conditions where the data sets are not very large (very large depends on the size of your RAM, processing power of your CPU and the time within which the answer is expected back).

The following algorithm describes the k-Means workflow:

Algorithm: *K-M*eans. The *K-M*eans aims at splitting/clustering dataset, where each cluster's center is represented by the mean value of the data points in the cluster.

Input:

k: number of required partitions/clusters,
D: a dataset *n* data points.

Output:

A set of *k* clusters.

Steps:

(1) Choose k data points from D randomly as the primary cluster centroids;
(2) **repeat**
(3) Update each data point association to the cluster which the object is the most similar, based on the value of the mean of the data points in the cluster;
(4) update each cluster mean, by calculating the mean value of the data points for each cluster **until** no change;

Figure 3.4 K-Means clustering algorithm Pseudo code

This traditional approach of K-Means does not scale well for many reasons:

- Sensitivity to initial conditions of the clusters.
- The complexity is considerably high – k * n * O (distance metric) * num (iterations).
- We need a solution which can scale with very large datasets.

3.3.1 K-Means Clustering Properties:

Basically, we can say, K-Means clustering characteristics are:

- Predefined K clusters.
- Each cluster should contains at least one item.
- Clusters don't overlapping due to non-hierarchical clustering approach.
- Every member of a cluster is closer to its cluster than any other cluster because closeness does not always involve the 'center' of cluster [35].

3.3.2 Advantages of K-Means:

K-Means have some advantages. Because of those advantages, K-Means is the most used clustering algorithm in last fifty years. Advantages are:

- With a large number of variables, K-Means may be computationally faster than clustering algorithms (if K is small).
- K-Means may produce specific clusters than other clustering algorithms, especially if the clusters have regular shapes.
- Fast, accurate and easier to realize and follow.
- Yields best results when dataset is distinct or well separated from each other.

3.3.3 Disadvantages of K-Means:

K-Means on the other hand also have some disadvantages. Those are:

- Clusters generated are not consistent for all runs.
- Fixed number of clusters makes it ambiguous to determine what K should be.
- The algorithm requires apriority determination of the number of clusters.
- Euclidean distance measurement usually unlikely weight underlying factors.
- Susceptible to converge to the local optima of the squared error function.
- Randomly choosing of the cluster center cannot lead us to the fruitful result.
- Applicable only when mean is defined i.e. fails for categorical data.
- Difficulty in handling outliers and noisy data.
- The method can't be applied on non-linear data set.
- Perform inefficiently with large scale data.

3.4 Inverse Weighted Clustering

IWC algorithm first produced by Wesam Barbakh and Colin Fyfe - solves the preliminary initialization inconsistency of K-Means method, they aggregated the data samples into a predefined number of clusters in a space. Then use the Euclidean distance to evaluate similarities. "The approach in this context is to cluster data points in somehow that the Euclidean distance between data points belonging to each single group is being minimised". Then the data points in each single group (cluster) are represented by the group center, which announced as the cluster center. Therefore, IWC method endeavours to determine the best points in the virtual space and consider them as clusters centers.

The IWC method has the following empirical approach:

$$J_I = \sum_{i=1}^{N} \sum_{k=1}^{K} \frac{1}{\|x_i - m_k\|^P} \tag{3}$$

$$m_k = \frac{\sum_{i=1}^{N} b_{ik} X_i}{\sum_{i=1}^{N} b_{ik}} \tag{4}$$

Where

$$b_{ik} = \frac{1}{\left\|x_{i-} m_k\right\|^{P+2}} \tag{5}$$

The partial derivative of J_I in regard to m_k will maximize the performance function J_I, and usually used to observe the output. Hence, the formulation of (4) will motivate m_k to the nearest data point and basically to maximize J_I to ∞,

Though, the formulation of (4) will not determine any clusters as the representatives like what [11] did, but centroid will always move to the nearest data points. However, we get a benefit by applying this function where it doesn't quit any representatives away

from a whole data space, which means that all the representatives are going to join the data.

The novelty in this method is developed to make the procedures able to determine the clusters and keeping its ability of moving the data points inside data space by modifying b_{ik} in (5) to the following:

$$b_{ik} = \frac{\|x_i - m_{k*}\|^{P+2}}{\|x_i - m_k\|^{P+2}} \qquad (6)$$

Where m_{k*} is the nearest representative to x_i.

By this modification, the algorithm has an exciting behaviour: (6) aims at maximizing J_I by motivating the representatives to the unbound data points rather than the nearest local cluster.

Therefore, (5) and (6) will not leave any representative far away from the data space although if they are originated out the space. The representatives are always moved to participate the nearest data points depending on (5) or to participate with the unbound data points using (6). Knowing that (5) doesn't determine clusters while (6) does.

(6) Remains consistent on the property of (5) for moving the representative to join data, and offers the ability of determining the clusters.

As soon as one of the below constraints achieved, the clustering execution stops:

1. A predefined maximum iterations are exceeded.
2. Negligible changes in cluster centroids are occurred.
3. No cluster membership update.

In Figure 3.5, all the representatives are allocated within a single cluster. As demonstrated in the top depict, whereas in the middle depict K-Means is failed to determine the clusters properly. Finally, in the bottom depict and based on (6), IWC determines all clusters successfully.

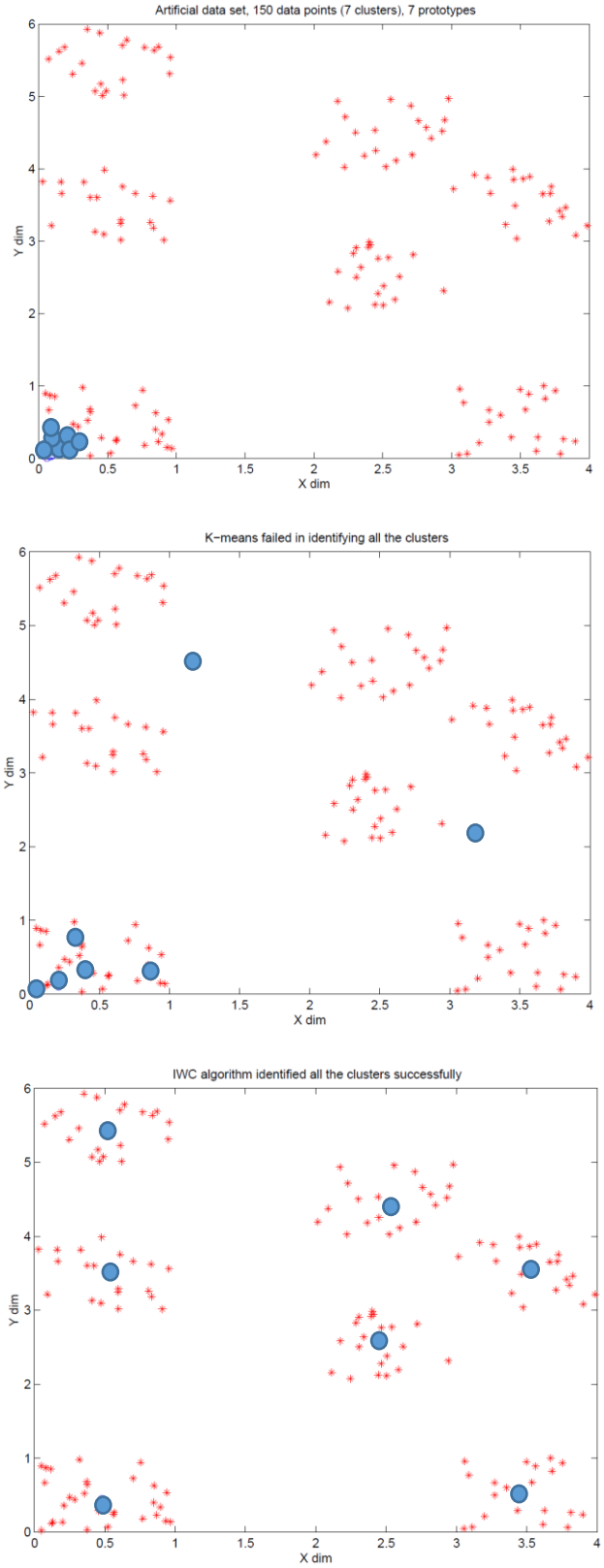


Figure 3.5: **Top**: Synthetic dataset: dataset is shown as 7 clusters of red '*'s, representatives are initialized randomly to be located within one cluster and shown as blue 'o's. **Middle**: K-Means output. **Bottom**: IWC method output.

Figure 3.6 demonstrates the output of applying IWC method to the same synthetic dataset but with messy representatives' initialization. As demonstrated in the top depict, IWC achieves good clusters determination under this bad initialization as shown in the bottom depict, while K-Means failed as shown in middle depict.

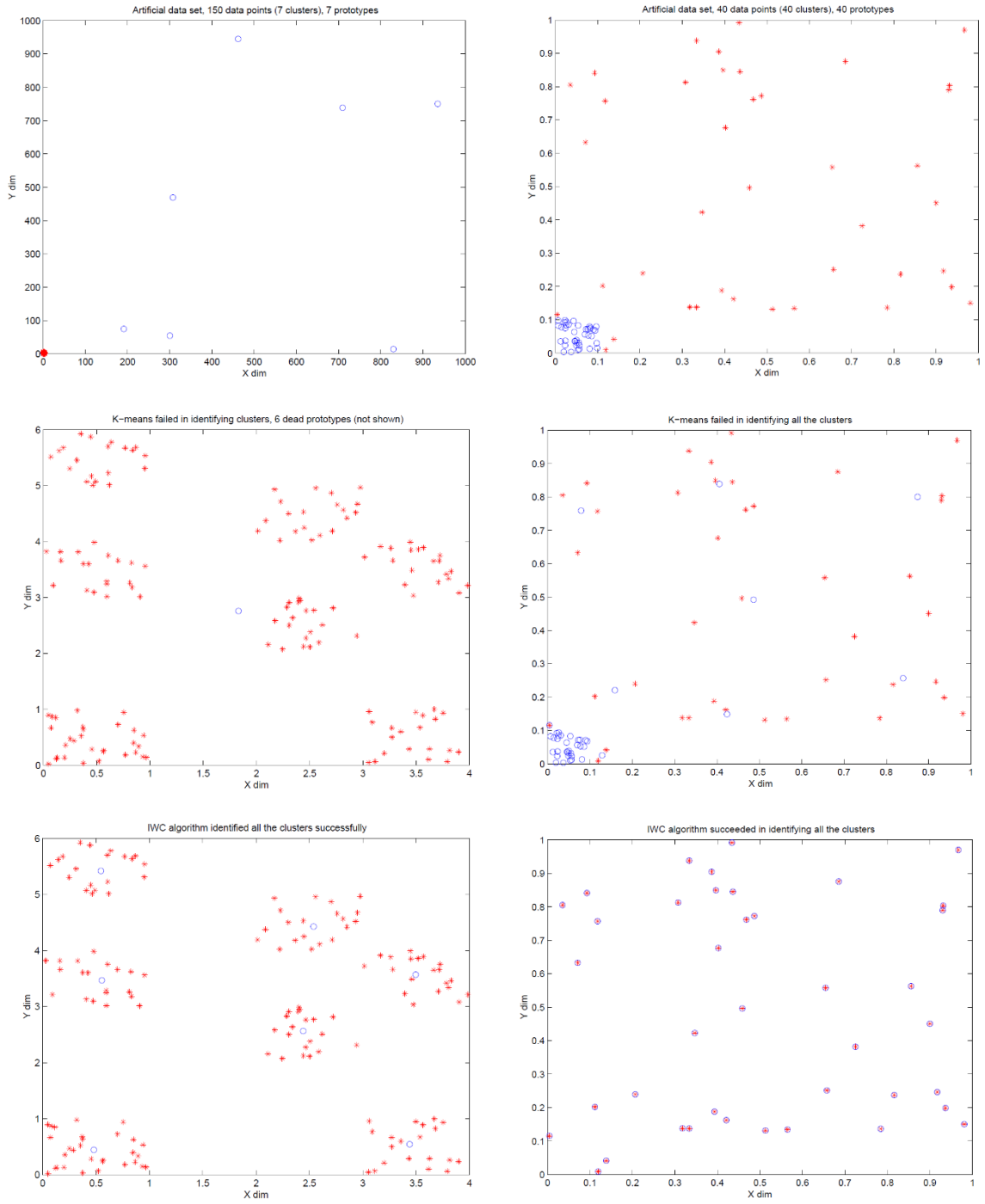


Figure 3.6: **Top**: Synthetic dataset: dataset is shown as 7 clusters of red '*'s, representatives are initialized very far from data and shown as blue 'o's. **Middle**: K-Means outputs. **Bottom**: IWC method outputs.

Generally initializing representatives far from data space is an unlikely situation to happen, however it may be to happen that all the representatives are really initialized very far from a specific cluster.

3.5 Particle Swarm Optimization (PSO)

"Particle swarm optimization (PSO) is essentially known as an optimization algorithm which represents the motion and huddling of birds" [36]. Where particles considered as the agents that express a single solution, and the swarm represents the group of particles that indicate the expected in the solution universe of discourse. Particles start moving over the solution space by adjusting a velocity V and following up its best previous location gained currently. The location value obtained which so-called its personal best position P_b and denoted by vector $P_i = \{p_{i1}, p_{i1}, \dots, p_{in},\}$. So far, the particle's velocity and its new position is defined for each iteration, based on to the below equations:

$$v_i^{(t)} = w \times v_i^{(t-1)} + c_1 \times r_1\left(P_i - X_i^{(t-1)}\right) + c_2 \times r_2\left(G - X_i^{(t-1)}\right) \qquad (7)$$

$$X_i^{(t)} = X_i^{(t-1)} + V_i^{(t)} \qquad (8)$$

Where, w is called the inertia weight that maintains the particle's previous obtained velocity and its impact on the current velocity. In [36] plenty of inertial weight w selection criteria have been introduced. Commonly, at the initial phases of PSO method, the inertial weight w should decrease expeditiously, as soon as the swarm start converging around the optimal solution, then the inertial weight must decrease slowly. Let r_1 and r_2 are two irregular variables allocated randomly in range$[0,1]$ and let c_1 and c_2 are two positive parameters defined as "acceleration coefficients" which control the maximum step size between consecutive iterations.

Regarding equation (7) the particle's velocity at each iteration is computed based on three terms: "the velocity of the particle at previous iteration, the distance of particle from it's the best previous position and the distance from the best position of the entire population" [12]. And according to the particle's velocity it flies to a new position according to Equation (8).

Another best solution of global best position indicated by vector $G = \{g_1, g_2, \dots, g_n,\}$ that represents the best fitness value obtained by any particles. Generally, then evaluate the fitness value for each particle in the swarm using a fitness criterion. The work flow of PSO algorithm is demonstrated in Figure 3.7.

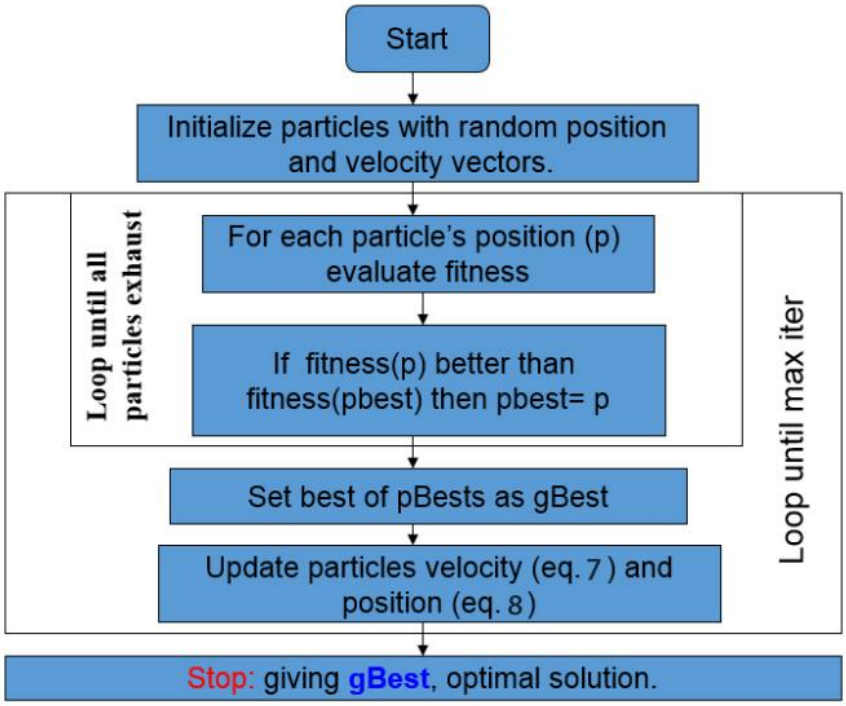


Figure 3.7. Basic flow diagram of PSO

Figure 3.8 shows the conceptual diagram depicting the workings of a population of individuals in a swarm. In the representation, the algorithm is initialized with random particles within a problem space and the particles are iteratively moving to find the optimum solution.

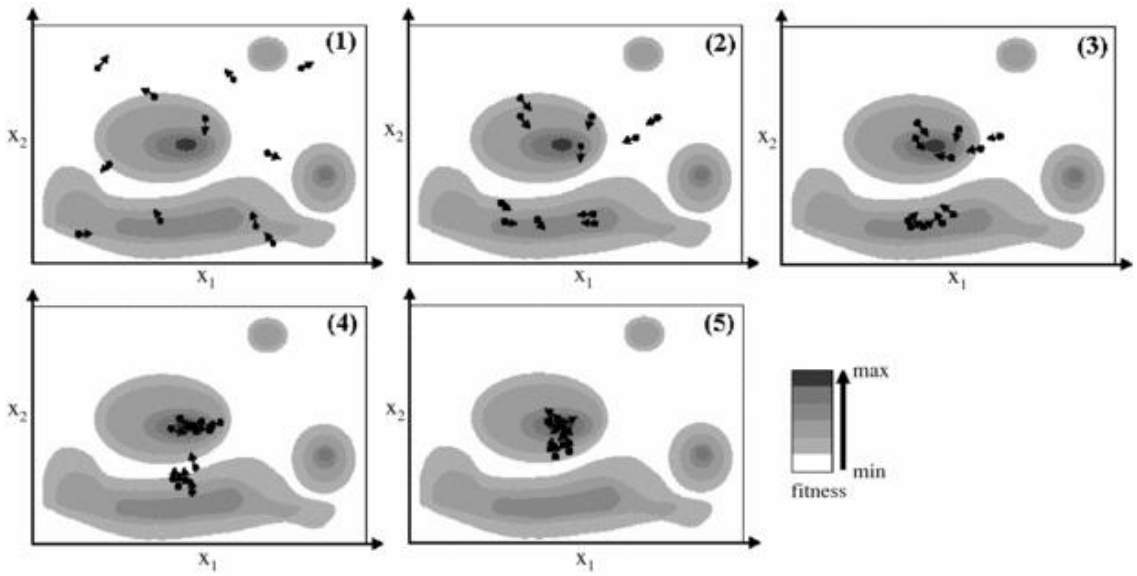


Figure 3.8: Swarm Intelligence

However, the solutions of the problem usually increases dramatically with the large dimensions and more feasible search algorithms are then required to determine all optimistic regions within a given space. The search performance of majority of methods is subjected to the earlier search experience. Regarding the limitations on the processing resources, the performance of the algorithm is affected by increasing of problem dimensions [37]. In this research we concentrate on the combination of the Particle Swarm Optimization algorithm with IWC to optimize the performance and effectiveness of the big data clustering.

Advantageous Characteristics of Particle Swarm Optimization Algorithm:

- PSO algorithm is based on swarm intelligence. It can easily be applied into wide variety of optimization problems including big data analysis.
- "The Particle Swarm Optimization algorithm has no overlapping, crossover and mutation calculation. The search can be carried out by the speed of the particle. During the development of several generations, only the most optimist particle can transmit information onto the other particles, and the speed of the updates is very fast" [38].

- PSO calculations are simple and have less parameters to adjust compared with the other optimization algorithms, where give it the bigger optimization ability and it can make completed easily.
- The fitness function can be non-differentiable (only values of the fitness function are used). The method can be applied to optimization problems of large dimensions, often producing quality solutions more rapidly than alternative methods.
- PSO is easily parallelized for concurrent processing.

3.6 Apache Spark

Spark is known as a creative cluster-computing framework which is capable to perform programs faster than Hadoop up to 40 times. Spark keeps MapReduce's linear scalability and fault tolerance; furthermore, it is extended in a few important approaches. First, instead of depending on an inflexible map-then-reduce format, "its engine can execute a more general directed acyclic graph (DAG) of operators. This means that, where MapReduce must write out intermediate results of the distributed file system, Spark sends them directly to the next step in the pipeline. In this way, an earlier version of MapReduce that began at Microsoft Research, Spark supplements its ability with a rich set of changes that allow users to express computation more compactly" [39]. It has a solid focus of developer and a streamlined API that can represent complex pipelines in just a couple lines of code.

Moreover, Spark is also suitable for iterative algorithms that require multiple steps passing over a dataset and for responsive applications that rapidly react to client queries by scanning large in memory datasets.

Spark involves a library comprising common machine learning (ML) utility, called MLlib. MLlib provides plenty of machine learning calculations like regression, classification, clustering, and filtering. Also it provides functionality such as data import and model evaluation. It also provides some lower-level ML primitives, including a generic gradient descent optimization algorithm.

Further, Spark enhances its ancestors using in-memory processing. "It's Resilient Distributed Dataset (RDD) abstraction provides developers with the ability to originate any point in a processing pipeline into memory over the cluster, denoting that future phases which need to use same data will not have to re-compute values, or fetch them from a disk" [40]. This ability tackles a number of approaches that distributed processing engines could not previously handle.

Figure 3.9 demonstrates the essential architecture when running an application on a Spark cluster. As shown in the figure, to run a Spark-based application three different parts are required (Worker node, Cluster manager, Driver program).

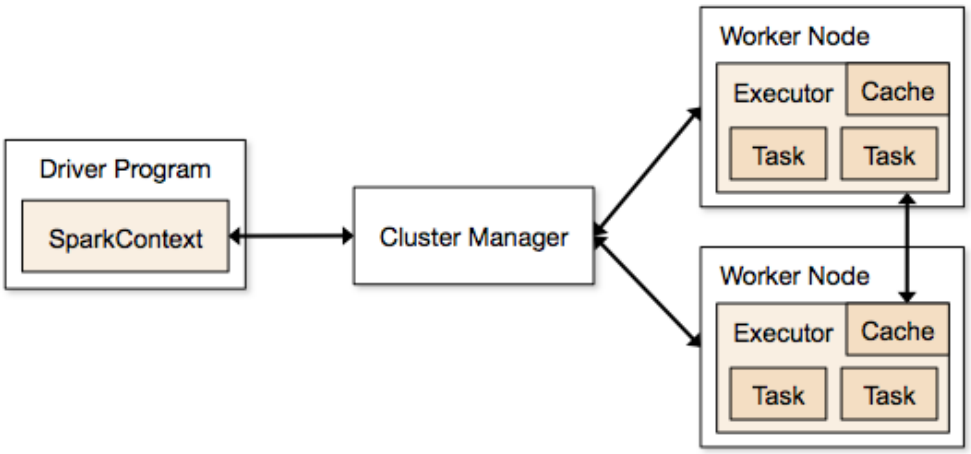


Figure 3.9. Overview of Spark's architecture

Chapter 4
Hybrid PSO-IWC for Big Data Clustering

Chapter 4
Hybrid PSO-IWC for Big Data Clustering

In this chapter, firstly we describe some basic definitions involved in the introduced procedures and then clarify the basic idea of the new approach and the proposed algorithm.

There have a lot of experiments and huge researches to improve standard K-Means. All researchers and scientists work for improve native K-Means. They worked for simplicity, save some time, try to make more accurate. They try to fix the limitations like number of clusters, way to find initial centroids and manipulating large scaled data. We get influenced by those thesis and researches. We make a decision to make some modification that can be more sufficient and faster.

Hence we considered to employ IWC (the modified algorithm over K-Means) to eliminate the inconsistency in the initial distribution of the clusters.

In more details, the introduced approach works in two consecutive phases. Phase I describes the Particle Swarm Optimization and how it going to reduce the search area and how it can find the global optimal, while Phase II is describe how to employ IWC Algorithm to find the final clusters.

We noticed that phase I produces better initial clusters selection and reduce the entire search space, that’s because the PSO is a global search method and has a strong ability in identifying the global optimistic solution.

The main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms like Genetic Algorithms.

"In PSO, individuals are referred to as particles and the population is called a swarm. A PSO algorithm maintains a swarm of particles, where each particle represents a potential solution. Each particle is given a position and a velocity. Once a particle finds a good direction to move, other particles are notified and will be able to steer toward that direction immediately" [41]. The particles roam in the space, convey their positions to

each other, and adjust their own positions and velocities based on these "better" positions.

The search behavior of a particle is thus affected by that of other particles within the swarm. Figure 4.1 depicts how particles move in a 2-dimensional search space to reach the single global solution. A star is an expected solution and circles are the particles in the swarm.

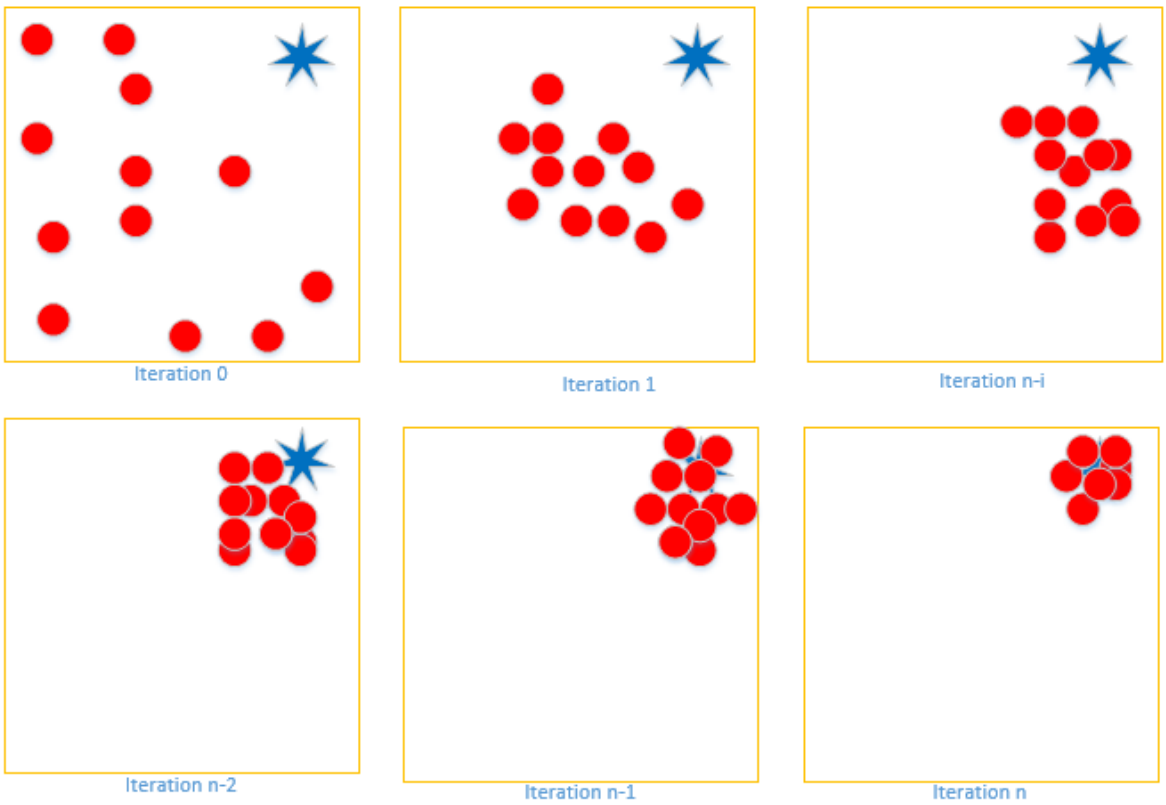


Figure 4.1. Conceptual diagram of PSO in action with a global best solution

Regarding to each particle location in the swarm, K cluster centroids in the denoted PSO algorithm will be represented as: $x_i(t) = (c_1, c_2, \ldots., c_k)$. So if we have D-dimensional samples, then the position for every particle will be $\mathrm{D}^{*}\mathrm{K}$ dimensions. Also the velocity of each particle is then going to be $\mathrm{K}^{*}\mathrm{D}$ dimensions. Therefore, once PSO arrives to the last iteration, then the current global best position $g(t)$ is the optimal solution for the primary cluster centroids.

It is well-known that larger w will affect PSO to be better in global search ability while on the contrary smaller w will make it more prone to local search. Therefore, to achieve a superior global search capability for the preliminary phase and improve local search

capability in the recent phase, in [42], w is set to be decreased linearly as the increment of the iterations. In (10), w_{max} is the maximum inertia weight, that is mostly set to be 0.85 and w_{min} is the minimum inertia weight that is set to be 0.38 as default. Hence, we limited each particle velocity to $[-v_{max}, v_{max}]$ as shown in (10) in order to keep the particle away of being trapped and preliminary converging to local minimum.

$$w = w_{max} - iter \times \frac{w_{max} - w_{min}}{iter_{max}} \qquad (10)$$

The key behind assessing the result of clustering is the fitness function construction. This representation of the fitness function shows the sum of intra-cluster distance calculations of all the clusters. In (11), c_j takes place the center of cluster G_j, and $d(x_i, c_j)$ take place the distance between one data point and the centroid of cluster G_j.

$$F(z_i) = 1 \Big/ \sum_{j=1}^{k} \sum_{\forall x_i \in G_i} d(x_i, c_j) \qquad (11)$$

Hence, the denominator J_c as demonstrated in (12) shows the aggregation of distances between all data points and the corresponding cluster centroids.

$$J_C = \sum_{j=1}^{k} \sum_{\forall x_i \in G_i} d(x_i, c_j) \qquad (12)$$

And to obtain better clustering output, the data points in one cluster should be as closed together as possible. In this research, J_c is considered as a fitness function to calculate the clustering output. When using smaller J_c then we absolutely going to obtain a better clustering result.

On the other hand, IWC is fast in converging to the local optimum, however its potency to obtain the global solutions takes too many iterations. The Phase I will output a result that will be delivered to Phase II as input which produces the final clusters using MapReduce parallelized concept. The cluster produced by this approach is much accurate, powerful, faster in comparison to standard IWC algorithm and able to deal with large-scaled data effectively.

A flow chart is demonstrated to help in reveal the general workflow of the IWC-PSO method, as shown in Figure 4.2.

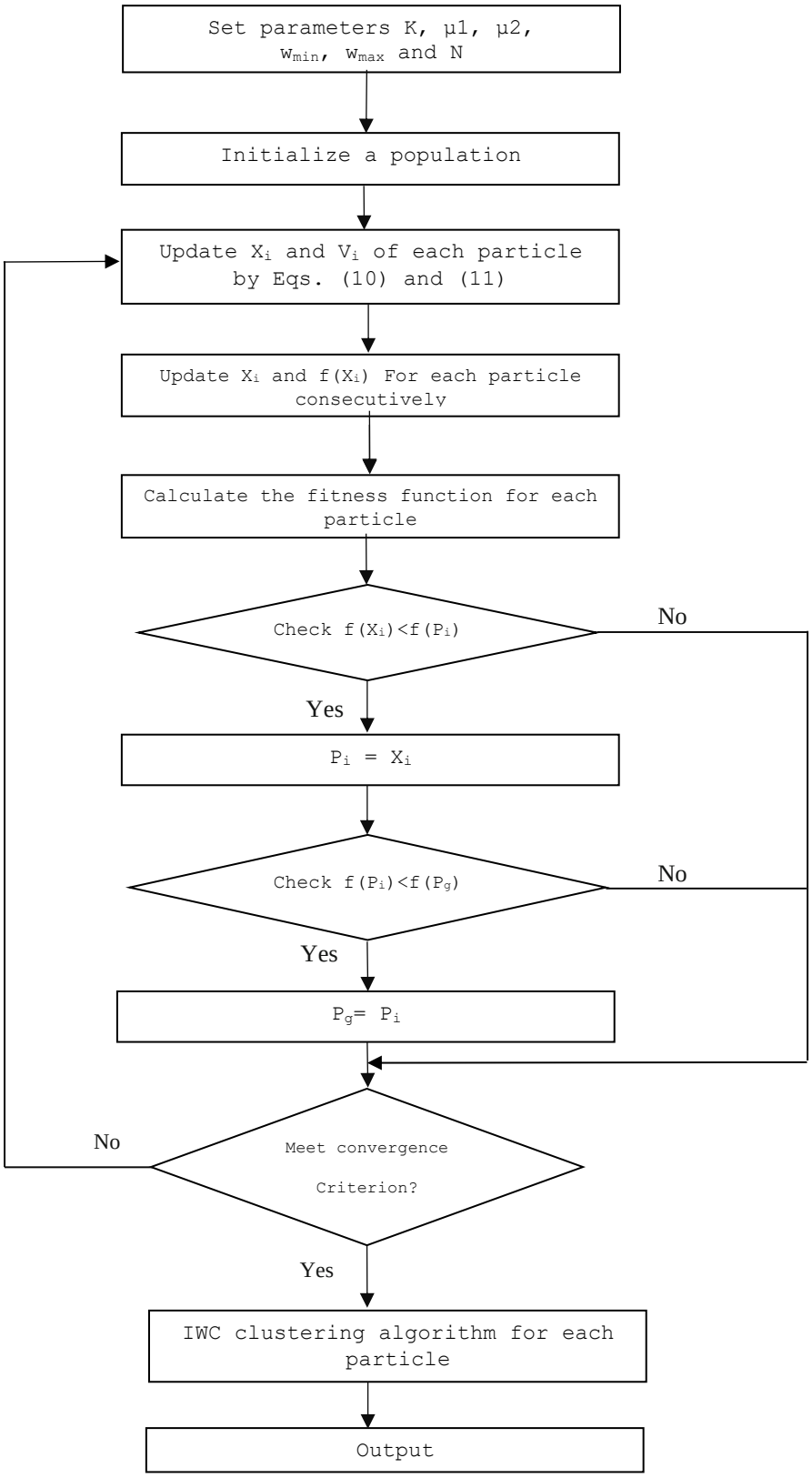


Figure 4.2 Flow chart of the IWC-PSO algorithm

Generally, the method of optimizing primary cluster centroids of IWC-PSO can be summarized as follow:

---

(1) Identify the number of clusters K and setup the parameters, including inertia weight (w_{min}, w_{max}), learning factor ($\mu 1$, $\mu 2$) and population size N.
(2) Setup a swarm population size.
(3) Setup personal best position BP_i for each particle and the swarm's global best position P_g. And for every single particle, set X_i to be P_i and set the particle's location with a minimum fitness value and considering it P_g.
(4) Update the location and velocity of each single particle regarding to the equations 7 and 8, respectively.
(5) For each single particle, adjust its location and fitness value according to equation (12).
(6) For every single particle i, compare the fitness value of its current position X_i with its previous personal best position BP_i, if $f(X_i) < f(P_i)$, then assign X_i to P_i (let $P_i = X_i$).
(7) For every single particle i, compare the fitness value of its personal best position BP_i with the swarm's previous global best position GP_g, if $f(Pi) < f(Pg)$, then assign P_i to P_g (let $P_g = P_i$).
(8) Test the convergence pattern, which is the iterations maximum number. If converged successfully, generate the clustering output, else loop to step 4.
(9) Make use of the global best location as a primary cluster centroids and execute the IWC.

Figure 4.3 IWC-PSO algorithm Pseudo code

4.1 Employing MapReduce in the algorithm

In the IWC-PSO new approach, we developed the clustering task to be an optimization problem and then we are going to acquire the best solution considering minimizing the distances between each cluster and data points associated with it. Whereas in IWC, the most costly time-consumption phase is the distances calculations between one data

point and all the centroids, so we considered to use MapReduce paradigm, where task of calculating the distance will be distributed on multiple machines.

So, if the Spark clusters have N nodes and each node is going to finish M Map tasks, then the overall computations per one single Map is `C*n*I/N*M` [43]. In such way the calculations of distance supposed to be computed in parallel in order to maintain the process time.

Where:

```
C = # clusters.
N = # cluster nodes.
I = # iterations
M = # parallel map tasks.
```

And since at the last iteration in the process the new centroids are adjusted, they are required in the succeeding iteration where it will execute sequentially.

4.2 Map Function

The Map Function include two inputs: the dataset stored in the Spark RDD as mentioned in section 3.6 and the centroids obtained from last iteration from PSO. Essentially, the dataset is partitioned. Each partition is then stored in individual line and consecutively dispatched to a Mappers as set of `<key, value>` pairs, whereas the `key` is the line number and should be unique, and `value` is the line content. And for each Mapper, calculate the distance between one data point and all the centroids and then associate the data point to the nearest cluster according to the minimum distance determination. And continually performing this operation as far as all the data points associated the current partition are manipulated and assigned. The intermediate result is a list of `<key, value>` pairs where `key` is the id of the nearest centroids and `value` is the data point.

4.3 Reduce Function

A set of intermediate `<key, value>` pairs having similar and unique key is therefore dispatched to the same Reducer. The `key` is commonly the cluster id, and a list of data points associated to that cluster is considered as a `value` a. And for each Reducer,

aggregate all the data points and accumulate those having the same key. Therefore, the new centroids have generated by calculating the mean of the aggregation obtained. Create IWC Cluster Java Class that contains the above information (id, center and members of the cluster). And the result will be <key, value> pairs, where key is the cluster id, and value is IWC generated Cluster. The output obtained will be in the form of:

<<clusterID, cluster centroid, [member$_1$; member$_2$; member$_3$; ……; member$_n$]>>
and is listed into Spark RDD and prepared to be used in the next round of MapReduce job.

4.4 The idea behind incorporating these two clustering methods is

The significant advantages of the proposed algorithm are its capability to manipulate a massive data size. The effectiveness of optimization algorithm used in this approach for minimizing the calculations and search area along with the significant reduction in the complexity of processing the big data, whereas the large datasets are processed/clustered through representative subsets obtained in prior by PSO. The following are some of advantages we got by applying the algorithm:

- Takes advantage of the global search capability of PSO and the elegant search ability of IWC to improve the efficiency and productivity of the clustering.
- Figured out the different distributions dilemma of centroids for multiple runs.
- Expedite the search task for centroids through reducing searching area.
- Dealing with multi-dimensional data.
- Provide a high accuracy and speed algorithm for big data clustering.
- Deal with distributed and shared data warehouses.
- Reduce computational requirements time and complexity.
- Handle big data management, retrieval and analysis in a tolerable time.
- Makes use of MapReduce parallel computational capability in order to make the algorithm powerful, scalable and speeded up.

Chapter 5
Experimental Results

Chapter 5
Experimental Results

This chapter discusses the results generated by running the proposed algorithm as described in Hybrid PSO-IWC for Big Data Clustering (Chapter 4). The experiments are performed to compare the time consumption and the global search ability of the following algorithms: Standard K-Means Algorithm, IWC Algorithm, Adaptive Affinity Propagation Clustering (AAPC) and Parallel K-PSO.

5.1 Environmental Setup and Data Input

We used Java language for implementing the proposed algorithm and imposed Eclipse IDE with Spark/Hadoop 2.1.3 frameworks installed. Where the implementation of IWC-PSO through Spark framework is "over 200% faster than the MATLAB implementation" [58], obviously indicating that efficiency and robustness are gained by calculating the fitness value for every particle in the swarm in parallel. And because of the major setting up time needed by Spark is to acquire and allocate the data throughout RDD's, this is the reason that the overall complexity for both implementations are more similar than the per-iteration complexity. As soon as this task is done, we can take benefits of the performance of parallelization.

Experiments were conducted on distributed topology which composes of 1 master machine node (consisting of operating system windows 10 on Intel® Core™ i7-5700HQ CPU @ 2.70 GHz, 16.0 GB RAM) and 4 slaves/data-nodes (consisting of operating system windows 10 on Intel® Core™ i7-5700HQ CPU @ 2.70 GHz, 16.0 GB RAM).

The algorithm is applied on the following datasets:

- **US Public Assistance for Women and Children (WIC) dataset (~2,778,206 nodes)**:

The dataset focuses on public assistance in the United States with initial coverage of the WIC Program. The program is formally known as the Special Supplemental Nutrition Program for Women, Infants, and Children (WIC). The program allocates Federal and

State funds to help low-income women and children up to age five who are at nutritional risk. Funds are used to provide supplemental foods, baby formula, health care, and nutrition education.

- **Google+ social circles (1,076,14 nodes):** which includes data collected from Google+. This data was collected from users who had manually shared their circles using the 'share circle' feature. The dataset includes node features (profiles), circles, and ego networks.
- **Wikipedia navigation paths (3,348,63 nodes):** which includes human navigation paths on Wikipedia, collected through the human-computation game Wikipedia. In Wikipedia, users are asked to navigate from a given source to a given target article, by only clicking Wikipedia links. A condensed version of Wikipedia (4,604 articles) is used [44].
- **Amazon product co-purchasing network (7,781,990 reviews):** This data has been elicited by crawling Amazon website. It is based on Customers Who Bought This Item Also Bought feature of the Amazon website.

Also we used several artificial dataset to testing the algorithm 5000 to 5,000,000 randomly generated data points inputted for testing.

5.2 Algorithm Testing

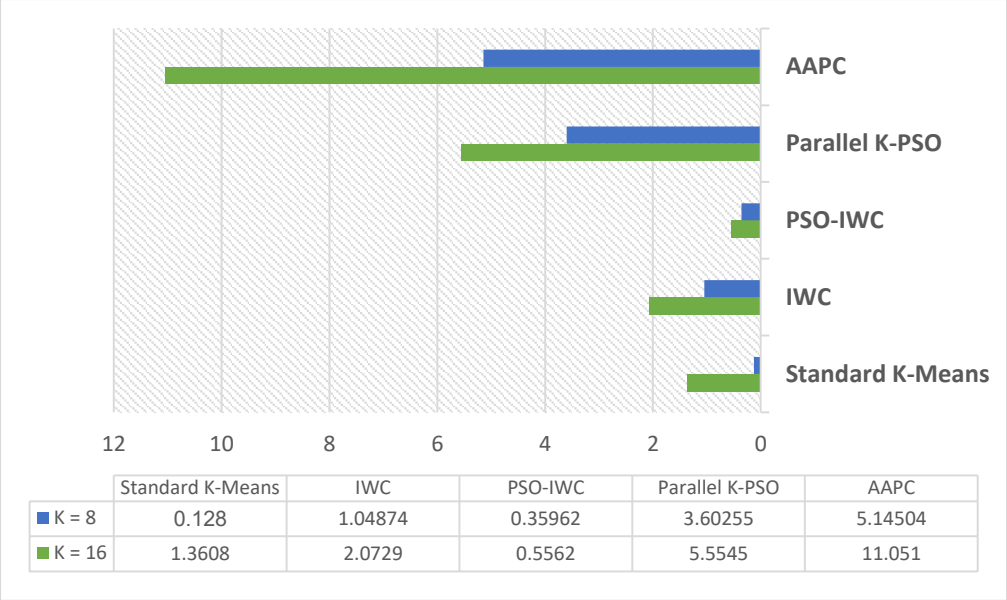
At first we tested all algorithms with 5000 data-points dataset. We assign number of clusters for those algorithms. And testify the effectiveness and scalability of speed-up for each fixed dataset as shown in Table 5.1, and then change the number of clusters K used in the tests.

Table 5.1: Result for 5000 data points with different value of K.

Algorithm	K	Cost		K	Cost	
		Time: (ms)	SSE		Time: (ms)	SSE
Standard K-Means	8	0.12800	8.5626E+05	16	1.3608	9.3330E+03
IWC	8	1.04874	5.4441E+04	16	2.0729	4.6469E+03
PSO-IWC	8	0.35962	1.488 E+04	16	0.5562	6.08224+03
Parallel K-PSO	8	3.60255	1.1024e+06	16	5.5545	1.1023E+06
AAPC	8	5.14504	8.156 E+06	16	11.051	8.25138+04

Then, number of extensive experiments were conducted to evaluate and judge the efficiency and performance of speed-up of the proposed methods whereas testing outcomes help us to find right decision on choosing algorithm.

The results presented in Figure 5.1 show that as the size of the cluster increases, the faster the IWC-PSO algorithm can execute.



	Standard K-Means	IWC	PSO-IWC	Parallel K-PSO	AAPC
K = 8	0.128	1.04874	0.35962	3.60255	5.14504
K = 16	1.3608	2.0729	0.5562	5.5545	11.051

Figure 5.1: Process time comparison of the algorithms with k =8, 16

In addition, we used different number of mappers and tried to address a sustainable number for all methods, but eventually decided to make the process of choosing number of mappers and load balancing is enclosed to SPARK to determine a correct number of mappers dynamically for each phase.

Figure 5.2 shows the results for the optimization of Wikipedia datasets with different number of mappers.

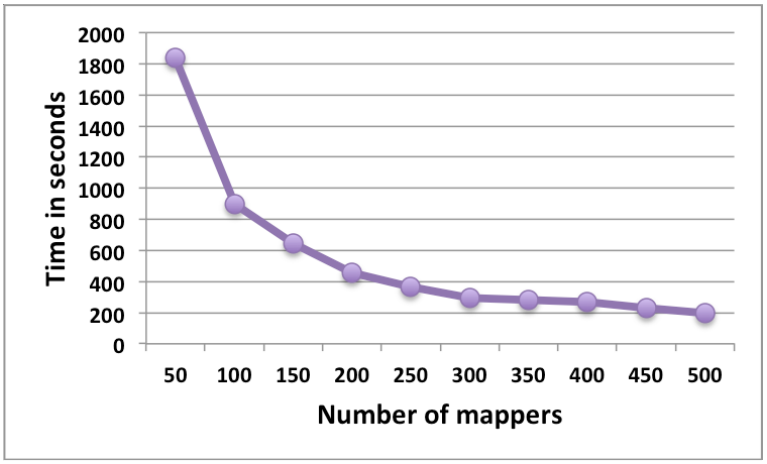


Figure 5.2: Time versus number of mappers

That is, the comparison of the above algorithms shows that the proposed algorithm performance is surprisingly much better on larger data set. Now observe Table 5.4 for larger dataset.

Table 5.2: Clustering cost (in terms of time consumed and sum of square errors SSE) for different data sets.

Algorithm	WIC		Google+		Wikipedia		Amazon		5,000 data points		10,000 data points		50,000 data points		100,000 data points	
	Cost		Cost		Cost		Cost		Cost		Cost		Cost		Cost	
	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE	Time (ms)	SSE
Standard K-Means	4.2082	12910	1.0932	31270	5.0455	34280	6.8152	45330	1.0177	28550	1.9192	28480	1.056	34640	3.4373	41350
IWC	3.3632	18100	1.6035	36960	4.1441	46340	4.2216	31230	1.0516	26750	1. 071	48950	1.903	31850	2.9033	36050
PSO-IWC	1.7524	12411	1.0287	28110	1.3029	27940	2.3672	29050	0.7287	21401	0.3263	10145	0.614	20940	2.2746	30530
Parallel K-PSO	8.036	28381	11.248	41390	7.7022	34220	10.275	58660	3.007	61050	3.3381	11020	6.911	45910	9.469	62480
AAPC	7.0352	63450	10.624	67190	6.0257	37320	12.265	42160	12.2	51750	5.045	55190	4.454	24510	8.364	42150

In the empirical experiments, all methods have been tested for 60 iterations. And as we mentioned earlier that "The Spark implementation of IWC-PSO is over 200% faster than the MATLAB implementation", which certainly showing the efficiency gained by calculating fitness value for each particle in parallel.

By observing Table 5.4, we can say, for big amount of data, the proposed algorithm performance is much better than other suggested methods.

And it's good to mention that IWC-PSO started to converge after 3 iterations in most datasets, despite we configured it to run up to 60 iterations.

Also, considered to use the SSE criteria to help judging the algorithms test results, and obviously we can see that the overall cost for every method in term of SSE and consumption time shows that the proposed algorithm is a superior one when dealing with large-scale of data.

Further, the execution and evaluation of the algorithm provides the optimal squared error results as shown in Figure 5.3.

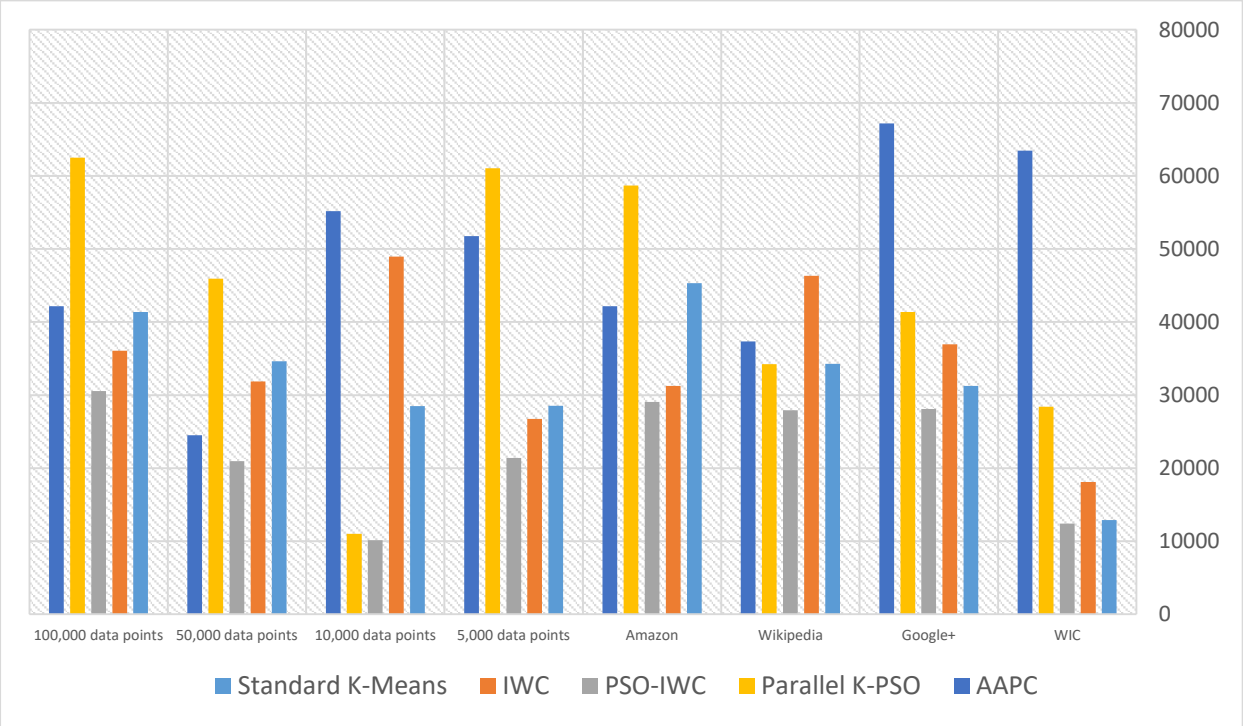


Figure 5.3: Comparing the algorithms results based on SSE

5.3 Performance Evaluation

In Figure 5.4 We can see the comparison results between the performance of the proposed approach and the further suggested algorithms in term of execution time.

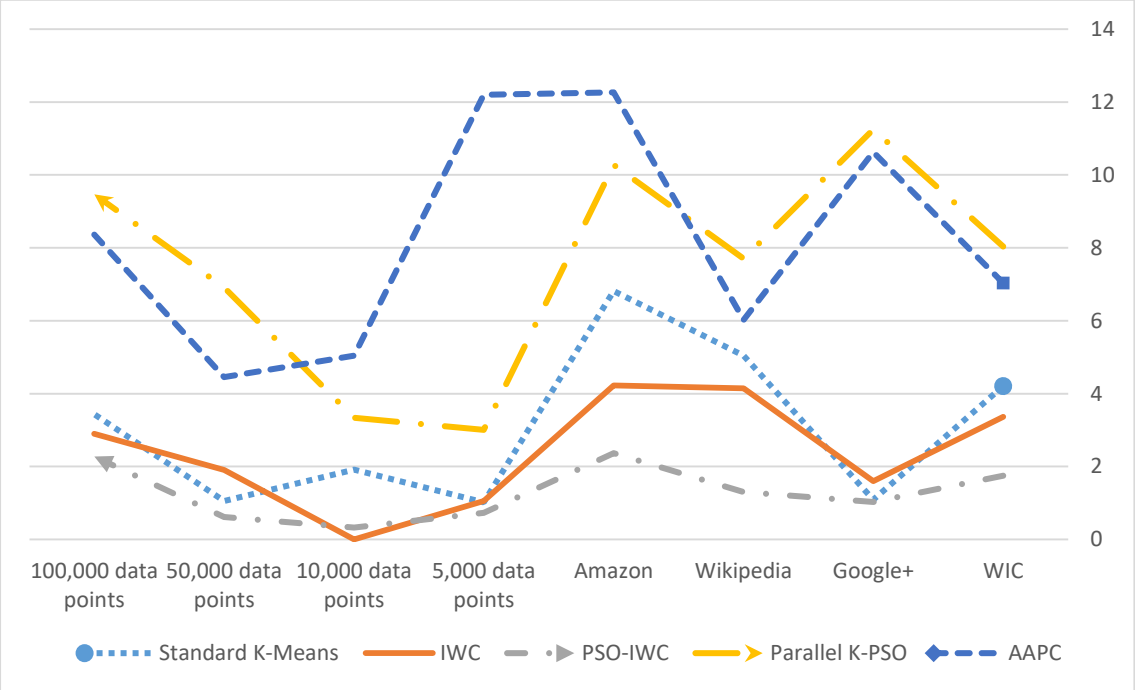


Figure 5.4: Process time of five algorithms

The proposed algorithm was evaluated on its clustering performance using the early mentioned datasets in section 5.1 on a single and multiple nodes clusters. We have performed 80 experimental attempts with all suggested algorithms and using the earlier mentioned datasets. All algorithms were run for 60 iterations.

And regarding the PSO based-on algorithms, swarms (groups of particles) with appropriate numbers were applied, in the setup of IWC-PSO algorithm on Spark's MLlib, tasks were run for 60 iterations to identify the primary `gBest` clusters for the datasets.

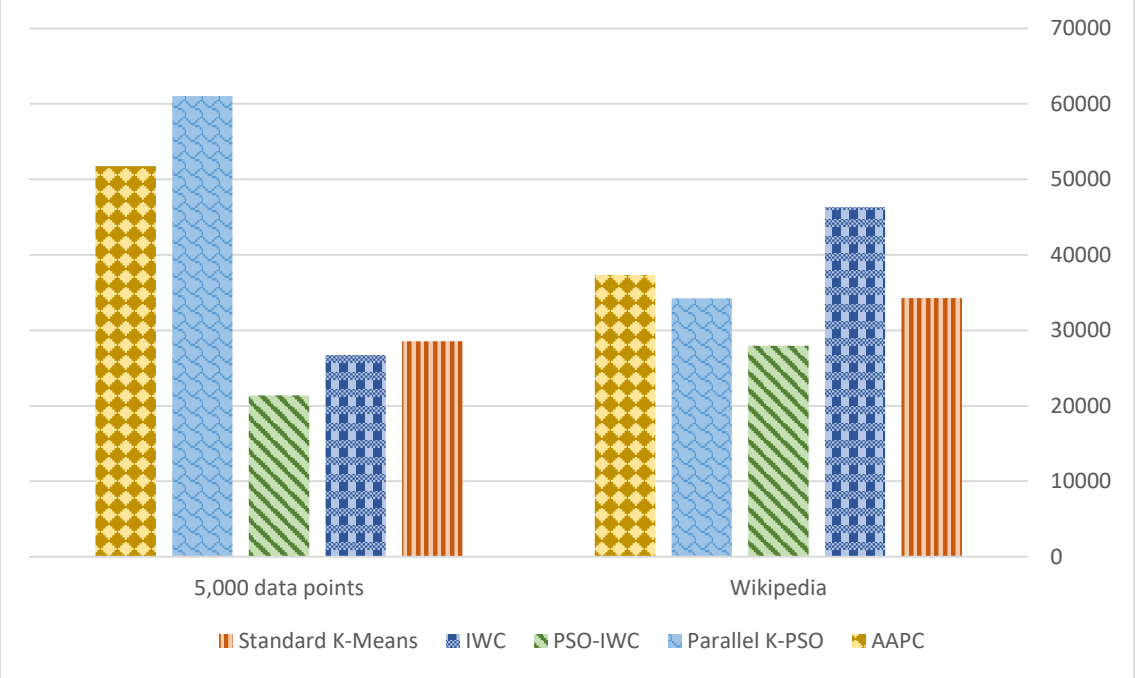


Figure 5.5: Sum of Squared Error on 5,000 and Wikipedia datasets

Figure 5.5 shows the result of sum of squared errors for Wikipedia and 5,000 datasets. We observe that in the Wikipedia dataset the IWC-PSO algorithm performs better clustering results. While in the 5,000 data-points dataset, where the data is likely to be clustered using K-Means algorithm, the IWC-PSO algorithm also achieves a lower error.

Although the K-Means and IWC algorithms run faster in small datasets, which is commonly understood, there is unsound acceleration noticed when rising the calculation size. This is mostly because of in each iteration of K-Means method, it is consider to calculate the center of each cluster in that dataset, while in IWC-PSO method, each cluster center is adjusted regarding to its velocity that's only relies on the other particles in the swarm.

Furthermore, IWC-PSO achieves better speedup results when using different/dynamic numbers of mappers as show in Figure 5.3 compared to serializing or manual assigning the number of mappers. This implies that the utilization of the MapReduce framework is better for the proposed algorithm.

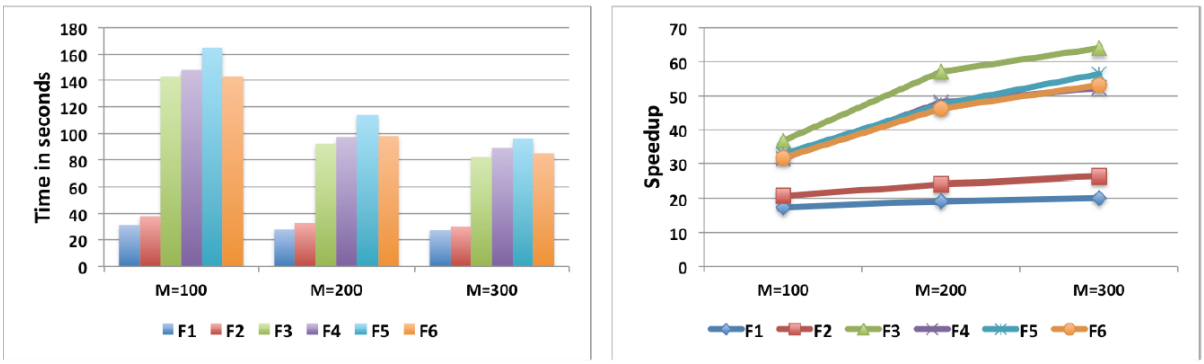


Figure 5.6: Time and speedup for different numbers of mappers

Figure 5.6 (Left) demonstrates the execution time obtained in seconds, while (Right) show the speedup results when using different number of mappers.

Also, SPARK performed dynamic load balancing (number of mappers assigning), and if a task failed, the reassigned map task would complete more quickly. The more processors in use, the greater the need for dynamic load balancing [45].

After analyse the experimental result, we can conclude that the proposed algorithm performance on big data clustering is much better than the others. It also more effective rather modern modifications of K-Means cluster algorithms.

Chapter 6
Conclusion and Further Research

Chapter 6
Conclusion and Further Research

In this research we have studied the improvements to K-Means and IWC clustering algorithms. There have been multiple algorithms which aims to enhance the K-Means method and to work around the limitations of K-Mean. The main goal of this research is to decrease the time consumption of the K-Means and dealing with large scaled datasets. The initial population of the K clusters is one of the major drawback of K-Means which leads to incorrect results. Earlier researches have been directed at solving this issue like [5]. The main purpose in this research is to minimize calculation by inventing a method that can reduce the search area of the initial distribution of the K centroids before calculating the clusters using the IWC algorithm. This is the key to save time and process.

Further, it put into action the outstanding global search capability of particle swarm algorithm and the significant search ability of IWC to improve the validity and robustness of the clustering process, also it exploits the features of MapReduce's parallel computation which makes the algorithm expedited. Further, due to the dynamic change of number of nodes involved in the processing, the method is with high scalability.

The results showed that the proposed algorithm is more effective and efficient in term of clustering time consumption and memory space consumption than traditional K-Means, IWC and AAPC and this was due to the proposed novel techniques used in this hybrid algorithm.

Hereafter, additional adjusted versions of IWC-PSO will be utilized in clustering big datasets. Furthermore, other partitioning algorithms will be used instead of IWC to find out better results.

Also, Future work will investigate the effect of other parameters settings such as numbers of mappers, particles, swarms and iterations on the execution time.

And In order to help to improve the future works, the following recommendations are considered in association to this study.

- **Parallelizing PSO**:

  In order to increase the performance and computational productivity, one of the most important concerns, is whether a serial or parallel approach is applied to change particle locations and velocities. The serial PSO approach changes all particle velocities and locations at the end of every iteration, where on the other hand, the parallel PSO approach changes particle locations and velocities regularly based on on-time available information.

- **Dynamically determining number of iterations:**

  To avoid trapping in a local minimum and boosting the process of finding the optimistic solution that will make a significant improvement in the algorithm, whereas in some datasets the algorithm start stabilizing in just a small number of iterations where in others it would take various number of iterations in order to be stated.

- **Handle multidimensional data processing, at maximum convergence rate and minimum error rate.**

  According the trouble of dimensionality, we empirically noticed that time complexity is directly subjected to the size of the dataset; therefore the future work is aimed at minimizing the complexity of the algorithm while obtaining the better performance.

Nevertheless, the future works will take in consideration a comprehensive investigation about performance attributes, such as convergence characteristics and distributed or parallel design of the proposed methods.

REFERENCES

References

[1] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. Proc of the 6th Symposium on Operating Systems Design and Implementation, Reading, MA, 2004.

[2] J. Pena et. al, "An empirical comparison of four initialization methods for the k-Means algorithm," Pattern Recognition Letters, vol. 20, pp. 1027-1040, 1999.

[3] P.S. Bradley and U. M. Fayyad, "Refining initial points for k-means clustering." In Proceedings Fifteenth International Conference on Machine Learning, pp. 91-99, San Francisco, CA, 1998, Morgan Kaufmann.

[4] Wesam Barbakh and Colin Fyfe, "Inverse Weighted Clustering Algorithm'', IEEE Transaction of Antennas Propagation 52 (3) (2004) pp 771–779.

[5] Carlisle, A. and Dozier, G. An Off-The-Shelf PSO, Proceedings of the 2001 Workshop on Particle Swarm Optimization, pp. 1-6, Indianapolis, IN, 2001.

[6] Kennedy J., Eberhart R. C. and Shi Y.,. Swarm Intelligence, Morgan Kaufmann, New York, 2001.

[7] Weizhong Zhao, Huifang Ma, and Qing He. “Parallel K-Means Clustering Based on MapReduce”. In: Cloud Computing. Ed. by MartinGilje Jaatun, Gansen Zhao, and Chunming Rong. Vol. 5931. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 674–679. URL: http//:dx.doi.org/10.1007/978-3-642-10665-1_71.

[8] H. J. Karloff, S. Suri, and S. Vassilvitskii. A model of computation for mapreduce. In Symposium on Discrete Algorithms (SODA) conference, pages 938–948, 2010.

[9] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. Theoretical Computer Science, 38:293 – 306, 1985.

[10] Ahmed Z. Skaik, Wesam M. Ashour, “Combining IWC and PSO to Enhance Data Clustering”, Journal of Engineering Research and Technology, vol. (4), no. 3, pp. 28 – 42 (2017), ISSN 2312-2307.

[11] Michael G. Noll. Running Hadoop on Ubuntu Linux (Multi-Node Cluster). July 2011. URL: http://www.michael-noll.com/tutorials/running-hadoop-onubuntu-linux-multi-node cluster/.

[12] R. Farivar, D. Rebolledo, E. Chan, and R. Campbell, “A parallel Implementation of K Means clustering on GPUs,” WorldComp 2008, July 2008.

[13] S. Kantabutra, and A. L. Couch, “Parallel K-means clustering algorithm on NOWs,” Technical Journal, vol. 1, no. 6, January-February 2000.

[14] Y. F. Zhang, Z. Y. Xiong, J. L. Mao, and L. Ou, “The study of Parallel K-Means algorithm,” Proceedings of the 6th World Congress on Intelligent Control and Automation, June 2006.

[15] W. Z. Zhao, H. F. Ma, and Q. He, “Parallel K-Means Clustering based on MapReduce,” Lecture Notes in Computer Science, vol. 5931, pp. 674-679, 2009.

[16] J. Dean, and S. Ghenmawat, “MapReduce: Simpllified data processing on large clusters,” Communications of the ACM – 50th anniversary issue:1958-2008, vol. 51, no 1, pp.107-113, January 2008.

[17] Chunne, A.P., Chandrasekhar, U., Malhotra, C.: Real time clustering of tweets using adaptive pso technique and mapreduce. In: Communication Technologies (GCCT), 2015 Global Conference on. pp. 452–457. IEEE (2015)

[18] Mariam El-Tarabily, Rehab Abdel-Kader, Mahmoud Marie, Gamal Abdel-Azeem, “A PSO-Based Subtractive Data Clustering Algorithm,” International Journal of Research in Computer Science eI. SSN 2249-8265 Volume 3 Issue 2 (2013) pp. 1-9.

[19] Sandeep Rana, Sanjay Jasola, and Rajesh Kumar, “A hybrid sequential approach for data clustering using K-Means and particle swarm optimization algorithm,” International Journal of Engineering, Science and Technology Vol. 2, No. 6, 2010, pp. 167-176.

[20] Bara’a Ali Attea, “A fuzzy multi-objective particle swarm optimization for effective data clustering,” Springer-July 2010, pp. 305-312.

[21] K. Premalatha and A.M. Natarajan, “Hybrid PSO and GA for Global Maximization,” ICSRS, Int. J. Open Problems Compt. Math., Vol. 2, No. 4, December 2009, pp. 597-608.

[22] Bo Thiesson, Christopher Meek, and David Heckerman. Accelerating EM for large databases. Machine Learning, 45(3):279,299, 2001.

[23] Osama Abu Abbas “Comparisons between data clustering algorithms”, The International Arab Journal of Information Technology, Vol. 5, No. 3, July 2008.

[24] Preeti Baser, Dr. Jatinderkumar R. Saini “A Comparative Analysis of Various Clustering Techniques used for Very Large Datasets”, International Journal of Computer Science and Communication Networks. 3. 271-275, 2013.

[25] Kehar Singh, Dimple Malik and Naveen Sharma "Evolving limitations in K-means algorithm in data mining and their removal" IJCEM International Journal of Computational Engineering & Management, Vol. 12, April 2011.

[26] Gaikwad, K.S., Patwardhan, M.S.: Tweets clustering: Adaptive pso. In: India Conference (INDICON), 2014 Annual IEEE. pp. 1–6. IEEE (2014)

[27] S. Khan and A. Ahmad, "Cluster center initialization algorithm for k-means clustering," Pattern Recognition Letters, vol. 25, pp. 1293–1302, 2004.

[28] J. Lu. et. al, "Hierarchical initialization approach for k-Means clustering," Pattern Recognition Letters, vol. 29, pp. 787–795, 2008.

[29] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburg, and Angela Hung Byers. Big data: The next frontier for innovation, completion and productivity. Analyst report, McKinsey Global Institute, 2011.

[30] John Gantz and David Reinsel. The DIGITAL UNIVERSE IN 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East. Study report, IDC, December 2012.

[31] R. Hathaway and J. Bezdek, “Extending fuzzy and probabilistic clustering to very large data sets,” Comput. Stat. Data Anal., vol. 51, no. 1, pp. 215–234, 2006. PDF | Big Data Clustering: Jun 25 2018.

[32] M. Usuelli, "An example of MapReduce with rmr2 - MilanoR", MilanoR, 2013.
URL: http://www.milanor.net/blog/an-example-of-mapreduce-with-rmr2/.

[33] M. Usuelli, "An example of MapReduce with rmr2 - MilanoR", MilanoR, 2013.
URL: http://www.milanor.net/blog/an-example-of-mapreduce-with-rmr2/.

[34] A. McNabb, C. Monson, and K. Seppi. "Parallel PSO Using Mapreduce". IEEE Congress on Evolutionary Computation, p1-9, 2007.

[35] H. Karloff, S. Suri, and S. Vassilvitskii, "A model of computation for MapReduce," presented at the Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, Austin, Texas, 2010.

[36] J Kennedy, and RC Eberhart, "Particle Swarm Optimization,'' in Proc. the IEEE International Joint Conference on Neural Networks, Vol. 4, pp 1942–1948, 1995.

[37] S. Cheng, Y. Shi, Q. Qin and R. Bai, "Swarm Intelligence in Big Data Analytics", Intelligent Data Engineering and Automated Learning”, p. 417-426, 2013.

[38] Q. Bai, "Analysis of Particle Swarm Optimization Algorithm", Computer and Information Science, vol. 3, no. 1, p. 5, 2010.

[39] A. S. Foundation, “Apache spark website,” http://spark.apache.org, Mar. 2014.

[40] A. S. Foundation, “Apache spark cluster overview,” May 2014.

URL: http://spark.apache.org/docs/latest/cluster-overview.html.

[41] I. Aljarah, and S. A. Ludwig. "Parallel Particle Swarm Optimization Clustering Algorithm Based On Mapreduce Methodology". 2012 Fourth World Congress on Nature and Biologically Inspired Computing (NaBIC), p104 – 111, 2012.

[42] J. M. Liu, HAN Li-chuan and HOU Li-wen, “Cluster analysis based on particle swarm optimizaiton algorithm,” System Engineering Theory and Practice, vol. 6, June 2005.

[43] J. Wang, D. Yuan, M. Jiang, "Parallel k-pso based on mapreduce", *Communication Technology (ICCT) 2012 IEEE 14th International Conference*, 2012.

[44] URL: http://snap.stanford.edu/data/amazon-meta.html.

[45] Junjun Wang, Dongfeng Yuan, Mingyan Jiang. "Parallel K-PSO based on MapReduce", 2012 IEEE 14th International Conference on Communication Technology, 2012.

[46] R. Lammel, “Google’s MapReduce Programming Model – Revisited,” Science of Computer Programming 70, pp. 1-30, 2008.